Федеральное государственное бюджетное образовательное учреждение высшего образования
«Волгоградский государственный социально-педагогический университет»

На правах рукописи

ШКАРБАН Фатима Витальевна

МЕТОДИКА ОБУЧЕНИЯ ОСНОВАМ ОБЪЕКТНО-ОРИЕНТИРОВАННОГО ПРОГРАММИРОВАНИЯ БАКАЛАВРОВ ПРИКЛАДНОЙ ИНФОРМАТИКИ С ИСПОЛЬЗОВАНИЕМ ВИЗУАЛЬНЫХ УЧЕБНЫХ СРЕД

13.00.02 — теория и методика
обучения и воспитания (информатика)

Диссертация
на соискание ученой степени кандидата
педагогических наук

Научный руководитель:
доктор педагогических наук,
доцент Сергеев Алексей Николаевич

Волгоград, 2018

Оглавление

ВВЕДЕНИЕ

Актуальность исследования. Современное общество, вступившее в информационную эпоху, характеризуется активным применением информационных технологий во всех сферах своей деятельности. Данные технологии, обеспечивая информационную деятельность человека, основываются на применении разнообразной вычислительной техники и программных средств. При этом именно программные средства определяют алгоритмы и логику работы с информацией, позволяют использовать типовую вычислительную технику для решения самых разнообразных задач, в наиболее значительной степени влияют на возможности и качество реализуемых информационных технологий. В этой связи актуальной является проблема обучения созданию программных средств специалистов в области информационных технологий, в частности – обучения программированию бакалавров прикладной информатики.

Обучение бакалавров прикладной информатики, согласно ФГОС ВО, предполагает формирование компетенций обучающихся для решения профессиональных задач в области разработки проектов автоматизации и информатизации прикладных процессов, а также создания информационных систем в прикладных областях. Проектная деятельность бакалавра прикладной информатики при этом включает в себя разработку, внедрение и адаптацию прикладного программного обеспечения, программирование приложений в ходе разработки информационных систем. Реализация данной части обучения бакалавра прикладной информатики осуществляется в рамках дисциплин «Информатика и программирование», «Высокоуровневое программирование», «Программная инженерия» и др., где формируются знания о программных инструментах и языках программирования, умения использовать различные стратегии разработки и реализации компьютерных программ, опыт и навыки решения проблем в области программирования.

Вместе с тем, несмотря на значительное внимание, которое уделяется вопросам обучения программированию бакалавров прикладной информатики, в данной области остается и ряд нерешенных проблем. В частности – проблема обучения основам объектно-ориентированного программирования как методологии, наиболее востребованной при создании программных средств, а также способствующей развитию логического и абстрактного мышления студентов при решении задач и разработке программ.

Традиционное обучение объектно-ориентированному программированию включает в себя, как правило, сложные задачи и вопросы, трудно воспринимаемые студентами, что приводит к дальнейшему непониманию учебного материала. Обучение объектно-ориентированному программированию традиционными методами основано на изучении синтаксиса конкретного языка программирования, который не вызывает интереса среди студентов или является непонятным в овладении. При этом такое обучение на этапе вузовской подготовки опирается на знания и умения обучающихся, полученные при изучении соответствующих разделов информатики в общеобразовательной школе. Как показывают данные многих исследований (А.Н. Бобров, О.Г. Нельзина, М.А. Павличенко, Е.С. Павлова), а также результаты проведенного нами констатирующего эксперимента, этих знаний и умений оказывается недостаточно для последующего изучения профессиональных языков программирования на соответствующем уровне. В связи с этим возникает необходимость изучения основ объектно-ориентированного программирования на этапе до обучения профессиональным языкам программирования для обеспечения фундаментальной подготовки студентов в области самой методологии, базовых концепций объектно-ориентированного программирования. Для этого требуется разработка соответствующей методики обучения основам объектно-ориентированного программирования, основанной на результатах исследований процесса обучения объектно-ориентированному

программированию, использования технологий проектирования и создания современных компьютерных программ.

В науке уже сложились определенные **теоретические предпосылки** для решения задачи обучения бакалавров прикладной информатики объектно-ориентированному программированию. Можно выделить ряд исследований, имеющих особую ценность в данном направлении. В первую группу входят исследования, посвященные проблемам профессиональной подготовки компетентного специалиста – внедрению компетентностного подхода в систему образования (В.И. Байденко, А.Г. Бермус, В.А. Болотов, И.А. Зимняя, М.П. Лапчик, Б.Д. Эльконин и др.), определения компонентов профессиональной компетентности специалистов (В.Е. Андреев, Е.М. Баранова, Л.В. Елагина, А.В. Еремина, И.В. Зороастрова, С.А. Скворцова, Е.О. Сучкова, А.В. Хуторской, и др.), разработки содержания и процесса формирования профессиональной компетентности (Ю.В. Кудинов, Л.Ю. Лазарева, С.Н. Ларин, Е.П. Нехожина, И.Н. Одарич, А.А. Орлов, Л.М. Резник, А.Н. Сакаева, Г.Н. Сериков, Е.А. Синкина и др.). Вторую группу составляют исследования в области профессиональной подготовки специалистов информационных технологий (В.В. Андреев, И.Е. Вострокнутов, Н.К. Нуриев, В.А. Сухомлин, Ю.Ф. Тельнов и др.), формирования профессиональных качеств у обучающихся по направлениям подготовки прикладной информатики (В.Ю. Бодряков, А.А. Быков, М.Г. Гайда, Э.К. Самерханова, Л.Р. Ушакова). Третью группу составляют исследования, раскрывающие сущность обучения программированию (Ф. Брукс, Е. Дейкстра, А.П. Ершов, С. Макконнелл, М.Л. Смульсон, Б. Шнейдерман), теории и методики обучения информатике (И.Е. Вострокнутов, В.Е. Жужжалов, В.А. Касторнова, А.Б. Кузнецов, М.П. Лапчик, Ж.К. Нурбекова, Е.С. Павлова, А.Н. Петров, И.В. Роберт, Н.Г. Саблукова, М.В. Швецкий).

Как видим, проблема, содержание и методы обучения программированию, в том числе – в системе профессионального

образования, рассматривались во многих работах отечественных и зарубежных ученых. Первые отечественные работы и результаты исследований в области обучения программированию были освещены в работах известного российского академика А.П. Ершова, которые послужили базой для создания и всего курса основ информатики и вычислительной техники в Советском Союзе. В диссертации М.В. Швецкого рассматривается содержание учебных дисциплин, связанных с программированием, рассматривается курс «Визуальное программирование» и его взаимосвязи с другими. Монография Ж.К. Нурбековой посвящена методологии обучения программированию, построению методической системы обучения программированию с применением методов информационного и математического моделирования, а также разработке теоретических основ электронного обучения программированию. В исследовании В.Е. Жужжалова определяются основные подходы и принципы, способствующие совершенствованию системы обучения информатике в высших учебных заведениях. В работе А.Н. Петрова отражена проблема обучения объектно-ориентированному программированию и показана необходимость совместного использования визуального языка моделирования, являющегося средством объектно-ориентированного проектирования, а также автоматизированных средств генерирования объектно-ориентированного программного кода с использованием профессиональных языков. Имеются также работы, посвященные обучению визуальному программированию с использованием профессиональных сред. В них, в частности, обосновано, что визуальные среды программирования могут использоваться как средство создания мультимедийных обучающих программ с определенной системой упражнений (В.А. Касторнова), а использование визуальных сред способствует повышению уровня учебной мотивации к программированию (Н.Г. Саблукова).

Одновременно с теоретическими предпосылками сформировались и **практические предпосылки** обучения основам объектно-ориентированного

программирования как начальному этапу профессиональной подготовки специалистов в области разработки компьютерных программ. В частности – это практические наработки, связанные с применением для такого обучения визуальных учебных сред. Среди этих сред особое место занимают Alice и Scratch, созданные в качестве учебных продуктов для обучения программированию. Результаты практического применения этих сред, представленные в трудах Рэнди Пауча, Ванда Данна, Стивена Купера, Митчелла Резника, убедительно показывают, что их использование позволяет избежать многих проблем начального обучения программированию. Визуальные учебные среды позволяют мотивировать обучающихся, концентрировать внимание на развитие мышления, понимание самой сути изучаемых парадигм программирования. Проведенная работа, однако, в большей степени раскрывает пути построения курса программирования для учащихся общеобразовательных школ и недостаточно раскрывают возможности применения данных сред как средства обучения объектно-ориентированному программированию в системе подготовке специалистов, предполагающей также изучение профессиональных языков и сред.

Таким образом, в науке и образовательной практике сложились определенные предпосылки для разработки методики обучения основам объектно-ориентированного программирования бакалавров прикладной информатики, имеются наработки в плане использования визуальных учебных сред, которые позволяют обучать самим концепциям объектно-ориентированного программирования до этапа изучения конкретного языка.

При этом, отмечая наличие большого числа исследований и практического опыта в области обучения программированию, а также высокую ценность проведенных работ, следует констатировать, что проблема изучения объектного подхода на ранних стадиях обучения программированию бакалавров прикладной информатики пока еще не являлась предметом специального научного исследования. Отсутствуют также исследования в области обучения основам объектно-ориентированного

программирования бакалавров прикладной информатики с использованием визуальных учебных сред. В этой связи всё более остро обнаруживаются **противоречия** между:

– потребностью подготовки высококвалифицированных специалистов в области разработки программного обеспечения и недостаточным для освоения профессиональных языков программирования уровнем подготовки обучающихся вузов, приступивших к освоению образовательных программ по направлениям информатики и вычислительной техники;

– усложнением парадигм программирования, увеличением номенклатуры языков программирования, а также средств разработки компьютерных программ и отсутствием адекватных этой ситуации изменений в содержании обучения программированию и существующих методиках;

– наличием практики использования визуальных учебных сред для обучения основам программирования и отсутствием методик их применения при изучении базовых концепций объектно-ориентированного программирования на ранних этапах профессиональной подготовки бакалавров прикладной информатики.

Указанные противоречия обозначили **проблему исследования**, которая заключается в недостаточной разработанности методических основ применения визуальных учебных сред как средства обучения основам объектно-ориентированного программирования бакалавров прикладной информатики. С учетом этого была выбрана **тема исследования**: «Методика обучения основам объектно-ориентированного программирования бакалавров прикладной информатики с использованием визуальных учебных сред».

Объектом исследования является процесс обучения программированию бакалавров прикладной информатики.

Предметом исследования является методика обучения основам объектно-ориентированного программирования бакалавров прикладной информатики с использованием визуальных учебных сред.

Цель исследования заключается в разработке и научном обосновании методики обучения основам объектно-ориентированного программирования бакалавров прикладной информатики с использованием визуальных учебных сред.

Гипотеза исследования заключается в том, что процесс обучения основам объектно-ориентированного программирования бакалавров информатики будет более результативным, если:

1) обучение в области объектно-ориентированного программирования будет рассматриваться приоритетной целью предметной подготовки бакалавра прикладной информатики и включать в себя освоение объектно-ориентированного программирования как самостоятельной области знаний, а также изучение конкретного объектно-ориентированного языка;

2) компетенция бакалавра прикладной информатики в области объектно-ориентированного программирования будет раскрываться как совокупность личностных качеств обучающихся, включающая в свой состав мотивы и потребности к профессиональной разработке компьютерных программ, знания и умения в области ключевых концепций объектно-ориентированного программирования и конкретного объектно-ориентированного языка, способности к оценке собственной деятельности, самоорганизации и самообразованию;

3) методика обучения основам объектно-ориентированного программирования бакалавров прикладной информатики будет предполагать погружение обучающегося в проблематику объектно-ориентированного программирования на самом начальном этапе обучения программированию с использованием визуальных учебных сред для формирования базовых представлений и понятий объектно-ориентированного подхода к разработке компьютерных программ;

4) процесс реализации методики обучения основам объектно-ориентированного программирования бакалавров прикладной информатики, а также оценка успешности ее реализации будут осуществляться в рамках двух взаимосвязанных дисциплин, первая из которых посвящена основам объектно-ориентированного программирования, а вторая – изучению конкретного объектно-ориентированного языка.

Для достижения цели исследования и проверки гипотезы были сформулированы **следующие задачи исследования**:

1. Уточнить роль и место обучения объектно-ориентированному программированию в системе подготовки бакалавров прикладной информатики.

2. Выявить сущностные характеристики компетенции бакалавра прикладной информатики в области объектно-ориентированного программирования.

3. Разработать целевой, содержательный и процессуальный компоненты методики обучения основам объектно-ориентированного программирования бакалавров прикладной информатики с использованием визуальных учебных сред.

4. Экспериментально обосновать эффективность методики обучения основам объектно-ориентированного программирования бакалавров прикладной информатики с использованием визуальных учебных сред.

Теоретико-методологической основой исследования являются:

– исследования в области внедрения компетентностного подхода в систему образования, разработки содержания и процесса формирования профессиональной компетентности специалистов (В.А. Болотов, И.А.Зимняя, М.П. Лапчик, Г.Н. Сериков, и др.);

– фундаментальные исследования в области теории и методики обучения информатике (В.Е. Жужжалов, В.А. Касторнова, А.Б. Кузнецов, М.П. Лапчик, Ж.К. Нурбекова, Е.С. Павлова, А.Н. Петров и др.);

– исследования профессиональной подготовки специалистов в области информационных технологий (В.В. Андреева, Е.В. Бондарева, И.Е. Вострокнутов, А.С. Гиглав, Н.К. Нуриев, В.А. Сухомлин, Ю.Ф. Тельнов и др.);

– труды по формированию профессиональных качеств у обучающихся по направлениям подготовки прикладной информатики (В.Ю. Бодряков, А.А. Быков, М.Г. Гайда, Э.К. Самерханова, Л.Р. Ушакова);

– исследования, раскрывающие сущность обучения программированию в системе общего и профессионального образования (Ф. Брукс, Е. Дейкстра, А.П. Ершов, С. Макконнелл, М.Л. Смульсон, Б. Шнейдерман);

– положения в области организации обучения циклу компьютерных наук в ведущих университетах мира (CIP 2010, Career Space, CC 2016, Software Engineering 2017 и др.);

– опыт зарубежных университетов в области обучения основам объектно-ориентированного программирования с использованием учебных сред визуального программирования (W.P. Dann, S. Fincher, S. Cooper, R. Pausch и др.).

Этапы исследования. Исследование проводилось в 2007–2018 гг. и включало в себя три этапа.

На первом этапе проведен анализ исследований в области обучения основам объектно-ориентированного программирования бакалавров информатики с использованием визуальных учебных сред; определены цели и задачи, сформулирована гипотеза, конкретизированы методы исследования; проведен констатирующий эксперимент.

На втором этапе разработана методика обучения основам объектно-ориентированного программирования бакалавров прикладной информатики с использованием визуальных учебных сред, проводился поисковый эксперимент.

На третьем этапе был проведен формирующий эксперимент, сформулированы выводы и подведены итоги, оформлено диссертационное исследование.

Методы исследования: *теоретические* (анализ, сравнение, классификация, систематизация, обобщение) – анализ научной, психолого-педагогической и методической литературы по проблеме, изучение опыта обучения программированию и особенностей практической подготовки будущих бакалавров прикладной информатики; *эмпирические* (анкетирование, наблюдение, интервьюирование, метод экспертных оценок) – для определения уровня профессиональной подготовки обучающихся вуза; *педагогический эксперимент* – для проверки эффективности предлагаемой методической системы; *методы математической статистики* – для анализа полученных данных, определения количественных показателей по исследуемым явлениям и процессам, проверки гипотезы исследования.

Эмпирическую базу исследования представляют данные опытно-экспериментальной работы, проводившейся в ГБОУ ВО РК «Крымский инженерно-педагогический университет». Всего на разных этапах в исследовании участвовали 208 студентов университета.

Положения, выносимые на защиту:

1. Обучение основам объектно-ориентированного программирования бакалавров прикладной информатики как самостоятельной области знаний включает в себя формирование представлений о ключевых концепциях данной парадигмы, а также знаний, умений и опыта применения конкретного языка программирования для объектно-ориентированной разработки программ. Такое обучение эффективно реализуется на ранних курсах программы подготовки бакалавров прикладной информатики в рамках двух взаимосвязанных дисциплин (введения в объектно-ориентированное программирования, а также объектно-ориентированного программирования с использованием конкретного языка), предваряющих изучение всех других

дисциплин профессионального цикла, посвященных разработке компьютерных программ.

2. Компетенция бакалавров прикладной информатики в области объектно-ориентированного программирования конкретизирует компетенции образовательного стандарта, определяющие способности разрабатывать, внедрять и адаптировать прикладное программное обеспечение в структуре профессиональной компетентности бакалавра прикладной информатики и включает в себя мотивационно-ценностный, организационно-содержательный, когнитивно-операционный, личностно-рефлексивный компоненты. Каждый компонент характеризуется своим содержанием (знания, умения и личностные установки), функцией (роль компонента в организации процесса освоения объектно-ориентированного программирования, а также дальнейшей профессиональной деятельности по разработке компьютерных программ) и характеристикой (индикаторы достижения необходимого уровня образования в процессе подготовки бакалавров прикладной информатики). Обучение основам объектно-ориентированного программирования нацелено на формирование мотивационно-ценностного, организационно-содержательного и личностно-рефлексивного компонентов, а изучение конкретного объектно-ориентированного языка – мотивационно-ценностного, когнитивно-операционного и личностно-рефлексивного компонентов.

3. Методика обучения основам объектно-ориентированного программирования бакалавров прикладной информатики с использованием визуальных учебных сред представляет собой целостную систему проектирования и организации процесса обучения в рамках двух взаимосвязанных дисциплин. Методика раскрывается через описание своих компонент (цели, содержание, средства, методы и формы обучения), соотнесённых с компонентами компетенции бакалавра прикладной информатики в области объектно-ориентированного программирования, а также дисциплинами введения в объектно-ориентированное

программирование и изучения конкретного объектно-ориентированного языка. Основополагающим фактором конкретизации содержательного и процессуального компонентов методики для каждой из двух взаимосвязанных дисциплин является выбор программных средств для обучения бакалавров прикладной информатики основам объектно-ориентированного программирования – визуальных учебных сред Alice и Scratch в рамках дисциплины введения в объектно-ориентированное программирование, а также одной из профессиональных сред программирования (Visual Studio или др.) в рамках дисциплины по изучению конкретного объектно-ориентированного языка.

4. Эффективная реализация методики обучения основам объектно-ориентированного программирования бакалавров прикладной информатики возможна с использованием визуальных учебных сред Alice и Scratch (организация практической работы по изучению базовых принципов и ключевых понятий объектно-ориентированного программирования), педагогических программных средств Piazza и OpenClass (постановка задач для практической работы, выполнение тестовых заданий для самоконтроля, текущей и промежуточной аттестации по основам объектно-ориентированного программирования), а также инструментальных средств Prezi и PowerPoint (подготовка учебно-методических материалов, оформление портфолио обучающихся).

Научная новизна исследования состоит в том, что разработана новая методика обучения объектно-ориентированному программированию бакалавров прикладной информатики с использованием визуальных учебных сред, а именно:

– конкретизировано содержание компетенции бакалавра прикладной информатики в области объектно-ориентированного программирования, включающей в свой состав мотивационно-ценностный, организационно-содержательный, когнитивно-операционный и личностно-рефлексивный компоненты;

– адекватно данной компетенции разработаны целевой, содержательный и процессуальный компоненты методики обучения основам объектно-ориентированного программирования бакалавров прикладной информатики с использованием визуальных учебных сред, реализуемой в рамках двух взаимосвязанных дисциплин – введения в объектно-ориентированное программирование, а также изучения конкретного объектно-ориентированного языка.

Теоретическая значимость результатов исследования обусловлена вкладом в теорию и методику обучения информатике (уровень высшего образования) за счет описания структурных компонент компетенции бакалавров прикладной информатики в области объектно-ориентированного программирования (мотивационно-ценностный, организационно-содержательный, когнитивно-операционный, личностно-рефлексивный), теоретического обоснования компонент методики обучения основам объектно-ориентированного программирования бакалавров прикладной информатики с использованием визуальных учебных сред (цель, содержание, средства, методы, формы).

Теоретические результаты исследования могут служить основой для дальнейших теоретических разработок в области обучения программированию бакалавров и магистров по направлениям и профилям подготовки в области информационных технологий.

Достоверность результатов исследования обеспечивалась обоснованностью исходных теоретико-методологических позиций; репрезентативной выборкой с учетом содержания и характера эксперимента; использованием комплекса методов исследования, адекватных его предмету, задачам, логике; сочетанием опытной и экспериментальной работы; длительным характером опытно-экспериментальной работы по разработке методики обучения основам объектно-ориентированного программирования бакалавров прикладной информатики с использованием визуальных учебных сред.

Личный вклад соискателя состоит в участии во всех этапах работы над диссертационным исследованием, непосредственном участии при получении данных на диагностическом этапе, по окончании формирующего эксперимента и на этапах контрольных срезов; личном участии в разработке теоретических основ обучения основам объектно-ориентированного программирования бакалавров прикладной информатики с использованием визуальных учебных сред, обработке, анализе и интерпретации полученных данных; в подготовке научных статей и докладов по итогам выполненной работы.

Практическая ценность результатов исследования: разработана и внедрена в образовательный процесс методика обучения основам объектно-ориентированного программирования бакалавров прикладной информатики с использованием визуальных учебных сред Alice и Scratch; создано учебно-методическое обеспечение (рабочие программы, комплекты заданий для всех видов занятий, оценочные средства, электронные образовательные ресурсы на платформах Piazza и OpenClass) для реализации разработанной методики в рамках дисциплин «Программирование для начинающих» и «Информатика и программирование».

Разработанное учебно-методическое обеспечение может использоваться преподавателями учреждений высшего образования, реализующих основные профессиональные образовательные программы подготовки бакалавров по направлению 09.03.03 «Прикладная информатика» для обучения основам объектно-ориентированного программирования, а также объектно-ориентированному программированию с использованием конкретного языка.

Апробация результатов исследования осуществлялась через:

– участие в международных научных и научно-практических конференциях: XV научно-теоретическая конференция профессорско-преподавательского состава, аспирантов и студентов Крымского инженерно-педагогического университета (Симферополь, 21–24 апреля 2009 года),

«Информационные технологии в образовании, науке и технике» (ИТОНТ 2010) (Черкассы, 2010), «Современные информационные технологии и инновационные методики обучения в подготовке специалистов: методология, теория, опыт, проблемы (Киев-Винница, 2010), «Компьютерно-ориентированные системы обучения» (Киев, 2010), «Информационно-компьютерные технологии в экономике, образовании и социальной сфере» (Симферополь, 2011-2017), «Pedagogy and Psychology in the age of globalization» (Budapest, 2015), «Pedagogy and Psychology in an Era of increasing from of information» (Budapest, 2015); II Международная научно-практическая конференция «Исследования и разработки в перспективных научных областях»» (Новосибирск, 2017); «II Межрегиональная научно-практическая конференция» (Архангельск, 2018); «в V Международной научно-практической конференции «Электронное обучение в непрерывном образовании 2018»» (Ульяновск, 2018).

– публикацию материалов исследования в различных научных и научно-методических изданиях (всего 28 работ, из них 4 статьи в ведущих рецензируемых научных изданиях, определенных Высшей аттестационной комиссией Минобрнауки России и 2 учебных пособия).

Внедрение результатов исследования осуществлялось в практике подготовки бакалавров прикладной информатики ГБОУ ВО РК «Крымский инженерно-педагогический университет».

Объем и структура диссертации. Диссертация состоит из введения, двух глав, заключения, списка литературы (224 наименования), 17 приложений. Текст диссертации содержит 27 таблиц и 16 рисунков.

ГЛАВА 1. ТЕОРЕТИЧЕСКИЕ ОСНОВЫ ОБУЧЕНИЯ В ОБЛАСТИ ОБЪЕКТНО-ОРИЕНТИРОВАННОГО ПРОГРАММИРОВАНИЯ БАКАЛАВРОВ ПРИКЛАДНОЙ ИНФОРМАТИКИ

1.1. Роль и место обучения программированию в системе подготовки бакалавров прикладной информатики

Современное общество характеризуется рядом особенностей, среди которых одно из наиболее значимых мест занимает утверждение роли информационных технологий как ведущего средства совершенствования интеллектуальной, трудовой, досуговой и многих других видов деятельности человека. В самом общем виде информационные технологии понимаются как процессы, методы поиска, сбора, хранения, обработки, предоставления, распространения информации и способы осуществления таких процессов и методов с использованием средств вычислительной техники. Возможности развития общества напрямую зависят от уровня владения каждым человеком информационных технологий, а также наличия высококвалифицированных специалистов, способных к продуцированию информационных технологий, разработке и внедрению информационных систем, обеспечивающих новый уровень реализации технологических процессов.

Задача подготовки граждан в области информационных технологий как одна из основных решается в системе образования. В нашей стране решение данной задачи связано с реализацией образовательной области информатики на всех уровнях образования.

Системная реализация данного направления началась в 1985 году, когда в структуру школьной и вузовской подготовки был введён курс «Основы информатики и вычислительной техники», направленный на реализацию задачи обеспечения всеобщей компьютерной грамотности молодёжи. Компьютерная грамотность понималась как «владение навыками решения задач с помощью ЭВМ, умение планировать действия и предвидеть их последствия, понимание основных идей информатики, представление о

роли информационных технологий в жизни общества» (А. П. Ершов, [54]), а содержание обучения ориентировалось на программирование.

А.П. Ершов писал, что для эффективного использования возможностей вычислительной техники при любой форме взаимодействия с ней необходимо владеть определённым стилем мышления, определёнными навыками умственных действий. Согласно мнению учёного, эти навыки в наиболее явном виде присущи профессиональным программистам, в связи с чем их формирование связывалось с изучением информатики [52, 53, 55].

Безусловно, с момента начала реализации курс информатики значительно изменился. Так, В.В. Гриншкун отмечает, что первоначальная концепция формирования у обучающихся знаний и умений в области элементарных основ устройства и использования компьютерной техники постепенно изменяется на более фундаментальный подход, в рамках которого при обучении информатике рассматриваются информация, ее типология, общие приемы оперирования информацией, способы выработки адекватного отношения к информации [41, с. 7]. При этом обучение программированию – это не только изучение конкретного языка, но скорее умение составлять алгоритмы различной сложности, что дает возможность изучать разные языки и среды программирования [42, с. 50].

Задача обеспечения компьютерной грамотности, таким образом, последовательно трансформировалась в задачи формирования информационной культуры, а ныне – информационной компетентности обучающихся. Обучение алгоритмизации и программированию является не единственной задачей курса информатики, а входит в этот курс в качестве одной из равноправных линий [152]. Анализируя характер и причины таких изменений, в полной мере можно согласиться с мнением Е. В. Данильчук о том, что динамичность развития предметной области информатики актуализирует вопрос поиска «универсального, инвариантного, что поможет учащемуся самому осваивать «информационное новое» не только «здесь и сейчас», но и завтра, послезавтра и всегда» [47]. Обучение алгоритмизации и

программированию, связанное с формированием навыков мышления высокого уровня, безусловно, относится к инвариантной части обучения информатике.

Обучение программированию входит в образовательные программы различных направлений подготовки в области информационных технологий по стандартам высшего образования. Так, проведя анализ ФГОС ВО по направлениям бакалавриата «Прикладная математика и информатика», «Математика и компьютерные науки», «Фундаментальная информатика и информационные технологии», «Математическое обеспечение и администрирование информационных систем», «Информатика и вычислительная техника», «Информационные системы и технологии», «Прикладная информатика», «Программная инженерия» мы выяснили, что во всех из них в структуре задач профессиональной деятельности выделяются задачи, связанные с алгоритмизацией и программированием – изучением языков программирования, разработкой и анализом алгоритмов, разработкой программного обеспечения и даже новых языков программирования. В зависимости от направлений подготовки, акценты на обучение программированию могут ставиться в большей или в меньшей степени, но в любом случае – в структуре профессиональной компетентности в общепрофессиональной или профессиональной части присутствуют компетенции, связанные с разработкой компьютерных программ.

Согласно справочнику «Вузы России», представленному на Федеральном портале «Российское образование» [38], самым популярным направлением подготовки в области компьютерных наук является направление «Прикладная информатика», программы бакалавриата которого в 2017 году реализуются в 759 высших учебных заведениях Российской Федерации.

Направление подготовки «Прикладная информатика» – это современная подготовка специалистов в области информационных технологий, владеющих навыками разработки программных продуктов,

современными средствами и методами разработки программных систем. Важными для данных специалистов являются компетенции, связанные с применением стандартов программной и системной инженерии, управлением коллективной разработкой программного обеспечения, основными приемами программирования, разработки дизайна программных приложений, администрирования систем.

Связь программ подготовки бакалавров прикладной информатики с профессиональными стандартами в области информационных технологий описывается в новых версиях Федеральных государственных образовательных стандартах высшего образования, называемых как ФГОС ВО 3++. Эта связь представлена в таблице 1, из которой видно, что компетенции, связанные с программированием и руководством разработкой программного обеспечения, формируются на уровнях как бакалавриата, так и магистратуры прикладной информатики. При этом подготовка в области непосредственно программирования более актуальна для уровня бакалавриата, как именно с ним связан профессиональный стандарт «Программист».

Таблица 1

Связь образовательных программ прикладной информатики и профессиональных стандартов в области информационных технологий

Направление подготовки	**Наименование профессионального стандарта**	**Код профессионального стандарта**
09.03.03 «Прикладная информатика» (бакалавриат)	Программист	06.001
	Специалист по информационным системам	06.015
	Руководитель проектов в области информационных технологий	06.016
	Руководитель разработки программного обеспечения	06.017
	Системный аналитик	06.022
09.04.03 «Прикладная информатика» (магистратура)	Менеджер по информационным технологиям	06.014
	Специалист по информационным системам	06.015
	Руководитель проектов в области информационных технологий	06.016
	Руководитель разработки	06.017

Направление подготовки	Наименование профессионального стандарта	Код профессионального стандарта
	программного обеспечения	
	Системный аналитик	06.022

В программах подготовки бакалавров прикладной информатики предусматриваются дисциплины программирования, изучение которых, как правило, предполагается на первом и втором году обучения. Эти дисциплины включают разделы структурного и объектно-ориентированного программирования. Полученные в рамках этих дисциплин знания и умения используются в дальнейшем при изучении других профессиональных дисциплин («Программная инженерия» и др.), прохождении учебных и производственных практик, подготовки курсовых и выпускных квалификационных работ.

Современная динамика трансформации высшего образования России требует опережающего развития национальной системы образования и науки в целом. Исходя из этого, проведя описанный выше анализ подготовки специалистов в области информационных технологий в нашей стране, целесообразно обратиться и к сравнительному анализу обучения дисциплинам из цикла компьютерных наук в разных странах мира, где стремительные экономические и технологические процессы приводят к постоянной модификации и совершенствованию программ и методов обучения.

Так, образовательные стандарты США регулируются документом – «Классификатор образовательных программ» (Classification of Instructional Programs – CIP 2010), разработанным Национальным центром образовательной статистики США [219].

Классификация учебных программ (CIP) – это таксономия академических дисциплин в учреждениях высшего образования в Соединенных Штатах и Канаде. CIP первоначально была разработана Национальным центром статистики образования (NCES) Соединенных

Штатов Америки и Департамента образования в 1980 году. Позже была пересмотрена в 1985, 1990, 2000 и 2010. Издание 2016 года (CIP 2016) является текущей версией таксономии.

Каждая программа классификатора CIP 2016 представлена в виде следующей схемы:

– шифр и название программы;

– краткое содержание;

– перекрестные ссылки на коды и названия соответствующих профессий по четырем профессиональным классификационным системам;

– коды и названия соответствующих кластеров в индустрии (Industry Cluster) по классификатору National Skill Standards Board (NSSB);

– коды и названия соответствующих карьерных кластеров (Career Cluster) по классификации Департамента образования (Department of Education).

К программам CIP 2016, касающимся области компьютерных наук, можно отнести программы из области обучения «Компьютерные и информационные науки и услуги поддержки». Это учебные программы, ориентированные на компьютерные и информационные науки, а также подготовку специалистов к различным занятиям в области информационных технологий и компьютерных операций (табл. 2) [205]:

Таблица 2

Область обучения:

«Компьютерные и информационные науки и услуги поддержки»

Код	**Подсерии**
11.01	Компьютерные и информационные науки и вспомогательные службы, общие
11.02	Компьютерное программирование
11.03	Технология обработки данных и обработки данных / техник
11.04	Информационные науки / исследования
11.05	Анализ компьютерных систем / аналитик
11.06	Приложения ввода / микрокомпьютера

Код	Подсерии
11.07	Информатика
11.08	Компьютерное программное обеспечение и мультимедийные приложения
11.09	Компьютерные сети и телекоммуникации
11.10	Управление и управление компьютерами / информационными технологиями
11.99	Компьютерные и информационные науки и вспомогательные службы, другие

Как видим из таблицы, в области «Компьютерные и информационные науки и услуги поддержки» присутствует подсерия, непосредственно относящаяся к обучения программированию – 11.02 «Компьютерное программирование» [206]. Данная подсерия содержит классы учебных программ с 11.0201 по 11.0299 (табл. 3), которые ориентированы на компьютерные и информационные науки и готовят специалистов к различным занятиям в области информационных технологий и компьютерных операций [136].

Таблица 3

Характеристика подсерии 11.02 «Компьютерное программирование»

Наименование класса учебной программы	Характеристика
11.0201 Компьютерное программирование / программист, общее	Данный учебный класс включает программы, которые фокусируются на общем написании и реализации общих и настраиваемых программ для управления операционными системами, которые в целом подготавливают специалистов к разработке программного обеспечения и программированию для установки и обслуживания программного обеспечения. Эти программы включают курсы по разработке программного обеспечения: языки с низким и высоким уровнем и программирование; программы настройка и отладки; тестирование прототипов; исправление проблем; связанные с программированием аспекты операционных систем и сетей. Примеры программ: • разработка компьютерных приложений; • компьютерное программирование; • технологии компьютерного программирования; • программирование; • разработка программного обеспечения.
11.0202 Компьютерное программирование,	Данный учебный программный класс включает программы, которые подготавливают специалистов к применению знаний и

Наименование класса учебной программы	Характеристика
специальные приложения	навыков общего компьютерного программирования для решения конкретных операционных задач и требований к настройке программного обеспечения. Включает в себя обучение конкретным видам программного обеспечения, его установке и обслуживании. Примеры программ: • программирование бизнес-приложений; • бизнес-компьютерное программирование; • компьютерное программирование, бизнес-приложения; • разработка приложений для электронной коммерции.
11.0203 Компьютерное программирование, сертификация поставщика / продукта	Данный учебный программный класс включает программы, которые подготавливают специалистов к выполнению требований, сертифицированных специалистов по установке, настройке и обслуживанию для конкретных программных продуктов и процессов. Включает обучение конкретным программным продуктам, поддерживающим поставщиков, установка и их обслуживание. Примеры программ: • Сертификация CISCO; • Cisco Certified Internetwork Expert; • Cisco Certified Network Associate; • Cisco Certified Network Professional; • Сертификация Microsoft; • Microsoft Certified Systems Administrator; • Сертифицированный системный инженер Microsoft; • Microsoft Certified Technology Specialist; • Java-программист, сертифицированный Sun.
11.0299 Компьютерное программирование, другое	Этот учебный программный класс включает программы, не указанные выше, но которые относятся к компьютерному программированию.

Следует отметить, что университеты США при подготовке студентов компьютерных специальностей ориентированы на исследовательскую работу студентов. Основной целью является обучение всех студентов на уровне, достаточном для последующей профессиональной или научной карьеры. Большинство курсов, прослушанных студентами университетов в США и Канаде, как правило, находятся за пределами их основной специализации.

Проблема подготовки специалистов в области информационных технологий является актуальной и в странах Европы, где она контролируется европейской организацией Career Space [223]. Career Space была создана при поддержке Европейской комиссии (European Commission), девяти ведущих

европейских компаний отрасли информационных технологий (BT Group PLC [137], Cisco Systems [138], IBM Europe [139], Intel [140], Microsoft Europe [141], Nokia [142], Philips Semiconductors [143], Siemens AG [144], Thales [145]), а также ассоциацией EICTA (European Information, Communications and Consumer Electronics Industry Technology Association) [146]. Деятельность данной организации направлена на формулирование требований к компетенциям выпускников университетов в соответствующих образовательных стандартах. Эти требования формулируются на основе анализа результатов экспертного опроса руководителей ведущих компаний, работающих в секторе информационных и коммуникационных технологий.

Консорциум Career Space выделил следующие наиболее актуальные области информационных технологий [146]:

1. Telecommunication – Телекоммуникация: радиотехника (Radio Frequency (RF) Engineering); САПР (Digital Design); технологии электросвязи (Data Communications Engineering); разработка приложений для обработки сигналов (Digital Signal Processing Applications Design); разработка телекоммуникационных сетей (Communications Network Design).

2. Software & Services – Программное обеспечение и IT-сервисы: программное обеспечение и приложения (Software & Applications Development); разработка и архитектура программного обеспечения (Software Architecture and Design); разработка мультимедийных технологий (Multimedia Design); консалтинг в области IT-бизнеса (IT Business Consultancy); техническая поддержка IT-систем (Technical Support).

3. Products & System – Создание продуктов и систем ИКТ: разработка продуктов ИКТ (Product Design); системная интеграция (Integration & Test / Implementation & Test Engineering); проектирование систем (Systems Specialist).

4. Cross Sector – Междисциплинарный сектор: управление маркетингом в области ИКТ (ICT Marketing Management); управление ИКТ-проектами (ICT Project Management); исследование и разработка новых технологий

(Research and Technology Development); администрирование ИКТ-систем (ICT Management); управление продажами ИКТ-продуктов (ICT Sales Management).

Для построения эффективной системы подготовки высококвалифицированных кадров в области информационных технологий члены консорциума разработали спецификацию профилей в соответствии с ключевыми специализациями (Generic Skills Profiles: Future Skills for Tomorrow's World [222]), а также при участии двадцати ведущих университетов разработали руководства по проектированию соответствующих учебных курсов для подготовки бакалавров (Curriculum Development Guidelines [210]). В моделях соответствующих образовательных программ предусмотрены научная и технологическая подготовка, развитие системного мышления, изучение прикладных технологий, развитие личностных и деловых качеств.

Среди мировых проектов, предлагаемых по преподаванию информатики, можно выделить также проект Computing Curricula, применяемый в сфере подготовки ИТ-специалистов разного уровня. Рекомендационные документы, разработанные совместно с профессиональными обществами программистов ACM и IEEE Computer Society, отражают современные подходы по преподаванию информатики [210]. Они содержат ряд рекомендаций для университетских программ в области информатики, где для каждой области определен минимальный уровень обязательных знаний, также имеется факультативный материал.

Так, разработчики документов Computing Curricula 2016 [209] и Software Engineering 2017 [221] предлагают набор требований в виде списка разделов и тем, которые должен охватить курс по основам программирования:

– основные конструкции программирования;
– алгоритмы и решение задач;
– фундаментальные структуры данных;

– основы анализа алгоритмов;
– фундаментальные вычислительные алгоритмы;
– алгоритмические стратегии;
– проектирование программного обеспечения;
– спецификации и требования к программному обеспечению;
– проверка соответствия программного обеспечения.

Язык программирования является для программиста основой его деятельности. Однако, знаний одного языка недостаточно, специалист должен понимать различные парадигмы программирования. В стандарте Computing Curricula 2013 в области «Языки программирования» содержится одиннадцать разделов, среди которых шесть являются фундаментальными. Например, для изучения обязательного раздела «Объектно-ориентированное программирование» выделяется половина времени, отведенного на изучение всей области «Языки программирования» [208, с. 65].

Что касается процесса обучения программированию, то в документах Computing Curricula 2016 предлагается несколько стратегий для составления учебного плана подготовки специалистов компьютерного направления, а сами учебные курсы разработчики предлагают разбить на три уровня: вводные курсы, основные курсы и углубленные курсы. К вступительным курсам относятся дисциплины, которые читаются на первом и втором курсах обучения. Основными курсами являются дисциплины второго и третьего годов обучения, которые закладывают фундамент для дальнейшего обучения по специальности. Углубленные курсы направлены на темы, требующие значительной предварительной подготовки на ранних курсах, которые подразделяют на обязательные и выборочные [209].

В аспекте проблематики нашего исследования значительный интерес представляет структура курсов обучения программированию на вводном уровне. Так, для обучения программированию в рамках вводного курса авторы документа Computing Curricula 2016 предлагают использовать шесть стратегий обучения с элементами:

1) императивного программирования;

2) объектно-ориентированного программирования;

3) функционального программирования,

4) максимального охвата материала;

5) ориентации на алгоритмы;

6) ориентация на аппаратную составляющую.

Рассмотрим более подробно данные стратегии.

Обучение программированию с элементами *императивного программирования* – это традиционный подход, который подразумевает изучение одного процедурного языка программирования. Но разработчики программы отмечают, что можно также использовать и объектно-ориентированный язык программирования – если обучение ведётся с использованием объектно-ориентированного языка программирования, то на первом этапе внимание должно концентрироваться на императивных аспектах этого языка (выражениях, структурах управления, процедурах и функциях и других ключевых элементах традиционной процедурной модели), а технологии объектно-ориентированного программирования изучаются на следующем этапе [209]. По мнению М.Е. Зверинцевой, Т.В. Зверинцевой [61], при использовании данной модели обучения, дополнительное внимание изучению объектно-ориентированным темам в последующих курсах все-таки требуется уделять [61].

Ориентация обучения на *объектно-ориентированное программирование* также концентрируется на программировании, но при этом с самого начала делается акцент на принципах объектно-ориентированного проектирования и программирования. При объектно-ориентированном подходе к обучению программированию осуществляется раннее знакомство с принципами объектно-ориентированного программирования, которое, как отмечают авторы Computing Curricula 2016, в последние годы стало чрезвычайно актуальным и важным как для академической среды, так и для промышленности.

Подход с ориентацией на *функциональное программирование* подразумевает использование функциональных языков программирования Lisp, Scheme. Данная модель направлена на изучение и знакомство с языком программирования только в высших учебных заведениях и не зависит от предварительной подготовки студента. Понятный синтаксис функциональных языков позволяет преподавателю акцентировать внимание на фундаментальных принципах и вопросах программирования. В функциональном программировании вся совокупность последовательных состояний вычислительного процесса представляется явным образом, что позволяет изучать эти вопросы в самом начале обучения программированию [106].

Обучение программированию с элементами *максимального охвата материала* позволяют студентам получить более широкое представление о программировании, что дает им возможность двигаться дальше, уверенно и целенаправленно, овладевая другими компьютерными дисциплинами. Студенты во время обучения имеют возможность перед началом курса оценить разнообразие тем учебной программы дисциплины и только после этого приступить к традиционному циклу изучения программирования. Или же наоборот – можно предложить обзор дисциплины уже после завершения вводного цикла по программированию [58].

Подход с *ориентацией на алгоритмы* при изучении программирования минимизирует усилия, затрачиваемые на изучение специфических синтаксических конструкций конкретного языка программирования, за счет использования графического способа описания алгоритмов и псевдокода. Это дает возможность студентам работать с широким диапазоном типов данных и структур управляющей логики. После того, как студенты овладеют основными видами алгоритмов и типами данных, они могут начинать использовать язык программирования на практике [17].

Обучение программированию с элементами *аппаратной составляющей* – это обучение с машинного уровня. Только после

формирования у студентов понимания структурных особенностей аппаратной составляющей, машинной логики и математики, курс переходит к рассмотрению программирования на языках высокого уровня [58].

Проанализировав современные стратегии обучения программированию, в данном исследовании будем исходить из того, что ориентация обучения на объектно-ориентированное программирование отличается от всех рассмотренных моделей и обеспечивает ряд преимуществ, которые другие модели не предусматривают. Объектно-ориентированному программированию характерно сочетание фундаментальности и глубокой практической направленности, что позволяет сформировать у студентов объектно-ориентированный стиль алгоритмического мышления, четкую последовательность действий, образное и логическое мышление, а также качества и умения, позволяющие вести непосредственную разработку программных приложений.

Объектно-ориентированное программирование – это технология программирования, в которой составляющие программы представлены в виде совокупности объектов. Под объектом понимается реальный компонент задачи, который моделирует часть окружающей действительности и которому характерны определенные функции и свойства. Каждый объект является реализацией некоторого класса, образующий иерархию по принципу наследования. Операции, которые можно применять к объектам называются методами. Каждому объекту характерен собственный набор методов.

Как видим, обучение основам объектно-ориентированного программирования, введение основных понятий и концепций объектно-ориентированного подхода – это важные проблемы подготовки бакалавров прикладной информатики. В связи с этим возникает необходимость в разработке соответствующей методики обучения, нацеленной на формирование профессиональной компетенции в области объектно-ориентированного программирования и реализующей одну из

содержательных линий информатики в профессиональной подготовке бакалавров прикладной информатики. Такая методика должна опираться на использование специализированных программных средств и предполагать в итоге изучение профессионального языка.

Выбор языка программирования, с которого следует начинать обучение в системе профессионального образования, является важной задачей, стоящей перед университетами в нашей стране и за рубежом. Задача не является простой, ведь, с одной стороны, нужно учитывать отсутствие теоретической и практической базы у студента-первокурсника, а с другой – необходимость дальнейшего изучения более сложных дисциплин, входящих в состав той или иной специальности в области компьютерных наук.

Исследуя проблему выбора языка программирования, З.С. Сейдаметова и В.А. Темненко ссылаются на рейтинг языков программирования, составленный согласно индексу TIOBE, который определяет частоту запросов в поисковых системах [150]. К числу наиболее популярных языков в данном рейтинге относятся Java, C, C++, Python, C#, PHP, JavaScript. При этом в рейтинге парадигм программирования, также составляемом TIOBE, безусловным лидером является объектно-ориентированный подход [224].

Другой проект по определению рейтинга языков программирования – Ohloh [207], который определяется по критерию частоты упоминания языка программирования в сети Интернет. Согласно этому рейтингу, лидируют такие языки, как C (C++), Java, JavaScript, Objective-C, PHP, Python.

Исследования популярности языков программирования проводятся не только по частоте запросов или их упоминания в Сети – существуют и рейтинги, основанные на экспертных оценках. К таким рейтингам относится, например, рейтинг языков программирования от ассоциации программистов DOU. Рейтинг составлен по результатам опроса, в котором приняли участие 6181 человек [128]. К числу наиболее популярных языков экспертами отнесены Java, C#, JavaScript, PHP, Python, C++, Ruby, Objective-C.

Качественную оценку языков программирования предприняла компания Google, которая провела исследования некоторых характеристик языков программирования, в которые вошли C++, Java, Scala и их собственный язык Go [214]. В исследовании учитывались особенности языка, сложность кода, время компиляции, размер двоичного кода, время работы, объем используемой памяти. Исследования указали на лидирующее положение языка C++, хотя он оказался наиболее сложным с точки зрения оптимизации программного кода.

К программным средствам обучения объектно-ориентированному программированию следует относить не только профессиональные инструментальные среды, но и специализированные визуальные учебные среды, которые, являясь интегрированной средой разработки с простым интерфейсом, позволяют наглядно представлять модель создаваемой программы, выполнять методы классов объектов в процессе их написания, тестировать классы объектов, задавая различные параметры для методов по мере их написания. К таким средам, наиболее известным в академической среде, относятся Alice и Scratch. Помимо них могут использоваться также Dr.Java, Greenfoot, BlueJ, NetBeans и др. Основные характеристики этих сред приводятся в таблице 4.

Таблица 4

Характеристика визуальных сред обучения программированию

Среда	**Особенности**	**Концепции обучения**
Scratch	Простейший инструмент программирования, позволяющий создавать программу путем подбора и совмещения графических блоков, представляющих данные и структуры управления.	Позволяет вести обучение в области основ императивного и объектно-ориентированного программирования: – последовательные процессы; – циклические процессы; – разветвление; – глобальные и локальные переменные, предоставление и изменение величин переменных; – типы данных: символьные, числовые, логические, графические, аудио; – выражения (числовые, текстовые, логические, сравнения), операции, функции, операторы;

Среда	Особенности	Концепции обучения
		– ввод и вывод данных; – координация, синхронизация работы отдельных частей программы; – параллельные процессы – одновременное выполнение различных программных блоков; – объекты, свойства объектов, методы, события.
Alice	Простейший инструмент программирования. Код программы представляет собой набор вложенных блоков, выделенных цветом в зависимости от типа (циклы, условные переходы и др.). Языком программирования в Alice является язык Java. Среда разработки Alice позволяет отображать созданную программу как в стиле языка Java, так и в стиле языка Smalltalk.	Обучение всем основным конструкциям языка, необходимым для освоения основ программирования (операторы ветвления, циклы, использование переменных, функций, рекурсий, методов и параметров и т.п.). Позволяет вести обучение основам объектно-ориентированного программирования с помощью манипулирования объектами их свойств, функций и методов, как встроенных, так и сконструированных пользователем.
Dr.Java	Простой инструмент программирования. Поддерживает интерактивный ввод кода и классы Media вычислений.	Поддерживает все концепции Alice без учета дизайна пользовательского интерфейса. Позволяет вести обучение на основе манипуляций с аудио и видео, работы с видео через классы Media вычислений.
Greenfoot	Простой инструмент программирования. Инструменты визуализации и взаимодействия встроены в среду. Программы написаны на стандартном языке программирования Java.	Поддерживает все концепции Alice. Актеры программируются с помощью текстового кода Java, обеспечивая сочетание опыта программирования с визуальным исполнением.
BlueJ	Простой инструмент программирования, представленный в виде структуры классов в графическом виде. Объекты можно создавать и	Позволяет вести обучение с помощью разработки программ в терминах объектов, классов и их взаимодействий.

Среда	Особенности	Концепции обучения
	тестировать интерактивно.	
NetBeans	Комплекс инструментов программирования, с открытым исходным кодом, предназначенный для обеспечения надежных продуктов разработки программного обеспечения (NetBeans IDE и платформы NetBeans).	Свободная интегрированная среда разработки приложений, позволяющая вести обучение на языках программирования: – Java; – Ruby; – Groovy; – Python; – PHP; – JavaScript.

Как видно из представленной таблицы, именно Alice и Scratch позволяют работать с объектами и методами, а также вести разработку программ в терминах объектов, классов и их взаимодействий. Эти среды ориентированы на освоение фундаментальных концепций объектно-ориентированного программирования, закрепление их на практике программирования с использованием конкретного языка, например C++. Это позволило нам выбрать именно эти среды для начального обучения объектно-ориентированному программированию бакалавров прикладной информатики.

Проект Alice был создан в начале 1990-х годов под руководством доктора Рэнди Пауча (Dr. Randy Pausch) и группы исследователей-разработчиков в Университете Вирджинии. В конце 1990-х и начале 2000-х годов этот проект продолжил развитие с участием Ванда Данна (Wanda Dann) и Стивена Купера (Stephen Cooper), которые вели свою работу в университете Карнеги-Меллона. Проект Scratch был создан под руководством Митчелла Резника с MIT Media Lab в 2003 году.

Учебные среды Alice и Scratch, разработанные специально для обучения программированию, поддерживают объектно-ориентированную парадигму: с одной стороны, программа представляет собой набор команд,

которые выполняются последовательно одна за другой, с другой стороны программа представляет собой описание свойств и поведение отдельных объектов, которые могут взаимодействовать между собой. Объектами являются персонажи мультипликационных роликов, камера и источник света, для манипулирования которыми пользователю доступны их свойства и методы.

Alice и Scratch позволяют манипулировать объектами с помощью их свойств, функций и методов, как встроенных, так и сконструированных пользователем. Данные визуальные учебные среды максимально сокращают ввод данных с клавиатуры. Код программы не является текстом в привычном понимании: в пределах одного метода – это набор вложенных блоков, выделенных цветом в зависимости от типа (циклы, условные переходы и др.), их можно сворачивать, перетаскивать, удалять, менять порядок, и т.д. А среда их разработки позволяет отображать созданную программу как в стиле языка Java (в случае Alice), так и в стиле языка Smalltalk (в случае Alice и Scratch) [216].

Исследования, проводимые группой ученых (B. Moskal, D. Lurie, S.B. Cooper) показали высокий результат применения указанных сред при обучении программированию. Они зафиксировали повышение уровня оценок студентов и усвоения информации в области программирования с 47% до 88% по направлению подготовки «Компьютерные науки» [218].

Оценивая эффективность предлагаемого подхода, один разработчиков визуальной среды программирования Scratch М. Резник, отметил, что ее использование в процессе обучения позволяет:

– воображать и представлять, что именно можно сделать и получить в результате программирования;

– создать проект, основанный на собственных представлениях;

– играть с результатами собственной деятельности;

– делиться результатами своей деятельности;

– обдумывать и обсуждать собственные результаты;

– разработка более сложных и интересных проектов [220].

По мнению О.А. Феоктистовой и М.В. Храмовой Scratch позволяет реализовать научно-познавательную деятельность учащихся, как проектную, так и самостоятельную с использованием методов научного исследования [168]. В.А. Векслер в своей работе отмечает, что для создания программных проектов Scratch имеет все необходимые средства: полноценный язык программирования высокого уровня и объектно-ориентированного типа, интерпретатор языка, графический редактор, систему оказания помощи пользователю, большой набор примеров готовых проектов по разнообразным тематикам, каталоги рисунков и звуковых файлов [31].

Важной особенностью использования Alice и Scratch для обучения основам объектно-ориентированного программирования является наличие визуального представления кода программы и готовых шаблонов управляющих конструкций, которые можно добавлять в свою программу методом перетаскивания. Например, в Scratch шаблон каждой языковой конструкции визуально похож на элемент пазла или конструктора «Лего» (рис. 1).

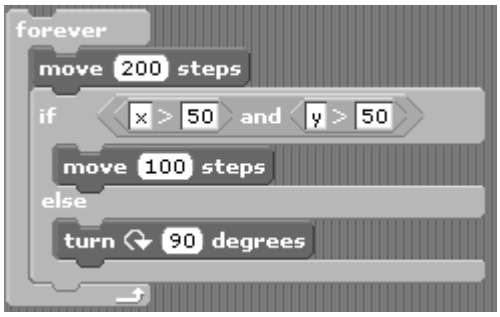


Рисунок 1. Фрагмент программы в среде Scratch

Таким образом, готовая программа создает впечатление правильно составленных частей одного целого, исключается возможность появления в программе синтаксических ошибок. Следовательно, на начальном этапе обучения основам объектно-ориентированного программирования нет необходимости запоминать синтаксис множества команд, внимание

сосредотачивается на понимании логики работы алгоритма, формирование умения определять, в какой ситуации, какая из доступных конструкций является наиболее подходящей. Автоматическое оформление текста программы, соблюдение отступлений строк кода, обозначающих вложенность языковых конструкций, делает привычным и понятным грамотное оформление кода и на других языках программирования.

Кроме этого, визуальные среды для обучения программированию формируют навыки будущего программиста по работе в команде – студенты в командах обсуждают проблему решения задачи, определяют уровень собственных знаний и готовности к решению проблемы, проявляют недостаток знаний в определенных вопросах.

Как указывает М.С. Можаров, в процессе командной работы с использованием визуальных учебных сред студенты учатся оценивать проблему и собственный уровень подготовки, анализировать проблему, искать возможные варианты решения проблем, формулировать и выражать свои мысли, убеждать коллег, делиться знаниями друг с другом и коллективно работать над решением проблемы, планировать свою деятельность, координировать свои действия с членами команды, представлять результаты своей работы [90]. Практический опыт в данном исследовании показывает, что студенты, используя на практике визуальные учебные среды программирования, способны самостоятельно и достаточно эффективно получать знания и формировать умения и навыки. Это способствует формированию целостной системы профессиональных компетенций бакалавров прикладной информатики.

Подытоживая вышесказанное, можно сделать вывод о том, что программирование является ключевым компонентом системы подготовки бакалавров прикладной информатики, главной задачей которого является обеспечение теоретического базиса и практических умений и навыков в профессиональной подготовке студентов. Поэтому знание программирования является необходимым условием для усвоения большинства

профессиональных дисциплин программ подготовки бакалавров прикладной информатики, поскольку:

– эта область знаний состоит из тех концепций, которые важны для практики программирования независимо от парадигмы программирования, который используется в учебном процессе;

– успешное усвоение студентами содержания профессиональных дисциплин и, как следствие, их будущая профессиональная деятельность зависит от качества усвоения учебного материала курса по программированию;

– программирование должно изучаться на младших курсах, перед изучением всех дисциплин профессионального цикла;

– обучение программированию бакалавров прикладной информатики целесообразно начать с изучения языка C++, используя при этом и специальные среды для обучения программированию, такие, как Alice и Scratch. Такое обучение с самого начала должно ориентироваться на основы объектно-ориентированного программирования, так как это дает возможность вывести преподавание курса программирования на более современный уровень и расширяет возможности будущих профильных курсов, а также устраняет противоречия между современным состоянием программирования как науки и содержанием преподавания этой дисциплины в высших учебных заведениях.

При этом под основами объектно-ориентированного программирования мы считаем необходимым понимать самостоятельную область знаний, состоящую из важнейших концепций объектно-ориентированного программирования, которые важны для теории и практики разработки компьютерных программ. Обучение основам объектно-ориентированного программирования бакалавров прикладной информатики должно осуществляться на ранних курсах и в рамках самостоятельной дисциплины, взаимосвязанной с курсом технологий программирования на конкретном языке и предваряющей изучение всех других дисциплин

профессионального цикла, посвященных разработке компьютерных программ. Данное положение, выдвинутое нами в качестве гипотезы, будет более подробно раскрыто и проверено в последующих разделах диссертационной работы.

1.2. Компетенция бакалавра прикладной информатики в области объектно-ориентированного программирования

Определение роли и места обучения программированию в системе подготовки бакалавров прикладной информатики, анализ зарубежных стандартов и программ в области организации учебного процесса в ведущих университетах мира (см. п. 1.1) свидетельствует о росте требований к будущим IT-специалистам. Согласно современным подходам, бакалавр прикладной информатики должен обладать фундаментальными знаниями, профессиональными умениями и навыками деятельности своего профиля, опытом творческой и исследовательской деятельности решения профессиональных задач, что формируется в процессе обучения в вузе – в процессе профессиональной подготовки бакалавра прикладной информатики.

Как отмечает группа исследователей [50], современное образование является процессом и результатом целенаправленного формирования определенных знаний, умений, методологической культуры, комплексной подготовки специалистов через содержание и методы обучения. Главное назначение процесса профессиональной подготовки – сформировать осознанное стремление студента к самопознанию, познанию мира и его проблем, к продуктивной профессиональной деятельности.

Профессиональная подготовка имеет целью ускоренное приобретение обучающимися навыков, необходимых для выполнения определенной работы, группы работ [167].

Современная проблема профессиональной подготовки студентов является предметом пристального внимания и изучения. Так, Ю.В. Кудинов в своих трудах определил сущность профессиональной подготовки, как целенаправленного и организованного педагогического процесса по овладению студентами системой профессиональных знаний, навыков и умений, формированию личных качеств, качественном решению профессиональных задач в соответствии с должностным назначением [73]. Л.М. Резник отметила, что важным условием совершенствования процесса профессиональной подготовки студентов является специально организованное, научно обоснованное педагогическое управление учебной деятельностью, что предполагает поэтапную реализацию ее соответствующих целей, функций и структуры [127].

По мнению А.Н. Сакаевой, повышенные требования к профессиональной подготовке специалистов способствуют созданию и использованию в образовательном процессе высшей школы все более совершенных средств и технологий обучения, инициируют поиск новых форм реализации образовательных услуг [147].

Группа исследователей [18] интерпретирует понятие профессиональной подготовки студентов через усвоение профессиональных знаний, умений и навыков. При этом, по их мнению, освоение определенных знаний, умений и навыков есть необходимое, но не единственное условие качественной подготовки студентов. Главным в определении качества является ориентация профессиональной подготовки на конкретную практическую деятельность.

А.А. Орлов рассматривает профессиональную подготовку как процесс и результат освоения студентами системы профессиональных знаний, осознания личностного смысла этих знаний, формирование основных общепедагогических умений, развитие важнейших профессионально-личностных качеств [102].

Г.Н. Сериков в своей работе отмечает, что профессиональная подготовка характеризуется личностным опытом, который приобретается в процессе совместной деятельности преподавателей и студентов и становится инструментом профессиональной деятельности [153].

Таким образом, обобщая рассмотренные выше положения, можно сделать вывод о том, что при всех различиях во взглядах ученых прослеживается и некоторое сходство. Все авторы заявляют об усвоении фундаментальных знаний, умений и навыков как основном показателе профессиональной подготовки студентов. Кроме того, многие исследователи отмечают, что профессиональная подготовка предполагает формирование профессиональной готовности к практическому применению полученных знаний и развитию личностных свойств. И как отмечают И.А. Ларионова и В.А. Дегтерев, в качестве центрального понятия профессиональной подготовки выступают профессиональные компетенции [86].

Современное представление компетентностного подхода раскрывается в трудах множества авторов. Объясняется это тем, что применительно к системе высшего образования в современной российской педагогической науке, а также в практике отечественного образования компетентностный подход утвердился в качестве основного. Активная разработка фундаментальных основ этого подхода отражена в работах В.И. Байденко [13], В.А. Болотова [22], Б.Д. Эльконина [203], И.А. Зимней [63], М.П. Лапчика [83] и других российских учёных. Вопросам определения различных компонентов профессиональной компетентности посвящены работы Алдашевой А.А. [5], Е.М. Барановой [14], А.В. Ереминой, И.В. Зороастровой, Е.О. Сучковой [51], В.Е. Андреева [6], Л.В. Елагиной [49], С.А. Скворцовой [155] и др. Особенности процесса формирования профессиональных компетенций в системе высшего образования раскрываются в трудах В.А. Адольфа [3], В.Н. Пелевина [116], Е.П. Нехожиной [92, 93], Е.С. Кулюкиной [78], Е.А. Синкиной [154], Л.Ю. Лазаревой и С.Н. Ларина [80], И.Н. Одарича [99] и других авторов.

Так, А.Г. Бермус, рассматривая проблемы и перспективы реализации компетентностного подхода в образовании, описывает компетентностный подход как идею общего и личностного развития, сформулированные в контексте психолого-педагогических концепций развивающей и личностно-ориентированного образования, где содержание образования определяется четырехкомпонентной моделью: знания, умения, опыт творческой деятельности и опыт ценностного отношения [19].

В.И. Байденко под компетентностным подходом понимает метод моделирования результатов образования как норм его качества, что означает отражение в системном и целостном виде образа результата образования и формирование результатов как признаков готовности студента продемонстрировать соответствующие компетенции [12].

Проведенный анализ трудов В.А. Болотова, В.В. Серикова [22], И.А. Зимней [63], М.П. Лапчика [83] в области усовершенствования образования путем использования компетентностного подхода позволяет сделать вывод о том, что основной идеей данного подхода является образование и организация учебного процесса, направленные на формирование готовности к эффективному решению профессиональных, социальных, личностных проблем в современных условиях.

К составляющим компетентностного подхода относят понятия «компетенции» и «компетентность». *Компетенция* – это совокупность взаимосвязанных качеств личности, задаваемых по определенному кругу предметов и процессов [170]. *Компетентность* – это владение человеком соответствующей компетенцией, которая включает его личностное отношение к ней и предмету деятельности» [170].

В теории компетентностного подхода используются также термины, актуальные в аспекте проводимого нами исследования, как «образовательная компетенция» и «профессиональная компетентность». *Образовательная компетенция* – это совокупность смысловых ориентаций, знаний, умений, навыков и опыта деятельности студента по отношению к определенному

кругу объектов реальной действительности, необходимых для осуществления личностной и социально-значимой продуктивной деятельности [69]. *Профессиональная компетентность* – это взаимосвязь опыта, знаний, умений и отношения личности как самостоятельных процессов, направленных на достижение результата профессиональной деятельности [69].

Компетентностный подход заложен в основу действующих в настоящее время Федеральных государственных образовательных стандартов, главной функцией которого является целостность приобретаемых умений и навыков в профессиональной деятельности. Понятия «профессиональный модуль», «междисциплинарный курс», «уровень освоения», «формулировка компетенции», «уровень деятельности» – рассматриваются как базовые термины ФГОС третьего поколения. Компетентностная модель образовательного процесса увеличивает системность в его организации, что акцентирует внимание на интегративных характеристиках подготовки выпускника как целостной личности. Компетенции рассматриваются как своеобразный каркас для всего многообразия результатов обучения, а компетентностный подход выступает как метод моделирования результатов образования и представления их как нормы качества образования [30]. При этом, согласно мнению И.Е. Востроknутова и Д.В. Шагбазян, формулировка многих компетенций носит общий рекомендательный характер, наполнение их содержанием должно основываться на требованиях работодателей [34, с. 126].

На основании анализа научно-педагогического знания компетентностного подхода, требований Федеральных государственных стандартов высшего образования в нашем исследовании будем говорить о формировании *компетенции бакалавра прикладной информатики в области объектно-ориентированного программирования*. Данная компетенция относится к числу образовательных, и она конкретизирует компетенции образовательного стандарта, определяющие способности разрабатывать,

внедрять и адаптировать прикладное программное обеспечение в структуре профессиональной компетентности бакалавра прикладной информатики.

Профессиональные компетенции бакалавра, по мнению Г.В. Прозоровой – это компетенции, определяющие готовность к деятельности в профессиональной сфере и включающие следующие компоненты:

– *операционально-функциональный* – обобщенные способы действий по выполнению профессиональных задач.

– *знаниево-ориентировочный* – практико-ориентированные знания, необходимые для выполнения профессиональных задач;

– *мотивационно-ценностный* – мотивы и ценности, определяющие отношение и побуждающие к освоению и выполнению профессиональных задач;

– *рефлексивно-целевой* – самооценка готовности к выполнению профессиональных задач, определяющая ответственность за результаты их выполнения, и конкретные цели дальнейшего освоения профессиональной деятельности [125, с.32-33].

Несколько иная трактовка уровневой структуры компетентности представлена в исследованиях И.А. Зимней. Указывая на расширение, конкретизацию и переосмысление понятия «компетентность» в современном образовании, автором выделяется следующий компонентный состав ее содержания:

а) знание содержания компетентности (когнитивный аспект);

б) умение, опыт проявления компетентности в разнообразных стандартных и нестандартных ситуациях (поведенческий аспект);

в) ценностное отношение к содержанию, процессу и результату актуализации компетентности (ценностно-смысловой аспект);

г) эмоционально-волевая регуляция процесса и результата проявления компетентности (регулятивный аспект);

д) готовность к актуализации проявления компетентности в разнообразных ситуациях решения социальных и профессиональных задач (мотивационный аспект) [62, с.12].

Проведенный таким образом анализ научно-педагогических знаний о компетентностном подходе, концепций и гипотез о возможности их формирования [13, 22, 58, 62, 63, 125, 203], позволил выделить основные компоненты профессиональной компетентности, позволяющие адекватно описать и структуру компетенции бакалавра прикладной информатики в области объектно-ориентированного программирования: *мотивационно-ценностный*, *организационно-содержательный*, *когнитивно-операционный* и *личностно-рефлексивный* компоненты. Перечисленные структурные компоненты образуют единое целое и находятся в тесной взаимосвязи. Опишем их ниже более подробно.

Мотивационно-ценностный компонент компетенции бакалавра прикладной информатики в области объектно-ориентированного программирования представляет собой совокупность интереса к объектно-ориентированному программированию, осознания мотивов и целей профессиональной разработки компьютерных программ. К основным характеристикам компонента относят осознание студентами наличия у себя знаний по программированию и способности к их использованию на практике. Функцией компонента является то, что он направлен на создание условий активизации познавательной деятельности студентов и развития положительной мотивации к обучению.

Организационно-содержательный компонент компетенции бакалавра прикладной информатики в области объектно-ориентированного программирования – это набор теоретических знаний и познавательной активности, необходимых *для изучения* объектно-ориентированного программирования. Основными характеристиками данного компонента являются полнота, глубина, владение базовыми концепциями объектно-ориентированного подхода, которые позволяют дальше изучать конкретные

языки программирования. Организационно-содержательный компонент выполняет *образовательную* функцию, которая заключается в создании условий усвоения знаний в области объектно-ориентированного программирования и его разделов, методов и технологий решения задач.

Когнитивно-операционный компонент компетенции в области объектно-ориентированного программирования указывает на степень освоения теории объектно-ориентированного программирования и способности использования этих знаний *в профессиональной деятельности*. Уровень сформированности компонента определяется такими характеристиками, как системность знаний бакалавра прикладной информатики в области объектно-ориентированного программирования. Данный компонент имеет такие характеристики, как системность, оперативность, мобильность знаний, умение усваивать знания в области программирования, использование этих знаний при решении профессиональных задач. Основной функцией данного компонента является реализация условий *профессионального решения* задач в области объектно-ориентированного программирования.

Личностно-рефлексивный компонент компетенции в области объектно-ориентированного программирования заключается в наличии у бакалавра прикладной информатики собственного стиля, способности оценивать собственную деятельность и ее результаты, совершенствовать знания в области современного программирования, осознавать собственную значимость в коллективе, а также самореализовываться в профессиональной деятельности, используя современные информационные технологии. Данный компонент характеризуется глубиной и последовательностью анализа имеющихся опыта и знаний, способностью к принятию решений по корректировке своей деятельности с целью исключения ошибок и выработке стратегий совершенствования своего мастерства. Функция компонента – в создании условий профессионального роста, формирования собственного стиля, совершенствования знаний, осознания своей роли и значимости и в

целом – активизации внутренних механизмов развития профессиональной компетентности в части разработки компьютерных программ.

Компонентная структура компетенции бакалавра прикладной информатики в области объектно-ориентированного программирования в виде общей модели представлена на рисунке 2. Компетенция состоит из четырех компонент: мотивационно-ценностного, организационно-содержательного, когнитивно-операционного и личностно-рефлексивного, но знания и умения бакалавра прикладной информатики в области объектно-ориентированного программирования непосредственно формируются лишь в рамках организационно-содержательного и когнитивно-общенационального компонентов, в связи с чем весь процесс подготовки можно представить в виде двух частей:

1) подготовка в области методологии, базовых концепций и идей объектно-ориентированного программирования (мотивационно-ценностный, организационно-содержательный, личностно-рефлексивный компоненты);

2) подготовка в области применения объектно-ориентированного программирования с использованием конкретного языка программирования для решения профессиональных задач (мотивационно-ценностный, когнитивно-операционный, личностно-рефлексивный компоненты).

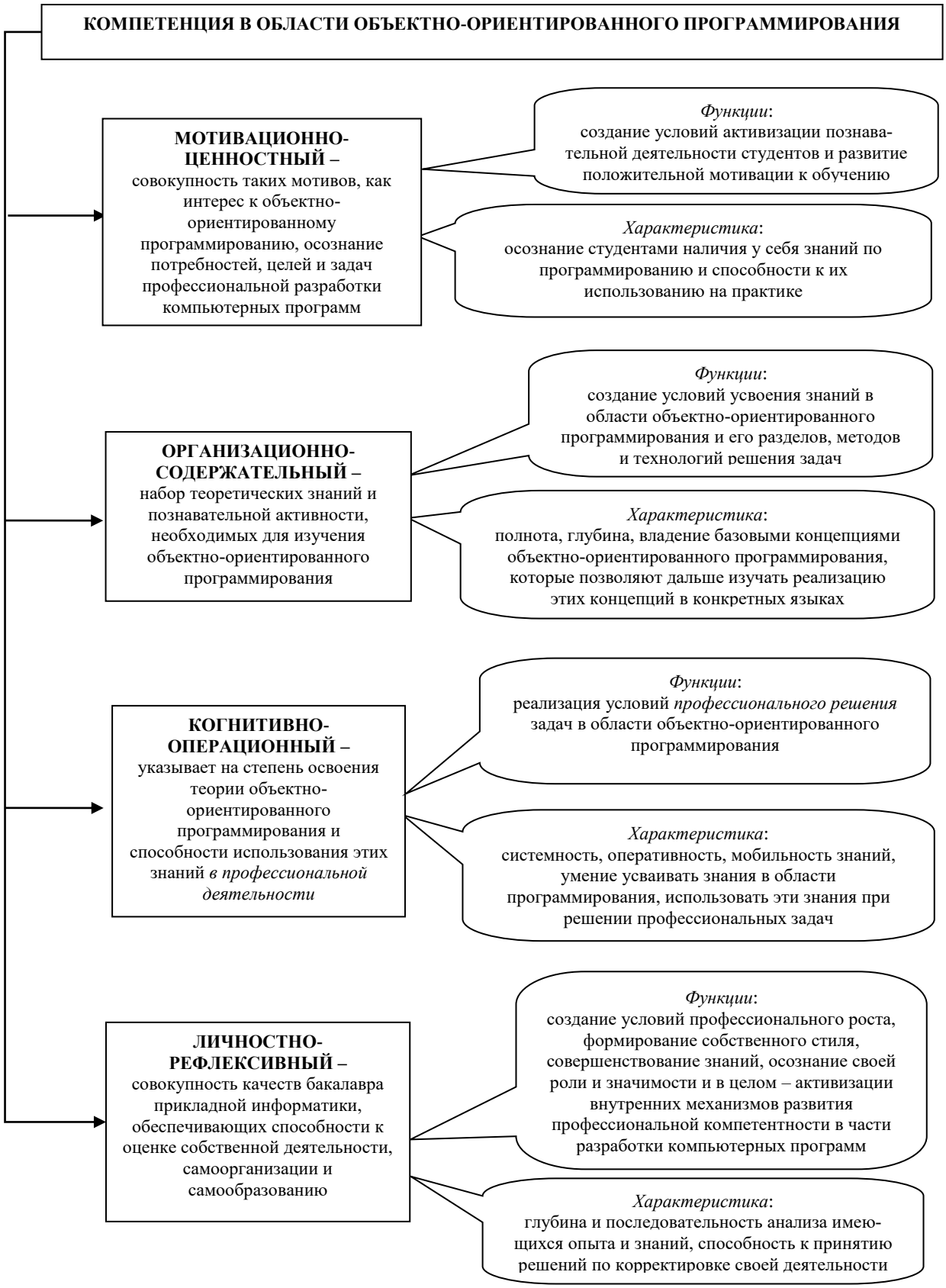


Рисунок 2. Основные компоненты профессиональной компетенции в области объектно-ориентированного программирования

Необходимость реализации двух частей подготовки бакалавра прикладной информатики в области объектно-ориентированного программирования обосновывает выбор двух взаимосвязанных дисциплин, в рамках которых осуществляется такая подготовка, – дисциплину введения в объектно-ориентированного программирования («Введение в объектно-ориентированное программирование», «Программирование для начинающих», «Визуальное программирование» или др.), а также дисциплину, где изучаются технологии объектно-ориентированного программирования с использованием конкретного языка («Информатика и программирование», «Языки и методы программирования» или др.) (рис. 3).

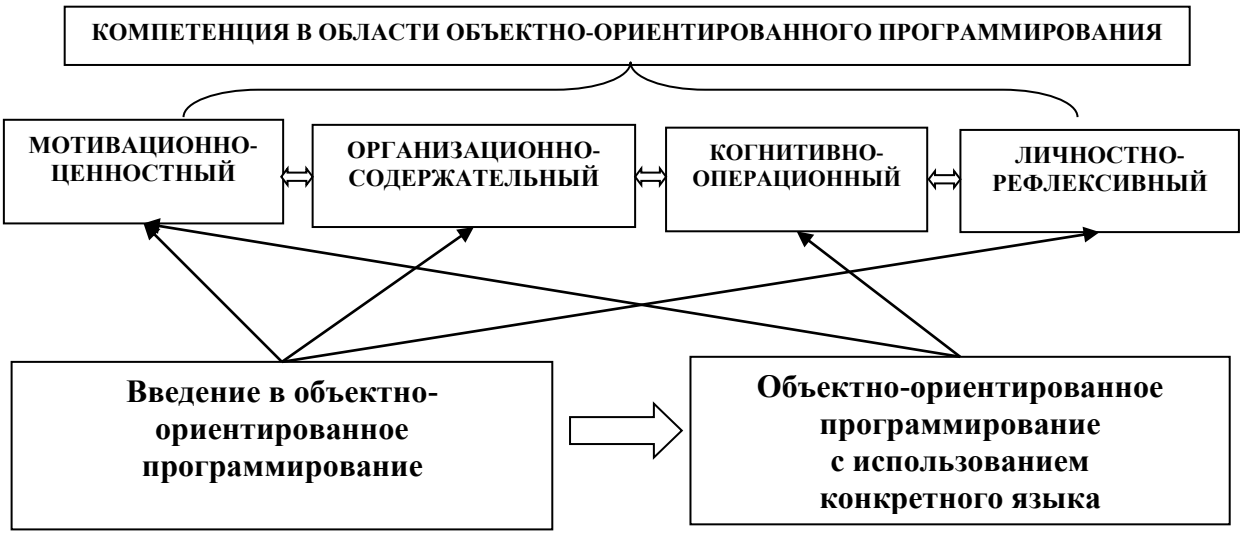


Рисунок 3. Формирование компонент компетенции в области объектно-ориентированного программирования в процессе подготовки бакалавра прикладной информатики

Опишем подробнее содержание подготовки бакалавров прикладной информатики по каждому из выделенных компонентов. В данном описании также будем обращать внимание и на то, как эти компоненты могут соотноситься с двумя частями подготовки – в рамках каждой из двух выделенных дисциплин.

Мотивационно-ценностный компонент

Мотивационно-ценностный компонент компетенции бакалавра прикладной информатики в области объектно-ориентированного программирования, как указано нами ранее, представляет собой совокупность мотивов к профессиональной разработке компьютерных программ. Данный аспект подготовки студента раскрывается в аспекте формирования психологической готовности студента к изучению программирования и созданию компьютерных программ.

Е. И. Машбиц, анализируя психолого-педагогические проблемы компьютеризации обучения, выделяет два подхода к пониманию сути психологической готовности студента: функциональный и личностный. Функциональный подход предполагает исследование готовности к деятельности как определенного состояния психологической функции, при котором достигается высокий уровень в определенной деятельности. Личностный подход предполагает результат подготовки к определенной деятельности. То есть, психологическая готовность – это интегральное образование личности, включающее ряд компонентов (мотивационный, когнитивный, эмоционально-волевой) и совокупность знаний, умений, навыков и личностных качеств, адекватных требованиям содержания и условиям деятельности. Существует глубокая взаимосвязь функционального состояния готовности и готовности как устойчивой характеристики личности. Эта готовность, как указывает Е. И. Машбиц, включает в себя мотивы и цели деятельности человека, работающего с ЭВМ, ее функциональное состояние и работоспособность [89, с.40].

Основой психологической готовности студента к использованию информационных технологий являются знания, умения, навыки и мотивы его деятельности. Поэтому формирование психологической готовности у студентов компьютерных специальностей – это овладение ими техническими процедурами и программами решения задач. Важнейшей, по мнению

Е.И. Машбица, остается компьютерная грамотность студентов, поскольку благодаря ней, студенты могут выдвигать гипотезы, ставить перед собой задачи, создавать проблемные ситуации, которые можно успешно решать с помощью компьютера.

В профессиональной деятельности бакалавров прикладной информатики особенно важным моментом является то, что они используют информационные технологии, как «орудие труда», что влечет за собой появление новых форм мышления, творческой деятельности. Как определяет М.В. Коннова – «происходит превращение умственной деятельности человека к появлению новых перспектив, при которых компьютер как оружие умственной деятельности изменяет эту деятельность» [70, с.180].

Таким образом, в структуре мотивационно-ценностного компонента компетенции бакалавра прикладной информатики в области объектно-ориентированного программирования можно выделить два типа готовности:

– готовность к освоению базовых концепций, теории, технологий и конкретных средств объектно-ориентированного программирования;

– готовность к использованию знаний, умений, навыков и личностных качеств в области объектно-ориентированного программирования для решения профессиональных задач по разработке компьютерных программ.

Формирование готовности первого типа в полной мере можно отнести к дисциплине введения в объектно-ориентированного программирования («Программирование для начинающих» или др.), а готовности второго типа – к дисциплине, ориентированной на изучение технологий объектно-ориентированного программирования с использованием конкретного языка («Информатика и программирование» или др.).

Организационно-содержательный компонент

Организационно-содержательный компонент компетенции бакалавра прикладной информатики в области объектно-ориентированного

программирования, согласно представленной нами модели, – это набор теоретических знаний и познавательной активности, необходимых для осуществления процесса обучения объектно-ориентированному программированию. Формирование этого компонента нами относится к дисциплине введения в объектно-ориентированное программирование («Программирование для начинающих» или др.)

Проблемы и вопросы формирования профессиональных качеств у бакалавров прикладной информатики являются открытыми и рассматривались как психологами, так и педагогами. В частности, А.П. Ершов [55], Ф. Брукс [27], Е. Дейкстра [48], С. Макконнелл [88], М.Л. Смульсон [157], Б. Шнейдерман [202] отмечали, что программисты имеют свои психологические черты и качества. Авторами были определены способности и особенности мышления, которые должны быть характерными для программистов.

Так, А. П. Ершов отмечал, что «программист должен обладать способностью первоклассного математика к абстракции и логическому мышлению в сочетании с эдисоновским талантом сооружать все, что угодно, из нуля и единиц» [55, с. 164]. Обучение программированию рассматривалось ученым в аспекте обучению информатике вообще. По мнению А. П. Ершова, для эффективного использования возможностей вычислительной техники человеку необходимо владеть определённым стилем мышления, определёнными навыками умственных действий, которые в наиболее явном виде присущи профессиональным программистам. К наиболее существенным из этих навыков учёный относил:

– умение планировать структуру действий, необходимых для достижения заданной цели при помощи фиксированного набора средств;

– умение строить информационные структуры для описания объектов и систем; умение организовать поиск информации, необходимой для решения поставленной задачи;

– умение правильно, чётко и однозначно сформулировать мысль в понятной собеседнику форме и правильно понять текстовое сообщение;

– привычка своевременно обращаться к ЭВМ при решении задачи из любой области;

– наличие минимальных технических навыков взаимодействия с ЭВМ [52].

Э. Дейкстра утверждал, что программисту должны быть присущи качества, связанные непосредственно с созданием программного продукта:

– способность определить архитектуру программы (разбивать сложную задачу на элементарные составляющие и задать варианты их комбинирования);

– умение видеть задачу одновременно на различных уровнях детализации (свободно переходить от описанной задачи в общих понятиях к сути низшего уровня, что стоит за этими понятиями);

– умение представлять процесс, который проектируется в динамике, поскольку данные, обрабатываемые в некоторый момент времени, могут иметь одни значения и взаимосвязи, а в следующий момент некоторые из них могут изменяться;

– умение видеть дальше программы, что разрабатывается в данный момент (учитывать широкое окружение данной части задачи и возможность ее включения в часть общей системы);

– умение обобщать типичные ситуации (находить в программе идейно однородные участки);

– умение применять и комбинировать хорошо известные приемы программирования и типовые алгоритмы [48].

Б. Шнейдерман, изучая человеческие факторы вычислительных и информационных системах, выделил следующие качества, которые должны быть присущи программисту:

– способность понимать программы;

– способность налаживать программы, находить ошибки в программе (сложность такой работы обусловлена психологическими факторами, в частности, усиленным беспокойством и нежеланием допускать ошибки);

– способность модифицировать программы (успешность в решении задачи модификации можно считать одним из критериев понимания программ);

– умение запоминать и воспроизводить текст программ [202].

Согласно исследованиям М.Л. Смульсона, основные качества, которыми должен обладать программист, являются:

– гибкость и стратегичность мышления;

– творческие особенности мышления;

– внимательность, проявляющаяся в умении предполагать меньше ошибок;

– логический характер мышления [157].

Конкретизируя содержание подготовки в аспекте изучения базовых концепций и идей объектно-ориентированного программирования следует указать, что такая подготовка должна быть нацелена на формирование определенных знаний, умений, навыков и опыта деятельности бакалавров приданной информатики.

Проектируя эти знания, умения, навыки и опыт деятельности мы исходим из того, что обучение основам объектно-ориентированного программирования должно предусматривать изучение не конкретного языка программирования, а интеграцию абстракции и элементов проектирования в курс. Объектно-ориентированное программирование – это построение и реализация абстракций, представление модели задачи в терминах объектов и их взаимодействия (структурирование программы и ее проектирование с помощью объектов и соответствующих классов), что не исключает предварительного анализа ее в терминах алгоритмического подхода (метод как составляющая объекта описывается как алгоритм). Алгоритмические

детали и структуры данных должны базироваться на моделировании и проектировании.

Подтверждение высказанного положения мы находим у Б.С. Садулаевой, которая в своем исследовании отмечает, что обучение объектно-ориентированному программированию способствует целостному представлению о методологии разработки программных средств, а современному студенту требуются знания не только пользовательских программных средств, но и идеологии их проектирования и разработки [133].

Так, опираясь на данную позицию, а также исходя из высказанного положения о том, что формирование рассматриваемого нами компонента должно относиться к дисциплине введения в объектно-ориентированного программирования, мы можем указать, что в результате освоения организационно-содержательного компонента компетенции в области объектно-ориентированного программирования бакалавры прикладной информатики должны:

1) знать содержание базовых понятий в области программирования;
2) понимать основные принципы объектно-ориентированного программирования, владеть терминологией указанной парадигмы;

2) знать операторы и управляющие конструкции Alice и Scratch, средства объектно-ориентированного программирования, используемые в данных средах;

4) уметь строить алгоритм решения задачи соответствующего уровня сложности;

5) уметь осуществлять разработку, модификацию, тестирование и отладку программ средствами визуальных учебных сред Alice и Scratch;

8) владеть опытом использования визуальных учебных сред Alice и Scratch на этапах разработки компьютерных программ.

Когнитивно-операционный компонент

Когнитивно-операционный компонент компетенции бакалавра прикладной информатики в области объектно-ориентированного программирования, как нами представлено ранее, указывает на степень освоения теории объектно-ориентированного программирования и способности использования этих знаний в профессиональной деятельности. Формирование этого компонента относится к дисциплине технологий объектно-ориентированного программирования с использованием конкретного языка («Информатика и программирование» или др.).

И.А. Барков в своей работе отмечает, что в процессе обучения объектно-ориентированному программированию бакалавры прикладной информатики должны знать базовые алгоритмические структуры, программный код, понимать понятие формализации задачи, назначение компьютерной программы и объяснять этапы ее выполнения на компьютере, уметь определять свойства объектов и значение этих свойств, осуществлять классификацию и распознавать объекты, осуществлять формализованное описание объекта, реализовать выделение подзадач в описании задачи, понимать и применять на практике принципы объектно-ориентированного программирования [15].

В стандартах Computing Curricula 2016 [209] и Software Engineering 2017 [221], представленных нами в п. 1.1, предлагается следующий набор требований в виде списка разделов и тем, которые должен охватить курс по основам программирования:

– *основные конструкции программирования* – основы синтаксиса и семантики языков высокого уровня; переменные, типы, выражения и операторы присваивания; основы ввода-вывода; операторы ветвления и цикла; функции и передачи параметров; структурная декомпозиция;

– *алгоритмы и решение задач* – стратегии решения задач; роль алгоритмов в процессе решения задач; стратегии отладки программ; концепция и свойства алгоритмов;

– *фундаментальные структуры данных* – стандартные типы; массивы; записи; строки и обработки строк; представление данных в памяти компьютера; статическое, автоматическое и динамическое выделение памяти; управление памятью во время выполнения программы; ссылки и указатели; связанные списки; методы реализации стеков, очередей и хэш-таблиц; методы реализации графов и деревьев; стратегии выбора соответствующей структуры данных;

– *основы анализа алгоритмов* – асимптотический анализ поведения алгоритмов в среднем и крайних случаях; нотация О-большого, О-малого, Ω (омега), θ (тетта); стандартные классы сложности; эмпирические измерения эффективности; компромисс между временем и объемом памяти в алгоритмах; использование рекуррентных отношений для анализа рекурсивных алгоритмов;

– *фундаментальные вычислительные алгоритмы* – простые многочисленные алгоритмы; алгоритмы последовательного и двоичного поиска; алгоритмы сортировки с квадратичной сложностью и со сложностью O (n • log (n)); хэширование; деревья двоичного поиска; представление графов; алгоритмы поиска кратчайшего пути; транзитивное замыкание; минимальное остовное дерево; топологическая сортировка);

– *алгоритмические стратегии* – алгоритмы полного перебора; «жадные алгоритмы»; алгоритмы «разделяй и властвуй»; перебор с возвратом; метод ветвей и границ; эвристики; сопоставления с образцом и алгоритмы обработки текста; алгоритмы численной аппроксимации;

– *проектирование программного обеспечения* – фундаментальные концепции и принципы проектирования; объектно-ориентированный анализ и проектирование; проектирование с целью повторного использования;

– *спецификации и требования к программному обеспечению* – важность спецификации в процессе создания программ;

– *проверка соответствия программного обеспечения* – основы тестирования; генерация тестовых примеров.

Таким образом, на основании анализа международных стандартов, а также с учетом выбора языка С++ как подлежащего изучению в качестве первого профессионального языка, в результате освоения когнитивно-операционного компонента компетенции в области объектно-ориентированного программирования бакалавры прикладной информатики должны:

1) знать понятие алгоритма и его свойств, различные формы записи алгоритма;

2) знать операторы и управляющие инструкции языка программирования С++;

3) знать средства объектно-ориентированного программирования с использованием языка программирования С ++, правила определения классов, их назначение и использование;

4) знать приемы объектно-ориентированного анализа и проектирования компьютерных программ;

5) уметь разрабатывать алгоритмы решения задачи соответствующего уровня сложности;

6) уметь создавать консольные приложения, реализующие разработанный алгоритм – программу на языке программирования С ++;

7) уметь использовать отладчик для поиска логических ошибок;

8) уметь работать с инструментальной средой программирования Microsoft Visual Studio 2012/15.

9) владеть навыками объектно-ориентированного программирования с использованием языка программирования С++.

Личностно-рефлексивный компонент

Личностно-рефлексивный компонент компетенции в области объектно-ориентированного программирования согласно предложенной нами модели, заключается в наличии у бакалавра прикладной информатики собственного стиля, способности оценивать собственную деятельность и ее результаты, совершенствовать знания в области современного программирования, осознавать собственную значимость в коллективе, а также самореализовываться в профессиональной деятельности используя современные информационные технологии.

Данные аспекты профессиональной подготовки бакалавров прикладной информатики раскрываются в трудах, где анализируются особенности формирования профессионализма и профессионального мастерства программиста, особых личностных качеств, присущих профессиональным программистам.

Так, И.О. Одинцов рассматривает профессионализм программиста как личностную характеристику человека, который:

– овладел нормами профессиональной деятельности и общения, а также осуществляет их на высоком уровне, достигая профессионального мастерства в области программирования;

– следует профессиональной ценностной ориентации, соблюдая профессиональную этику;

– развивает себя средствами профессии;

– стремится сделать творческий вклад в профессию, обогатив свой опыт [100].

Согласно Э. Дейкстре, программисту также должны быть присущи определенные психологические и общечеловеческие черты:

– наличие комплексного мышления – умение заранее определять этапы, которые нужно пройти, чтобы решить тот или иной вопрос;

– культура собственного труда – умение обеспечить себя необходимым инструментарием для работы;

– способность анализировать собственные ошибки

– умение работать в коллективе;

– умение работать с пользователем;

– соблюдение правил общечеловеческой этики;

– способность четко видеть действия;

– способность выявлять все случаи, где можно применить теорию, самостоятельно решиться на ее применение или обратиться за советом к специалисту программисту;

– способность при неудаче преодолеть самолюбие и найти другой подход к решению задачи [48].

Б. Шнейдерман при анализе человеческих факторов в вычислительных и информационных системах выделяет следующие психологические и общечеловеческие черты, присущие программистам:

– настойчивость / пассивность;

– интровертность / экстровертность;

– внутренняя / внешняя управляемость;

– высокая / низкая мотивация;

– высокая / низкая терпимость;

– умение быть точным;

– скромность;

– способность переносить стресс [202].

К психологическим и общечеловеческим чертам, присущим программисту, М.Л. Смульсон относит:

– работоспособность и исполнительность в работе;

– оперативность мышления;

– умение принимать решения в условиях ограниченного времени;

– умение создавать себе рабочее место, которое способствует повышению производительности труда [157].

Таким образом, личностно-рефлексивный компонент компетенции в области объектно-ориентированного программирования подразумевает формирование навыков самосовершенствования и самореализации бакалавров прикладной информатики. Это возможно путем вырабатывания определенного набора профессиональных знаний, умений и навыков в области объектно-ориентированного программирования и развития таких качеств, как работа в команде, лидерские качества, индивидуальность, способность к рефлексии, способность к самостоятельному обучению и освоению новых технологий в течение жизни, самообразование, планирование деятельности, логическое, алгоритмическое и объектно-ориентированное мышление, целеустремленность, настойчивость, умение самостоятельно принять решение, быстро адаптироваться к новым задачам.

Формирование таких качеств должно осуществляться в рамках всей системы подготовки бакалавров прикладной информатики в области объектно-ориентированного программирования. Логика развертывания данной линии подготовки будет зависеть не столько от содержания, сколько от методов, форм и средств учебной деятельности, применяемых на дисциплинах введения в объектно-ориентированное программирование («Программирование для начинающих» или др.), а также изучения технологий объектно-ориентированного программирования с использованием конкретного языка («Информатика и программирование» или др.).

Подводя итог описания компетенции бакалавров прикладной информатики в области объектно-ориентированного программировании и ее компонентов, мы можем заключить, что эта компетенция представляет собой совокупность личностных качеств, знаний и умений в области объектно-ориентированного программирования, обеспечивающих способность к разработке программного обеспечения с использованием современных языков программирования и соответствующих инструментальных средств,

готовность к освоению профессии программиста, осознаваемой как личностно значимой.

Данная компетенция состоит из четырех компонент: мотивационно-ценностного, организационно-содержательного, когнитивно-операционного и личностно-рефлексивного. Каждый компонент характеризуется своим содержанием, функцией и характеристикой. Содержание описывает знания, умения и личностные установки, составляющие в своей совокупности основу и всей компетенции бакалавра прикладной информатики в области объектно-ориентированного программирования. Характеристики компонентов задают индикаторы достижения необходимого уровня образования в процессе подготовки бакалавров прикладной информатики. Функции каждого компонента раскрывают их роль в организации процесса освоения объектно-ориентированного программирования, а также дальнейшей профессиональной деятельности по разработке компьютерных программ.

Таким образом, нами разработана и подробно описана компонентная модель компетенции бакалавра прикладной информатики в области объектно-ориентированного программирования. Эта модель послужит основой разработки методики обучения основам объектно-ориентированного программирования бакалавров прикладной информатики, что будет представлено во второй главе. Однако, чтобы приступить к этой работе, необходимо определить также уровни сформированности рассматриваемой нами компетенции. Эти уровни возможно определить через описание достижения тех или иных показателей по критериям сформированности компонентов компетенции бакалавров прикладной информатики в области объектно-ориентированного программирования.

На основе анализа психолого-педагогической литературы по проблемам изучения компетентностного подхода и ключевых его составляющих, а также системы оценки профессиональной компетентности и компонентный состав профессиональной компетентности, нами определены три уровня (*достаточный*, *средний*, *высокий*) сформированности

компетентности в области объектно-ориентированного программирования при подготовке бакалавров прикладной информатики. Показатели уровней базируются на выделенных компонентах компетентности в области объектно-ориентированного программирования, а также ФГОС ВО по направлению подготовки «Прикладная информатика».

Достаточный уровень сформированности компетенции бакалавра прикладной информатики в области объектно-ориентированного программирования характеризуется умением искать необходимые студенту данные в различных источниках и работать с ними; решать определенные практические задачи; использовать новые информационные технологии; работать с определенной средой программирования.

Для студентов с достаточным уровнем сформированности компетенции бакалавра прикладной информатики в области объектно-ориентированного программирования характерен низкий уровень теоретических знаний в области объектно-ориентированного программирования – они только различают понятия, оперируют представлениями о них, могут назвать определения, классификации, виды и др., но путаются в терминах. Знания поверхностные, наблюдаются пробелы в содержании программы курса. Цели и задачи ставятся формально, безотносительно к профессиональным функциям. Студенты раскрывают основные понятия объектно-ориентированного программирования на уровне теоретических определений. У них недостаточно сформирована положительная мотивация к изучению объектно-ориентированного программирования. Имеются низкие представления о роли объектно-ориентированного программирования в профессиональной деятельности, осознанность применения знаний в практической жизни выражена недостаточно. Студенты не имеют представления о собственных возможностях, способности к осуществлению профессиональной деятельности в качестве бакалавра прикладной информатики. Студенты знакомы с основными методами и формами в области объектно-

ориентированного программирования на практике и обладают отдельными умениями использования принципов объектно-ориентированного программирования при решении профессиональных задач, однако испытывают серьезные трудности в применении их на практике при разработке программ. Несложными и небольшими по объему разработками программ могут заниматься только при непосредственном участии и с помощью преподавателя.

Средний уровень сформированности компетенции бакалавра прикладной информатики в области объектно-ориентированного программирования характеризуется умением работать с данными; предвидеть возможные сложности при разработке программы; самостоятельно проектировать сложные информационные процессы и умело использовать свои знания, умения и навыки в новых ситуациях; владеть новыми информационными технологиями и использовать их в объектно-ориентированном программировании; разрабатывать программы, используя специализированное программное обеспечение; осуществлять самооценку; владеть одним иностранным языком.

Студенты со средним уровнем сформированности компетенции бакалавра прикладной информатики в области объектно-ориентированного программирования характеризуются положительным отношением к формированию профессиональной компетенции, осознают значимость профессиональных знаний и умений для будущей профессиональной деятельности, а также для повседневной жизни. Студенты имеют сложившиеся социально значимые качества личности. У бакалавров прикладной информатики развиты организационные умения, самостоятельность, творческая активность, формируется потребность в исследовательской и проектной деятельности. В процессе обучения раскрываются профессионально значимые качества личности, проявляется самостоятельность, инициативность и мобильность.

Студенты с *высоким* уровнем сформированности компетенции в области объектно-ориентированного программирования характеризуются умением создавать новые программы и проекты; использовать поисковые системы для поиска необходимых данных; создавать собственные программные средства, используя специализированное программное обеспечение; уметь прогнозировать возможные осложнения во время нахождения решения поставленной задачи; заниматься самообразованием и самосовершенствованием; отвечать за последствия своей деятельности; использовать личный опыт и оказывать помощь другим.

Студенты с высоким уровнем сформированности компетенции бакалавра прикладной информатики в области объектно-ориентированного программирования инициативные, коммуникабельные, проявляют высокую осведомленность в области объектно-ориентированного программирования. Проявляют понимание методологии объектно-ориентированного программирования, особенности ее применения в будущей профессиональной деятельности, активно занимаются самообразованием и исследовательской деятельностью. Студенты показывают высокий уровень проявления профессионально значимых личностных качеств.

Оценка уровня сформированности компетенции бакалавра прикладной информатики в области объектно-ориентированного программирования по указанным критериям может производиться на основе комплексных прикладных задач, для которых обязательным является применение современных информационных технологий [134].

Таким образом, каждый из рассмотренных уровней содержит иерархию мотивов, целей, их перспективу и зависит от самосознания и уровня самооценки. Учитывая классическое описание критериальной базы, мы в данном диссертационном исследовании выделили три уровня сформированности компетенции в области объектно-ориентированного программирования: достаточный, средний, высокий. Результаты анализа уровней сформированности компетенции бакалавра прикладной

информатики в области объектно-ориентированного программирования по каждому компоненту компетенции представлены в таблице 5.

Таблица 5.

Уровни сформированности компетенции
бакалавра прикладной информатики в области объектно-ориентированного программирования по представленным компонентам

Мотивационно-ценностный компонент	Организационно-содержательный компонент	Когнитивно-операционный компонент	Личностно-рефлексивный компонент
достаточный уровень			
Характеризуется перестройкой ценностно-мотивационной сферы студентов; ощущение необходимости изучения объектно-ориентированного программирования для повышения своего профессионального уровня	Понимание принципов работы, возможностей объектно-ориентированной парадигмы; умение использовать средства объектно-ориентированного программирования при решении профессиональных задач	Знание принципов объектно-ориентированного программирования и умение использовать принципы объектно-ориентированного программирования при решении профессиональных задач.	Умения использовать полученные знания для решения задач в области объектно-ориентированного программирования на основе самоанализа и саморегуляции
средний уровень			
Приобретение личных качеств ответственности, организованности, целеустремленности; способность к самовоспитанию; применять на практике полученные знания	Знание технологии программирования объектно-ориентированной парадигмы; некоторые знания и умения конкретных способов их реализации.	Стремление к углубленному программированию в области объектно-ориентированной парадигмы; умение практически использовать полученные знания в условиях современного программирования.	Способность разносторонне подходить к анализу ситуаций при решении объектно-ориентированных задач в зависимости от целей и условий.
высокий уровень			
Четкая мотивационная позиция в необходимости изучения объектно-ориентированного программирования для повышения профессионального	Способность классифицировать задачи по типам и выбирать соответствующее и наиболее подходящее решение для реализации	Знание методологии объектно-ориентированной парадигмы; знание возможностей современных информационных технологий; стремление к	Умение анализировать и оценивать эффективность использования объектно-ориентированное программирование в профессиональной

Мотивационно-ценностный компонент	Организационно-содержательный компонент	Когнитивно-операционный компонент	Личностно-рефлексивный компонент
уровня; приобретения ценностных ориентаций, мотивов адекватных целям и задачам деятельности.	программного кода.	обретению профессиональных компетенций.	деятельности, наличие профессионального опыта и определенной профессиональной позиции.

Итак, представленные уровни сформированности компетенции бакалавра прикладной информатики в области объектно-ориентированного программирования позволяют определить степень овладения методологией объектно-ориентированного программирования и определенный уровень сформированности каждого компонента компетентности. Использованные трехуровневые характеристики (достаточный средний и высокий уровни) позволят провести объективную оценку степени сформированности компетенции бакалавра прикладной информатики в области объектно-ориентированного программирования, а также эффективности соответствующей методики.

ВЫВОДЫ ПО ГЛАВЕ 1

В результате аналитической работы в рамках диссертационного исследования сделаны следующие выводы:

1. Задача подготовки специалистов в области информационных технологий связана с реализацией образовательной области информатики на всех уровнях образования, в частности – в программах подготовки бакалавров прикладной информатики. Одним из приоритетных направлений данной подготовки является обучение программированию, что реализуется на младших курсах и является основой для освоения последующих учебных дисциплин профессионального цикла.

Ведущее место в структуре подготовки бакалавров прикладной информатики отводится программированию в области объектно-ориентированной методологии, поскольку сегодня это современный стандарт в области разработки программных продуктов. Для реализации этого направления необходимо обеспечить достаточно глубокое изучение всех компонентов объектно-ориентированного программирования.

С этой целью целесообразно ввести в учебный план подготовки бакалавров прикладной информатики отдельный курс, посвященный основам объектно-ориентированного программирования. Целью данного курса является представление задачи в терминах объектов и их взаимодействия.

Перечень тем, которые являются ядром обучения основам объектно-ориентированного программирования, можно представить следующим образом:

– объект;

– класс;

– данные и переменные;

– методы (алгоритмическая составляющая);

– конструкторы и метод перезагрузки;

– инкапсуляция и модификаторы видимости;

– взаимодействие между объектами;

– наследование, программная рекурсия.

2. Компетентность бакалавра прикладной информатики в области объектно-ориентированного программирования предполагает высокий уровень обобщенных профессиональных знаний, готовность к разработке компьютерных приложений в процессе решения профессиональных задач. Эта компетенция является составной частью ядра профессиональной компетентности бакалавра прикладной информатики, что позволяет выпускнику вуза быть современным и конкурентоспособным на рынке труда.

Данная компетенция состоит из четырех компонент: мотивационно-ценностного, организационно-содержательного, когнитивно-операционного

и личностно-рефлексивного. Каждый компонент характеризуется своим содержанием, функцией и характеристикой. Содержание описывает знания, умения и личностные установки, составляющие в своей совокупности основу и всей компетенции бакалавра прикладной информатики в области объектно-ориентированного программирования. Функции каждого компонента раскрывают их роль в организации процесса освоения объектно-ориентированного программирования, а также дальнейшей профессиональной деятельности по разработке компьютерных программ. Характеристики компонентов задают индикаторы достижения необходимого уровня образования в процессе подготовки бакалавров прикладной информатики.

Формирование компетенции бакалавра прикладной информатики в области объектно-ориентированного программирования может осуществляться на основе реализации двух взаимосвязанных дисциплин, ориентированных на формирование теоретических знаний и познавательной активности обучающихся, необходимых для изучения самих основ объектно-ориентированного программирования, а конкретного объектно-ориентированного языка программирования и соответствующих инструментальных сред.

Основные результаты первой главы опубликованы в работах [174, 175, 176, 177, 179, 180, 181, 183, 187, 190, 192, 193, 194, 198, 199, 200, 201].

ГЛАВА 2. МЕТОДИЧЕСКИЕ АСПЕКТЫ ПРОЦЕССА ОБУЧЕНИЯ ОСНОВАМ ОБЪЕКТНО-ОРИЕНТИРОВАННОГО ПРОГРАММИРОВАНИЯ БАКАЛАВРОВ ПРИКЛАДНОЙ ИНФОРМАТИКИ

2.1. Компоненты методики обучения основам объектно-ориентированного программирования бакалавров прикладной информатики

На основании анализа целевых и содержательных характеристик обучения в области объектно-ориентированного программирования бакалавров прикладной информатики, учитывая роль объектно-ориентированного программирования в обучении бакалавров прикладной информатики, рассмотрим компоненты методики обучения основам объектно-ориентированного программирования бакалавров прикладной информатики.

Понятие «методика обучения» исследователи А.Б. Кузнецов [75], М.П. Лапчик [85], И.В. Роберт [129] трактуют как способ организации практической и теоретической деятельности участников обучения, обусловленный закономерностями и особенностями содержания учебного предмета. А.М. Пышкало [126], М.В. Швецкий [172], И.Б. Готская [40] методику обучения интерпретируют как построение методической системы обучения. Н.И. Рыжова и А.А. Ляш [130] в своей работе предлагают модель методики обучения, состоящую из пяти элементов – цели, содержание, методы, формы и средства обучения [130].

В данной диссертационной работе мы ссылаемся на исследования, проводимые Н.И. Рыжовой, А.А. Ляш [130] и методику обучения основам объектно-ориентированного программирования бакалавров прикладной информатики представим в виде пяти компонентов: цели, содержание, средства, методы, формы.

Цели обучения

Общей целью обучения бакалавров прикладной информатики основам объектно-ориентированного программирования является формирование целостной *компетенции бакалавра прикладной информатики в области объектно-ориентированного программирования*. Данная цель, согласно выделенным компонентам рассматриваемой нами компетенции, конкретизируется в целях двух взаимосвязанных дисциплин («Программирование для начинающих» и «Информатика и программирование»), а также четырех аспектах – формирования мотивационно-ценностного, организационно-содержательного, когнитивно-операционного и личностно-рефлексивного компонентов (табл. 6, с.73). Опираясь на содержание, функции и характеристики каждого компонента, целями обучения бакалавров прикладной информатики основам объектно-ориентированного программирования является формирование:

– *мотивационно-ценностный компонент*:

1) уверенности в необходимости получения новых знаний в области объектно-ориентированного программирования; понимания значимости этих знаний для овладения специальностью бакалавра прикладной информатики;

2) способности к творчеству с использованием средств информационных технологий при решении профессиональных задач;

3) стремления к самостоятельному получению знаний с использованием информационных технологий, визуальных учебных сред программирования, языка программирования C++;

4) способности к обоснованию, разработке и реализации собственных проектов с использованием визуальных учебных сред программирования, языка программирования C++.

– *организационно-содержательный компонент:*

1) общей системы знаний в области программирования (алгоритмы, виды алгоритмов, программа, компилятор, интерпретатор и т.д.);

2) системы знаний в области основ объектно-ориентированного программирования (проект, объект, свойства, методы и т.д.);

3) знаний о принципах проектирования, конструирования и редактирования отдельных компонентов программы, реализованных в Alice и Scratch;

4) умений составлять алгоритмы решения задач с использованием технологий объектно-ориентированного программирования в визуальных средах Alice и Scratch;

5) понимания принципов и ключевых понятий, лежащих в основе работы с визуальными учебными средами программирования Alice и Scratch.

– когнитивно-операционный компонент:

1) готовности к использованию возможностей языка программирования C++ для изучения объектно-ориентированного программирования;

2) способности настраивать и работать с Microsoft Visual Studio 2012/15;

3) умения создавать консольное приложение, реализующее разработанный алгоритм – программу на языке программирования C++;

4) готовности создавать компьютерные приложения с использованием объектно-ориентированной парадигмы.

– личностно-рефлексивный компонент:

1) умения работать в коллективе;

2) умения презентовать свои идеи и результаты разработки компьютерных приложений с использованием инструментальных сред Alice, Scratch и C ++;

3) аналитического (рефлексия, практический интеллект, анализ проблем, логические суждения, опыт планирования), концептуального (применение концепций, распознавание моделей, интуиция, идентификация проблемы и т.д.) и критического мышления (анализ собственного опыта).

Таблица 6.

Цели обучения бакалавров прикладной информатики основам объектно-ориентированного программирования

Курс	Компоненты компетенции			
	мотивационно-ценностный	**организационно-содержательный**	**когнитивно-операционный**	**личностно-рефлексивный**
«Программирование для начинающих»	формирование готовности к освоению базовых концепций, теории, технологий и конкретных средств объектно-ориентированного программирования	формирование теоретических знаний и познавательной активности, необходимых для изучения объектно-ориентированного программирования		формирование умений работать в коллективе, презентовать свои идеи и результаты разработки компьютерных приложений с использованием инструментальных сред
«Информатика и программирование»	формирование готовности к использованию знаний, умений, навыков и личностных качеств в области объектно-ориентированного программирования для решения профессиональных задач по разработке компьютерных программ		формирование знаний и умений в области конкретного объектно-ориентированного языка, определяющие способность бакалавра прикладной информатики вести профессиональную разработку компьютерных программ	формирование навыков аналитического, концептуального и критического мышления

Содержание обучения

Для создания методики обучения основам объектно-ориентированного обучения бакалавров прикладной информатики с использование визуальных учебных сред было подобрано содержание обучения. Как отмечают

И.Е. Востроknутов и Н.Г. Саблукова, процесс разработки структуры содержания обучения программированию в визуальных средах является итерационным [36, с. 19]. Прежде чем его подробно представить, уточним требования к содержанию учебного материала в рассматриваемой нами области знаний. Эти требования, с нашей точки зрения, могут быть представлены так:

1. Объем учебного материала должен охватывать все темы, без изучения которых сложно понять особенности объектно-ориентированного программирования.

2. Учебный материал должен быть четко структурированным, логично и последовательно изложенным.

3. Содержание учебного материала должно давать представление об основных особенностях объектно-ориентированной методологии программирования.

4. Учебный материал должен быть достаточно детализированным для обеспечения прочности знаний, умений и навыков студентов.

5. Учебный материал должен быть изложенным на доступном для студентов уровне.

Подбирая учебный материал, мы руководствовались следующими принципами, на которые опирается дидактика:

– изучение от простого к сложному;

– преемственности и последовательности;

– посильности;

– связи теории с практикой;

– индивидуализации и дифференциации обучения [118].

Рассмотрим подробнее, как реализуются указанные дидактические принципы в процессе изучения основ объектно-ориентированного программирования. При построении содержания разрабатываемой нами методики акцент был сделан на принцип «от простого к сложному». Сначала рассматриваются основные понятия, а на последующих занятиях учебный

материал постепенно усложняется. В то же время учебный материал каждой следующей темы базируется на предыдущих темах.

На основании анализа содержания и особенностей объектно-ориентированного программирования мы отобрали материал двух учебных дисциплин, направленных на обучение основам объектно-ориентированного программирования бакалавров прикладной информатики. В нашем случае эти дисциплины назывались «Программирование для начинающих», а также «Информатика и программирование» (таблица 7). Названия дисциплин, как представлено выше, не существенны и могут меняться в соответствии с требованиями конкретных образовательных программ, разработанных на основе стандартов подготовки бакалавров по направлению подготовки «Прикладная информатика».

Таблица 7

Содержание дисциплин «Программирование для начинающих» и «Информатика и программирование»

<table>
<tr><th>Дисциплина «Программирование для начинающих»</th><th>Дисциплина «Информатика и программирование»</th></tr>
<tr><td>История развития и использования программной среды Alice. Знакомство с интерфейсом Alice. Создание рабочего мира. Обработка стилей и свойств объектов. Методы, процедуры и функции. Структура программы в среде Alice. Использование языковых конструкций для записи программы.</td><td rowspan="2">Элементы объектно-ориентированного программирования и некоторые особенности языка С ++. Основные принципы технологии ООП. Классы в языке С ++. Члены класса – данные и функции, права доступа к членам класса. Примеры определения классов. Инкапсуляция - механизм сокрытия данных. Уровни доступа к членам класса. Класс как тип данных. Вызов методов класса. Конструкторы и деструктор класса. Автоматическая инициализация. Конструктор по умолчанию. Конструктор с параметрами. Деструкторы
Разработка программного проекта по объектно-ориентированной технологии</td></tr>
<tr><td>История развития и использования программной среды Scratch. Интерфейс Scratch. Работа со сценой. Блоки и область скрипты. Создание анимации для объекта. Костюмы объекта. Использование нескольких сцен. Управление объектами с помощью клавиш на клавиатуре. Звуковые эффекты. Типы данных. Переменные. Списки. Выражения.</td></tr>
</table>

Обучение основам объектно-ориентированного программирования бакалавров прикладной информатики с использованием визуальных учебных сред Alice и Scratch в рамках дисциплины «Программирование для начинающих» позволяет познакомить студентов с объектно-ориентированной парадигмой программирования, в соответствии с которой программа представляется в виде совокупности объектов, которые являются главным понятием данной методологии. Объект обладает некоторым внешним видом или свойствами, которые выражаются в значениях его переменных, и методами, представленными в виде его процедур. Свойства и методы не существуют обособленно друг от друга, а объединены вместе, образуя объект как качественно иную алгоритмическую структуру. Методы окружают свойства объекта, не позволяя напрямую обращаться к ним или изменять их значения. Свойства – инкапсулированы в объект. Доступ к ним осуществляются только посредством методов, предоставляемых объектом.

При этом, визуальные учебные среды программирования Alice и Scratch не предусматривают углубления в механизмы наследования и полиморфизма, которые реализуются объектно-ориентированными языками программирования. Они реализуют простую объектно-ориентированную модель, когда для понимания функционирования и создания программы достаточно приведенной информации об объектах. Более глубокое изучение объектно-ориентированного подхода продолжается уже в связанной дисциплине «Информатика и программирование», где предполагается изучения языка C++.

Содержание обучению конкретизируется через задания, которые предлагаются студентам. В дисциплинах «Программирование для начинающих» и «Информатика и программирование» должны быть предусмотрены виды и типы учебных заданий, с помощью которых будут сформированы соответствующие умения и навыки. Так как современное программирование предусматривает необходимость решения огромного разнообразия задач, мы считаем, что подготовка бакалавров прикладной

информатики в процессе обучения основам объектно-ориентированного программирования должна базироваться на разработке и предложении студентам различных типов задний, что позволит сформировать профессиональные умения и навыки. А для обеспечения высокого качества результатов обучения, соответствия системы заданий базовым положениям предложенной нами методики, целесообразно разработать классификацию учебных заданий; определить назначение каждого типа заданий и их необходимое количество в зависимости от важности, сложности, времени и запланированной трудоемкости рамках двух дисциплин – «Программирование для начинающих» и «Информатика и программирование».

Анализ существующих учебных пособий и сборников задач по программированию показывает, что разные авторы, планируя задания по основам объектно-ориентированного программирования, ведут речь про задания, системы упражнений и задач. Распределение таких заданий на собственно «задания», «упражнения» и «задачи» во многом условно и интуитивно.

Классификация учебных заданий рассмотрена в работах А.О. Бурдина [28], Е.Я. Голанта [39], Н.Б. Истоминой-Кастровской [65], В.А. Онищука [101], М.Э. Писоцкой [120], А.И. Умана [164], Л.И. Гриценко [44], Е.С. Павловой и др. [1, 2, 112]. Все перечисленные авторы предлагают классификации, ориентированные на деятельность студента, преподавателя, содержание и структуру изучаемого, а также структурно-компонентный состав задач. В данном исследовании, мы будем исходить из того, что классификация учебных заданий в процессе обучения основам объектно-ориентированного программирования (дисциплины «Программирование для начинающих» и «Информатика и программирование») должна быть ориентирована на: 1) вид деятельности студента (выбор структур данных, построение алгоритма, оценка эффективности программы, изучение языка программирования и т.д.); 2) характер деятельности студента

(репродуктивный, репродуктивно-поисковый, творческий). Два указанных аспекта позволяют решить вопрос по упорядочению задач в процессе обучения программированию.

Под учебными заданиями по дисциплинам «Программирование для начинающих» и «Информатика и программирование» будем понимать:

– вопросы;

– упражнения различного характера;

– текстовые задачи.

Опираясь на работу М.А. Даниловой и М.Н. Скаткина [46] отметим, что учебные задания предназначены для обеспечения:

– усвоения содержанием учебного материала курсов;

– усвоения знаний и способов деятельности на уровне восприятия, осознания и запоминания;

– усвоения знаний и умений на уровне применения по образцу, в знакомой ситуации;

– усвоения знаний и умений на уровне творческого применения;

– развития у студентов таких качеств, как интеллектуальная активность, уверенность в своих знаниях, выработке у них умения оперировать полученными знаниями;

– оценки значимости результатов учебной деятельности;

– формирования положительной мотивации обучения, потребности к самообразованию, навыков контроля и самоконтроля, умений и навыков коллективной работы над одним общим заданием.

При этом будем разделять и точку зрения ряда авторов о том, что методика обучения должна предполагать конструирование не отдельных заданий, а их систем [36].

Так, Е.С. Павлова, изучая вопросы построения методики формирования одаренности обучающихся при подготовке к олимпиадам по информатике, отмечает, что в «качестве основы для проведения занятий целесообразно использовать не отдельные задачи, а комплексные системы задач» [114].

Согласно автору, разнообразие наборов задач, входящих в данные системы задач, позволяют:

– постепенно усложнять изучаемый материал;

– поэтапно увеличивать объем работы;

– повышать уровень самостоятельности;

– привлекать элементы теории для решения познавательных задач;

– обучать способам рассуждения (как по образцу, так и самостоятельно) с учетом принципа вариативности задач;

– формировать важнейшие характеристики творческих способностей: беглость мысли (количество идей, возникающих за единицу времени), гибкость ума (способность переключаться с одной мысли на другую), оригинальность (способность находить решения, отличающиеся от общепринятых); любознательность (чувствительность к проблемам в окружающем мире), умение выдвигать и разрабатывать гипотезы.

А.И. Лапо, рассматривая методику построения систем задач по программированию в школьном курсе информатики, отмечает, что система задач должна удовлетворять принципам открытости, сложности и нелинейности. Такая система задач должна строиться по следующим этапам:

– определение фундаментальных составляющих системы;

– выбор родовых задач, которые покрывают фундаментальные составляющие системы;

– разработка индивидуальных задач, соответствующих родовым;

– разработка индивидуальных задач, требующих применения опыта решения родовых задач в измененных или новых условиях [81].

Е.В. Кудрина и М.В. Огнева для обучения программированию предлагают использовать примеры и наборы упражнений в логической взаимосвязи с примерами. При этом акцент делать не только на освоении языка программирования С++, но и на знакомстве с различными технологиями программирования, что в дальнейшем позволит перейти к изучению объектно-ориентированного программирования [74, 98].

По нашему мнению, система заданий для обучения основам объектно-ориентированного программирования должна содержать задания, которые способствуют:

– развитию логического, алгоритмического и аналитического мышлений;

– знакомству с разными стилями мышления и методами, которые используются для решения различных задач;

– развитию навыков принятия решений;

– формированию соответствующих знаний, умений и опыта практической деятельности студентов в области современного программирования.

Под системой заданий для обучения основам объектно-ориентированного программирования будем понимать вопросы, упражнения различного характера, текстовые задания, содержащие цель, условие и требования к ожидаемому результату. Учебные задания по основам объектно-ориентированного программирования, на наш взгляд, предназначенные для обеспечения:

– усвоения содержания учебного материала основ объектно-ориентированного программирования;

– общепризнанных уровней усвоения знаний и умений, а именно:

а) усвоение знаний и способов деятельности на уровне восприятия, осознания и запоминания;

б) усвоение их на уровне применения по образцу, в знакомой ситуации;

в) усвоение их на уровне творческого применения.

– развития у студентов таких качеств, как интеллектуальная активность, уверенность в своих знаниях, выработке у них умения оперировать полученными знаниями;

– оценки значимости результатов учебной деятельности;

– формирования положительной мотивации обучения сновам объектно-ориентированного программирования, потребности к самообразованию, навыков контроля и самоконтроля, умений и навыков коллективной работы над одним общим задачам.

Соответственно, в процессе обучения основам объектно-ориентированного программирования все задания должны содержать различные типы задач, которые наиболее полно раскрывают содержание темы, которая изучается и соответствуют индивидуальным способностям студентов.

Опираясь на таксономию целей Блума [163], а также таксономию учебных задач Д. Толлингеровой [161, 162], выделим следующие типы учебных заданий для обучения основам объектно-ориентированного программирования:

1) задания-упражнения, направленные на обеспечение усвоения учебного материала и формирования способов деятельности на уровне восприятия, понимания и запоминания;

2) задания на усвоение способов деятельности, направленные на обеспечение усвоения учебного материала и способов деятельности на уровне применения по образцу, в знакомой ситуации;

3) задания на составление тестовых вопросов по определенной теме учебного материала, направленных на качественное изучение учебного материала курса, формирование таких качеств, как лаконичность и точность выражения мнений, ответственность за результаты труда;

4) задания-исследования, направленные на формирование у студентов навыков работы с учебной, научной литературой и с периодическими изданиями с целью поиска необходимых сведений для выполнения учебных задач, оценки значимости найденных сведений и т.д.;

5) задания по разработке программного продукта, направленные на формирование у студентов навыков и опыта разработки реальных

программных продуктов с использованием технологий проектирования, а также развитие творческих способностей студентов.

Рассмотрим более подробно с примерами каждый из перечисленных видов заданий.

Задания-упражнения выполняются студентами после усвоения нового учебного материала для его закрепления на протяжении всего периода обучения основам объектно-ориентированного программирования в рамках лабораторной работы или после лекции, а также при выполнении заданий для самостоятельной работы. Кроме того, с помощью заданий-упражнений преподаватель может осуществлять контроль знаний и определять уровень усвоения учебного материала студентами во время лабораторных работ, контрольных работ, на коллоквиуме и т.п. Примеры заданий-упражнений для определения уровня усвоения учебного материала студентами во время лабораторных работ учебного материала по дисциплинам «Программирование для начинающих» и «Информатика и программирование» приводятся в таблице 8. Более подробно примеры заданий-упражнений представлены в приложении 1.

Таблица 8

Примеры заданий-упражнений для определения уровня усвоения учебного материала студентами во время лабораторных работ по дисциплинам «Программирование для начинающих» и «Информатика и программирование»

Дисциплина «Программирование для начинающих»		**Дисциплина «Информатика и программирование»**
Программная среда «Alice»	**Программная среда «Scratch»**	**Язык программирования C++**
Ознакомиться с методами (процедуры и функции), которые являются частью классов, которые используются в	Создать проект, в котором реализован поворот объекта по часовой стрелке при нажатии на кнопку мыши.	Построить класс Worker (рабочий) без потомков, содержащий объявления полей данных (name - фамилия, worker_id - код, salary - размер зарплаты рабочего) и определения метода класса – функцию вне класса

Дисциплина «Программирование для начинающих»		Дисциплина «Информатика и программирование»
Программная среда «Alice»	**Программная среда «Scratch»**	**Язык программирования С++**
анимации. Обратить внимание, что в Alice объекты представлены с набором встроенных процедур.		(Show_worker () - вывод информации о работника в консольном режиме)

Задания на усвоения учебного материала, являются одним из наиболее важным видом заданий в процессе обучения основам объектно-ориентрованного программирования, которые позволяют формировать систему знаний, умений и навыков бакалавра прикладной информатики. При выполнении этих заданий студенты:

– приобретают навыки структурирования данных и реализации простых программ, навыки распространения алгоритма решения задачи на круг типовых задач;

– могут овладеть методами выбора наиболее эффективного метода решения поставленной задачи;

– знакомятся с различными методами реализации программы.

Данные задания предполагают написание программы, ее настройку, документирование текста программы, анализ результатов ее выполнения. Задания требуют от студента больше интеллектуальных усилий и времени, чем упражнение. Задания целесообразно использовать в цикле лабораторных работ, при организации самостоятельной работы и для контрольных работ. Примеры заданий, направленных на обеспечение усвоения учебного материала (задания на формирование умений и навыков реализации программы методом структурной декомпозиции) приводятся в таблице 9. Более полный перечень заданий данного типа приводится в приложении 2.

Таблица 9

Примеры заданий на усвоение способов деятельности,
направленных на усвоение учебного материала
по дисциплинам «Программирование для начинающих»
и «Информатика и программирование»

Дисциплина «Программирование для начинающих»		**Дисциплина «Информатика и программирование»**
Программная среда «Alice»	**Программная среда «Scratch»**	**Язык программирования С++**
Создать мир и программно реализовать: дельфин (Dolphin) плавает в океане. На некотором расстоянии летает птица (Falcon). Через некоторое время птица возвращается к дельфину, а затем летит от него и возвращается к нему (на некотором расстоянии).	Реализовать программно следующий алгоритм: Объект Кот издалека подходит к мячу, а затем ударяет по нему и мяч улетает. Объект Кот кричит: «Yes!». Применить эффекты приближения и удаления.	Создать программу, которая объявляет класс с тремя полями данных (name, воркер_id, salary) и функцией-членом (show_worker ()), которую реализовано вне класса. Все элементы класса объявим, как общедоступные. В программе необходимо создать два объекта для класса worker, предоставим значение элементам данных и с помощью функции show_worker () выведем информацию о рабочих в консольном режиме.

Задания на составление тестовых вопросов по определенной теме учебного материала в процессе обучения программирования формируют умения точно выражать вопросы и лаконично излагать мысли, воспитывают чувства ответственности за результаты труда непосредственно в процессе обучения (табл.10).

Таблица 10

Задания на составление тестовых вопросов по определенной теме учебного материала в процессе обучения программирования

Дисциплина «Программирование для начинающих»		**Дисциплина «Информатика и программирование»**
Программная среда «Alice»	**Программная среда «Scratch»**	**Язык программирования С++**
Подготовить тестовые вопросы и варианты ответов к ним по теме «Структура программы в среде Alice»	Подготовить тестовые вопросы и варианты ответов к ним по теме «Интерфейс Scratch»	Подготовить тестовые вопросы и варианты ответов к ним по теме «Проектирование классов и создание объектов. Реализация программы»

В приложении 3 приведены примеры тестовых вопросов, предложенных студентами.

Задания-исследования позволяют сформировать у студентов такие качества, как поиск материала по теме реферата или доклада, анализ собранных сведений, способность делать выводы, опыт публичного выступления и опыт овладения приемами публичной защиты работы. Также формируются навыки работы с учебной, научной и специальной литературой, с периодическими изданиями, интернет-ресурсами и т.п. Примеры заданий-исследований на подготовку докладов представлены в таблице 11. Более полный перечень таких заданий приводится в приложении 4.

Таблица 11

Примеры заданий-исследований на подготовку докладов
по дисциплинам «Программирование для начинающих»
и «Информатика и программирование»

Дисциплина «Программирование для начинающих»		**Дисциплина «Информатика и программирование»**
Программная среда «Alice»	**Программная среда «Scratch»**	**Язык программирования C++**
Подготовьте доклад по предложенной теме. Примерные темы докладов: 1. История развития и использования программной среды Alice. 2. Интерфейс программной среде Alice 3. Создание рабочего мира в программной среде Alice	Подготовьте доклад по предложенной теме. Примерные темы докладов: 1. Базовые программные конструкции и их реализация в среде Scratch 2. Разветвления в Scratch 3. Циклы в Scratch	Подготовьте доклад по предложенной теме. Примерные темы докладов: 1. Парадигмы программирования 2. Основные понятия программирования 3. Интегрированная среда разработчика C++

Задания по разработке программного продукта, как и задания на усвоение учебного материала, направлены на решение наиболее важных задач в обучении бакалавров прикладной информатики основам объектно-ориентированного программирования. В частности, выполняя такие задания, студент получает опыт разработки полноценного программного продукта,

усваивает связь теории и практики, знакомится со сферой применения полученных знаний.

Создание программного продукта предполагает наличие, как минимум, знаний базовых конструкций определенного языка программирования, стандартных алгоритмов поиска и сортировки, методов построения алгоритмов; навыков кодирования, отладки, тестирования программ; умение интегрировать программные модули, созданные отдельно. Задания данного вида предполагают решение одной сложной, но посильной для студентов задачи, и выполняются в конце изучения курса программирования (в нашем случае это конец учебного семестра – подведение итогов по дисциплинам «Программирование для начинающих» и «Информатика и программирование»). Примеры заданий по разработке программного продукта приводятся в таблице 12. Более подробно такие примеры приведены в приложении 5.

Таблица 12

Примеры заданий по разработке программного продукта
по дисциплинам «Программирование для начинающих» и
«Информатика и программирование»

Дисциплина «Программирование для начинающих»		**Дисциплина «Информатика и программирование»**
Программная среда «Alice»	**Программная среда «Scratch»**	**Язык программирования C++**
Разработка программного продукта		
Создать программу – проект, с помощью которой реализуется фрагмент сказки «Алиса в стране чудес»	Создать программу – проект, с помощью которого реализуется игра «Тетрис»	Создать программу – проект, с помощью которого реализуется приложение «Калькулятор».

Средства обучения

Важнейшим компонентом методики обучения основам объектно-ориентированного программирования бакалавров прикладной информатики,

непосредственно и очень тесно связанным с содержанием обучения, являются средства обучения.

П.И. Пидкасистый определяет средства обучения как объекты, созданные человеком, используемые в образовательном процессе в качестве носителей учебной информации и инструмента деятельности педагога и обучающихся для достижения поставленных целей обучения, воспитания и развития [119]. Соответственно, средства обучения – это объекты, которые в учебном процессе выполняют роль сенсомоторных стимулов, воздействующих на органы чувств учащихся и облегчающих им непосредственное или косвенное познание мира (В.А. Сластенин [156]).

В данном исследовании будем исходить из того, что средства обучения, вместе с учебно-методическими материалами (учебники, учебные пособия для студентов, методические пособия, рекомендации для преподавателей) образуют целостную систему – учебно-методический комплекс дисциплины. Опишем подробнее каждое из указанных средств более подробно.

Средства учебно-методического обеспечения

Одним из основных средств обучения, относящихся к средствам данного вида, является учебник. Учебник – это учебное издание, содержащее систематизированное изложение учебной дисциплины, соответствует программе дисциплины и официально утвержденное в качестве такой вид издания [122]. Следовательно, учебники по основам объектно-ориентированного программирования должны соответствовать определенным требованиям и формировать научное мировоззрение и профессиональные качества будущих выпускников. Среди таких требований можно выделить то, что трактовка теоретических понятий, терминология и символика должны отвечать строго направлению программирования, а учебный материал должен быть представлен в доступном и научном стилях.

Учебное пособие – это средство обучения, источник учебной информации, вид учебной литературы, дополняющий учебник. Учебное пособие – это учебная книга, дополняющая или расширяющая учебник по отдельным вопросам или темам учебной программы [97]. На рисунке 4 представлена сравнительная характеристика понятий «учебник» и «учебное пособие», предложенные Е. Н. Овчинниковой [97].

Рисунок 4. Сравнительная характеристика понятий «учебник» и «учебное пособие», предложенные Е. Н. Овчинниковой [97]

Проанализируем некоторые учебники и учебные пособия по основам объектно-ориентированного программирования согласно следующим критериям: содержание, язык программирования, практические задания, самоконтроль (тесты).

Учебник Г.С. Ивановой [64] содержит основные теоретические положения разработки программного обеспечения с использованием

структурного и объектно-ориентированных подходов. Подробно рассмотрены основные приемы решения задач различных классов, в том числе приемы создания и обработки динамических структур данных, без которых невозможно современное программирование. Особое внимание уделено оценке точности получаемых результатов и анализу вычислительной сложности алгоритмов и методов. Данный учебник содержит достаточное количество примеров и поясняющих рисунков, что помогает лучшему усвоению материала. Учебник издан на русском языке и рекомендован для студентов вузов, обучающихся по специальностям, связанными с информатикой.

В учебном пособии М.П.Белова [17] рассмотрены понятие алгоритма, способы описания алгоритмов, основные свойства и характеристики алгоритмов, алгоритмические структуры и правила их оформления, методы разработки алгоритма, приводятся примеры. Учебное пособие состоит из двух частей. В первой части излагаются теоретические сведения об алгоритмах. Вторая часть посвящена практической работе – алгоритмизации и программированию. Каждая часть разделена на главы, которые, в свою очередь, разбиты на темы. Для закрепления теоретического материала автор рассматривает задачи и предлагает пояснения к ним. В учебном пособии есть также задачи для самостоятельного выполнения, но их недостаточно, и нет распределения этих задач по уровням сложности. Учебник издан на русском языке.

В настоящее время появилось достаточно много хороших иностранных учебников по алгоритмизации и программированию, переведенных на русский язык. В частности, это учебники:

– Т. Пратт, М. Зелковиц «Языки программирования: разработка и реализация» [124];

– Р. Себеста «Основные концепции языков программирования» [149];

– Т. Кормен, Ч. Лейзерсон, Р. Ривест «Алгоритмы: построение и анализ» [71];

– Дж. Макконнелл «Анализ алгоритмов. Вводный курс» [88];

– А.В. Ахо, Д.Е. Хопркрофт, Дж. Ульман «Построение и анализ вычислительных алгоритмов» [10], «Структуры данных и алгоритмы» [11];

– Б. Керниган, Д. Ритчи «Язык программирования С» [68];

– А. Фьюер «Задачи по языку Си» [169];

– Б.В. Керниган, Р. Пайк «Практика программирования» [67];

– С. Прата [123] «Язык программирования С++. Лекции и упражнения» и др.

Учебники, которые переведены с английского языка на русский язык, направлены на изучение конкретных дисциплин, которые отличаются от курса, рассматриваемого нами в данном исследовании. Но это не уменьшает значимости этих учебников, так как вполне возможно использовать некоторые их разделы.

Имеются также учебники только на английском языке, которые нам необходимы для нашего исследования. В частности, это учебники:

– Joel Adams «Alice 3 in Action: Computing Through Animation» [204];

– Wanda Dann, Steve Cooper and Randy Pausch «Learning to Program with Alice» [222];

– Quintin Cutts, Sarah Esper and Beth Simon « Expeditions through Alice » [211];

– Tony Gaddis «Starting Out with Alice: A Visual Introduction to Programming» [213];

– Marji Majed «Learn to Program with Scratch» [215].

Так как современное развитие информационных технологий протекает довольно стремительно, то бакалаврам прикладной информатики следует следить за данным направлением и пополнять профессиональные знания путем использования сети Интернет. В качестве таких источников информации можно рекомендовать сайты проектов Alice (http://alice.org) и Scratch (http://cratch.mit.edu), библиотеку MSDN (http://msdn.microsoft.com),

сайт национального открытого университета «Интуит» (http://intuit.ru), MIT OpenCourseWare (http://ocw.mit.edu) и др.

Технические и программные средства обучения

Под техническими средствами обучения будем понимать технические устройства, которые обеспечивают, прежде всего, представление учебной информации, контроль знаний, формирование практических умений и навыков. Как отмечают О.В.Фадеев, Д.В.Пешков, техническое обеспечение включает в себя адаптацию, совершенствование и разработку компьютерной техники, используемой для передачи информации, обратной связи, контроля знаний, организации самостоятельных занятий, обработки и документирования информации» [165].

Проблема применения технических средств обучения является одной из важных в методике преподавания дисциплин. Среди современных технических средств обучения выделяют технические средства на базе информационных технологий. Введение в учебный процесс технических средств обучения основам объектно-ориентированного программирования бакалавров прикладной информатики открывает значительные возможности для осуществления индивидуального подхода. Это дает возможность влиять на студентов с учетом их индивидуальных психических особенностей (памяти, воображения, наблюдательности), развивать критическое мышление.

Под техническими средствами обучения программированию будем понимать программно-аппаратные средства и устройства, функционирующие на базе компьютерной техники, а также современные средства и системы информационного обмена, которые обеспечивают операции по поиску, сбору, накоплению, хранению, обработке, представлению, передачи информации. К ним принадлежат:

– компьютеры;

– комплекты терминального оборудования для компьютеров всех аудиторий;

– локальные компьютерные сети;

– устройства ввода-вывода;

– средства и устройства манипулирования аудиовизуальной информацией (на базе технологи мультимедиа и систем «виртуальная реальность»);

– современные средства связи;

– программные комплексы (языки программирования, трансляторы, компиляторы, операционные системы, пакеты прикладных программ общего и учебного о назначения и т.д.).

Необходимым условием обучения программированию является применение в учебном процессе одной или нескольких сред программирования. Как представлено в прошлых разделах диссертации, свой выбор для обучения объектно-ориентированному программированию в рамках дисциплины «Информатика и программирование» мы остановили на языке C++. В сочетании с необходимостью поддержки операционной системы Windows это определило и выбор среды программирования – Microsoft Visual Studio. В частности – это бесплатная версия Microsoft Visual Studio Express, которая имеет некоторые ограничения в области программных библиотек и функционала, не имеющие существенного значения для изучения языка C++.

Alice и Scratch, использование которых рассматривается нами в курсе «Программирование для начинающих», составляют неотъемлемую часть разрабатываемой методики. Это визуальные учебные среды объектно-ориентированного программирования с интегрированной средой разработки, которые представляют собой визуальную среду программирования для создания компьютерной анимации с использованием объектов. Alice и Scratch способствуют решению следующих проблем:

– дальнейшего обучения объектно-ориентированного программирования со сложной семантикой, в частности С ++. Пользователь создает проект, размещает в нем объекты, разрабатывает программный код, используя встроенные конструкции;

– поддержки объектно-ориентированного подхода в программировании, управления моделями данных;

– подготовки студентов к современному профессиональному программированию.

Программные средства организации образовательного процесса

Программные средства организации образовательного процесса относят к инновационным средствам – это автоматизированные системы и программные продукты, созданные для поддержки процесса обучения, в том числе – обучения именно программированию. Данные средства отражают предметную область обучения объектно-ориентированному программированию, реализуют те или иные технологии обучения, обеспечивают условия для осуществления и компьютерной поддержки различных видов учебной деятельности.

Указанные программные средства используются в процессе обучения с целью развития личности студента, интенсификации процесса обучения и делятся по признакам функционального и методического назначения.

По функциональному назначению программные средства обучения основам объектно-ориентированного программирования состоят из:

– педагогических программных средств, предназначенных для организации и компьютерной поддержки учебно-познавательной деятельности (платформы Piazza и OpenClass);

– инструментальных программных средств, предназначенных для подготовки или генерирования учебно-методических и организационных

материалов (подготовка презентационных материалов преподавателем средствами Prezi, PowerPoint).

Так, для обучения основам объектно-ориентированного программирования бакалавров прикладной информатики мы предлагаем использовать онлайн-платформы Piazza и OpenClass – платформы, с использованием которых студенты обучаются под руководством преподавателей. С помощью них можно легко ответить на вопросы, а преподаватель «управляет» учебными материалами, а также «отслеживает» участие студентов в процессе обучения.

Рассмотрим более подробно структуру и процесс работы с данными онлайн-платформами. Обе платформы предоставляют широкий спектр возможностей для обучения, в том числе и для обучения программированию [109, 110]. Преподаватель участвует в составлении собственного курса с теоретическими материалами, задачами. Также данные платформы предоставляют возможность внесения в них изменений преподавателем. Форма платформ – блочная, то есть отдельные блоки могут заменяться, дополняться в ходе обучения. Как показывает практика, учебные электронные курсы целесообразно представлять в виде совокупности модулей, которым предшествует вступительная и организационная части.

В свою очередь, каждый модуль состоит из содержательно законченных частей, каждая из которых содержит теоретический материал. Задачи для практической работы, вопросы и тестовые задания для самоконтроля, промежуточный контроль, сопровождается подсказками и методическими пособиями. Ядро каждого модуля содержит тематический контроль. Кроме того, каждый модуль электронных курсов Piazza и OpenClass содержит:

- тему и цель обучения,
- ключевые слова и понятия,
- теоретический материал,
- глоссарий,

• практические задания и лабораторные работы,

• список рекомендуемой литературы,

• задания для самостоятельного выполнения и вопросы для самоконтроля,

• задания для тематического контроля.

Внешний вид страниц электронных курсов OpenClass и Piazza представлен на рисунках 5 и 6.

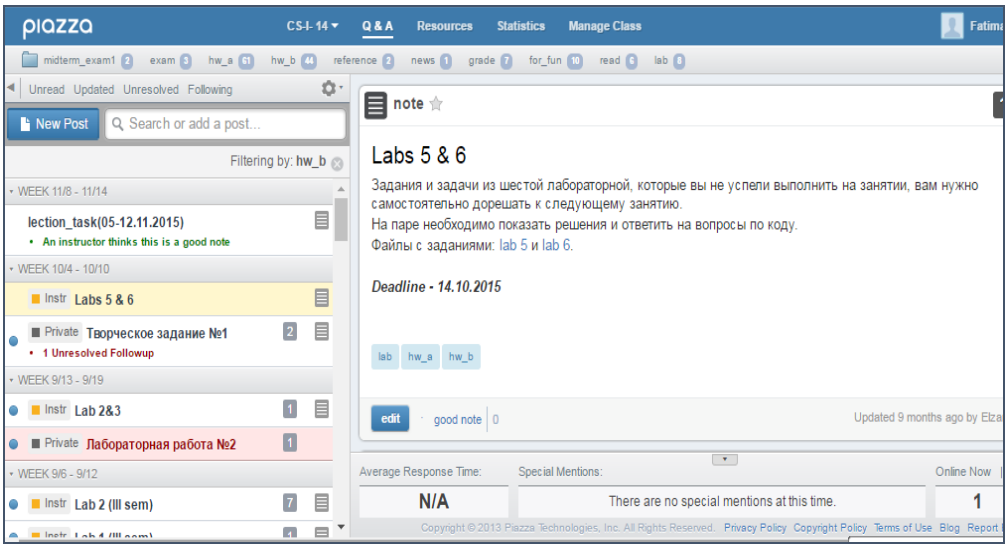


Рисунок 5. Рабочая область онлайн платформы Piazza

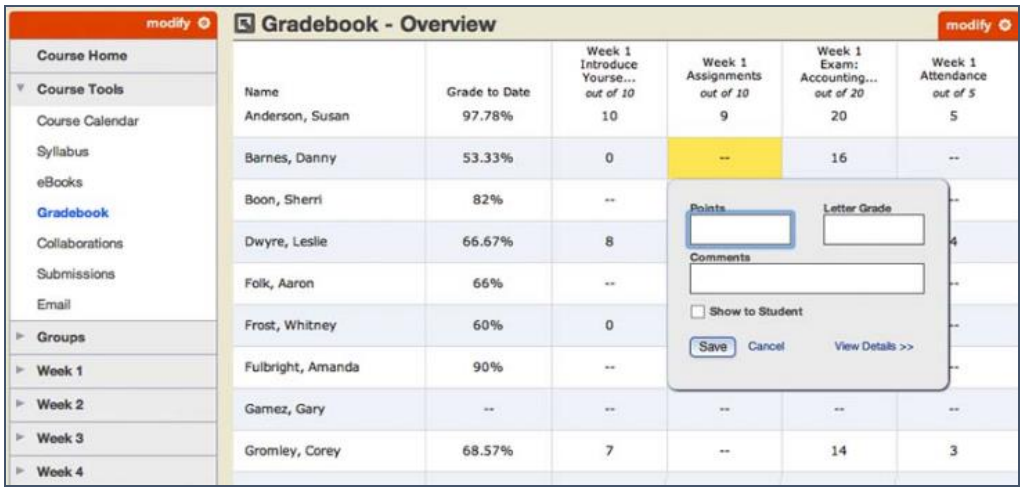


Рисунок 6. Рабочая область онлайн платформы OpenClass

С помощью онлайн платформ Piazza и OpenClass полностью автоматизируется не только деятельность студентов в процессе обучения, но и деятельность преподавателя, в частности:

– подготовка преподавателя к чтению лекций и проведения лабораторных занятий;

– проведение коллоквиумов и контрольных работ, прием зачетов, проведение лабораторных занятий;

– организация учебно-исследовательской работы студентов.

В качестве инструментальных программных средств, предназначенных для подготовки или генерирования учебно-методических и организационных материалов, могут быть использованы такие известные средства, как PowerPoint и Prezi.

PowerPoint – это вид программного обеспечения для создания презентаций, который позволяет демонстрировать цветные слайды с текстом и изображением, с использованием простой анимации. Это один из многих типов программ для создания презентаций [108].

Prezi – это инструмент для создания презентаций в нелинейной структуре. В отличие от PowerPoint в среде Prezi работа осуществляется не со слайдами, а со всей рабочей областью, на которой размещается необходимый контент в виде заголовков, текстовых блоков, стикеров, геометрических фигур, картинок, видеороликов, звуковых дорожек. То есть, Prezi имеет сплошное полотно, на котором сосредоточен весь процесс работы по созданию презентации. На нем происходит перемещение, приближения и удаления воображаемой камеры. Благодаря этому, зрители интуитивно понимают, на каком уровне обобщения и детализации они находятся в данный момент. Несмотря на это, единый образ всей презентации позволяет докладчику в любой момент выступления перейти к нужному кадре доклада, что значительно усиливает методические возможности докладчика, способствуя более эффективному запоминанию и усвоению материала благодаря нелинейному представлению материала [108, 218].

Методы обучения

Под методом обучения, вслед за А.И. Кузьминским, будем понимать способы совместной деятельности преподавателя и студента, направленные на достижение поставленных целей обучения, в частности программированию [77].

Современная педагогика насчитывает множество методов и подходов к их классификации. В рамках нашего исследования будем придерживаться классификации, предложенной И. В. Зайченко [60]. В этой классификации выделяются:

1. *Методы организации и осуществления учебно-познавательной деятельности:* словесный; наглядный; практический.

2. *Методы стимулирования учебно-познавательной деятельности:* устные и письменные проверки и самопроверки результативности овладения знаниями, умениями и навыками.

3. *Методы стимулирования учебно-познавательной деятельности:* определённые поощрения в формировании мотивации, чувства ответственности, обязательств, интересов в овладении знаниями, умениями и навыками.

Конкретизация методов обучения применительно к особенностям обучения программированию представлена в работах Ж. К. Нурбековой [95]. Автор предлагает следующие методы обучения программированию:

– метод ручной прокрутки программ (алгоритмов);

– метод раскрутки программ;

– метод программирования, как метод обучения данному разделу;

– метод обучения классическим алгоритмам и их использованию;

– метод демонстрационных примеров;

– метод проектов.

Обобщая разные подходы, в структуре разрабатываемой нами методики обучения основам объектно-ориентированного мы предлагаем использовать следующие методы обучения:

1. Объяснительно-иллюстративный метод – один из самых экономных способов передачи знаний. Суть данного метода заключается в том, что «преподаватель разными средствами предъявляет информацию об объекте изучения, а обучающиеся всеми органами чувств воспринимают ее, осознают и запоминают» [16, с.58].

2. Репродуктивный метод, позволяющий формировать умения и навыки студентов на уровне, достаточном для их применения в обычных условиях путем выполнения заданий по образцу [104, с.13].

3. Исследовательский метод. Предусматривает организацию поисковой творческой деятельности студентов путем постановки новых для них проблем и проблемных задач [4, с.91].

4. Словесные методы. Это методы беседы, рассказа, объяснения, лекции, дискуссии, работы с книгой и др. Все эти методы предполагают то, что материал для усвоения студент получает через слово [105].

5. Наглядные методы. При использовании данных методов учебные материалы студент получают с помощью демонстрации и иллюстрации [104].

6. Практические методы. Реализуются путем выполнения практических действий студентом, который получает некоторые сведения и в процессе их анализа приходит к тем знаниям, которые необходимо усвоить [45].

7. Метод проектов. Предполагает решение какой-либо проблемы, которая предусматривает использование различных средств обучения с одной стороны и необходимость применения различных знаний на практике, с другой. Результатом применения метода проектов должны являться осязаемые решения задач, например, готовый к использованию результат. В контексте рассмотрения метода проекта как педагогической технологии, можно отметить, что она предполагает собой использование поисковых,

исследовательских и иных проблемных методов, которые в большинстве случаев являются творческими [166, с. 377].

8. Эвристические методы. Это усвоение знаний и умений путем рассуждений, требующих догадки, поиска, изобретательности, что должно предусматриваться в задании. Основное назначение данного метода – постепенная подготовка обучаемых к самостоятельному выявлению и решению проблем [173].

9. Метод проблемного обучения. Подразумевает организацию учебных занятий, которые предполагают создание под руководством преподавателя проблемных ситуаций и активную самостоятельную деятельность студентов по их разрешению, в результате чего и происходит творческое овладение профессиональными знаниями, навыками, умениями и развитие мыслительных способностей [28].

Все перечисленные методы применимы для использования обучению объектно-ориентированному программированию при подготовке бакалавров прикладной информатики. Вместе с тем, основной акцент должен быть сделан на практические методы. Как отмечает Е. А. Борисова [24], при обучении курсам и разделам программирования целесообразно использовать практические методы (выполнение лабораторных работ, практикумов, решение задач), в процессе применения которых студенты не только получают новые знания, но и приобретают практические навыки. В деятельности студентов преобладает практическая работа (вещественные и умственные действия), в ходе которой особую роль играет самостоятельный мыслительный процесс, позволяющий осуществить поиск данных и парадигмы решения задачи. Роль преподавателя при этом заключается в чётком инструктаже, постановке целей лабораторных (практических) работ, контроле и корректировки деятельности студента в процессе выполнении заданий [24, с.46].

В данном исследовании в качестве методов обучения основам объектно-ориентированного программирования также были выделены метод

целесообразно подобранных задач, метод демонстрационных примеров, предложенные М. П. Лапчиком [84, 85], А. И. Бочкиным [26], В. В. Лаптевым и др. [82], которые по-нашему мнению способствуют именно формированию профессиональной компетенции в области объектно-ориентированного программирования.

Метод **целесообразно подобранных задач** рассматривается как метод, который используются для мотивации определенной деятельности студентов, а также для закрепления и изучения теоретического материала. В процессе решения этих задач, как в рамках дисциплины «Программирование для начинающих», так и дисциплины «Информатика и программирование» у бакалавра прикладной информатики формируются, прежде всего, организационно-содержательный и когнитивно-операционный компоненты компетенция в области объектно-ориентированного программирования.

В предложенной нами методике заметное место также занимает **метод демонстрационных проектов**, который позволяет студентам приобщиться к процессу разработки современного программного кода. Это осуществляется посредством первоначального анализа и дальнейшей доработки предлагаемого программного кода, выявления возможных ошибок, модификации и совершенствования программного продукта, что способствует самостоятельной разработке собственных проектов (дисциплины «Программирование для начинающих и «Информатика и программирование»).

При использовании **метода демонстрационных проектов** формируется компетенция в области объектно-ориентированного программирования в части *организационно-содержательного компонента*, что происходит посредством использования визуальных учебных сред Alice и Scratch (дисциплина «Программирование для начинающих). Демонстрационные проекты иллюстрируют определенные аспекты синтаксиса и семантики языка программирования, происходит знакомство с основами объектно-ориентированного программирования.

Для формирования профессиональной компетенции в области объектно-ориентированного программирования, однако, недостаточно задач и демонстрационных проектов, так как рассмотренные выше методы целесообразно подобранных задач и демонстрационных проектов не предусматривают достаточных условий для самостоятельности, поиска, объем полученных знаний и не находят своего использования в конкретных жизненных ситуациях. В связи с этим в рамках дисциплин «Программирование для начинающих и «Информатика и программирование» мы предлагаем применять также **метод проектов**, который предусматривает использование при обучении основам объектно-ориентированного программирования комплексных заданий, выполнение которых не ограничивается одним занятием.

В процессе обучения основам объектно-ориентированного программирования использование метода проектов приобретает особую значимость еще и потому, что способствует формированию всех компонент профессиональной компетенции, рассмотренных нами в п. 1.2. Во время работы над учебным проектом активно используется наблюдения, выдвигаются гипотезы, идет экспериментальная проверка, расширяется научное мировоззрение, что способствует преимущественному формированию таких компонент, как мотивационно-ценностный и личностно-рефлексивный. Организационно-содержательный, когнитивно-операционный компоненты формируются в результате выполнения проекта, получения конкретного результата, готового к использованию. Готовый проект должен пройти защиту, его можно расценивать как форму контроля.

Формы обучения

Форма обучения – это ограниченные в пространстве и времени взаимообусловленные деятельность преподавателя и студента [23]. Высшая школа России позволяет использовать такие организационные формы

обучения, как лекции, лабораторные работы, самостоятельную работу студентов, контрольные мероприятия и др.

Лекции предназначены для изучения теоретического материала, где материал излагается последовательно и систематично. В содержании каждой лекции имеются как внутренние, так и внешние логические связи. Проведение лекционных занятий по основам объектно-ориентированного программирования подразумевает развитие творческого мышления, познавательной активности студентов, интереса к изучаемой проблеме и преподаваемой учебной дисциплине.

Планирование и проведение лекций по дисциплинам «Программирование для начинающих и «Информатика и программирование» согласно разрабатываемой нами методике должно включать в себя следующие элементы:

1. Определение темы лекции.
2. Постановку цели лекции.
3. Разработку содержания (план лекции, материал лекции).
4. Подготовку средств наглядного сопровождения (MS PowerPoint).
5. Подготовку дидактического материала.
6. Составление списка рекомендованных источников.

Процесс проведения лекций делится на три основных этапа: 1) знакомство студентов с темой и планом лекции (10 мин); 2) выступление лектора с использованием презентации (65 мин); 3) вопросы и обсуждение лекционного материала (15 мин).

Лабораторные занятия по дисциплинам «Программирование для начинающих и «Информатика и программирование» предназначены для выработки практических умений и навыков применения теоретических знаний, полученных на лекциях. Планирование и проведение лабораторного занятия должно в себя включать:

1. Определение темы лабораторного занятия.
2. Постановку цели занятия.

3. Разработку содержания (методические рекомендации по выполнению лабораторной работы).

4. Подготовку средств для выполнения заданий (Piazza и OpenClass, Prezi, Power Point, Alice, Scratch, C ++).

5. Разработку дидактического материала (подготовленные преподавателем учебные пособия [151, 195, 196]).

6. Составление списка рекомендованных источников.

Лабораторная работа также предполагает самостоятельную работу студентов, в процессе которой с помощью учебно-исследовательской деятельности они повторяют, закрепляют и обобщают теоретические знания, практические умения и навыки. Каждую лабораторную работу студент должен обязательно сдать преподавателю. Отчет по лабораторной работе должен содержать название темы. К каждому заданию должны быть представлены полное описание задания, текст программы и ответы на контрольные вопросы. Кроме этого, должны быть представлены исходные коды программ в электронном виде. При сдаче работы преподавателю, студенту задается несколько теоретических вопросов по данной теме или о ходе выполнения решения практической задачи. Пример лабораторной работы представлен в приложении 6.

Относительно **самостоятельной работы** студентов, которая является обязательной формой организации учебного процесса в любом вузе, то согласно предлагаемой нами методике обучения, к каждой теме по дисциплинам «Программирование для начинающих и «Информатика и программирование» выделен ряд заданий [151, 195, 196], которые студенты обязательно должны выполнить и сдать преподавателю.

Практический опыт ученых Л.М. Левиной, Л.С. Пичковой, Т.В. Асламовой [87, 121, 8] подтверждает, что организация самостоятельной работы студентов требует инновационных подходов, что требует от студентов высокой самоорганизации, владение способами и методами получения знаний. Самостоятельная работа студентов включает в себя

подготовку к лекционным, лабораторным занятиям; выполнение определенных заданий по дисциплине и отдельным ее темам в течение семестра; подготовку контрольным работам; участие в олимпиадах, семинарах, конференциях.

Содержание самостоятельной работы студентов при обучении основам объектно-ориентированного программирования определяется программами учебных дисциплин «Программирование для начинающих» и «Информатика и программирование», разработанные по требованиям, соответствующими методическими материалами, практическими задачами и рекомендациями по их выполнению. Указанные материалы должны содержать инструкции о сроке, объеме, качестве усвоения материала с указанием учебных и научных изданий, используемых с этой целью, а также вопросы для самоконтроля, тесты, контрольные задания, примеры оформления самостоятельной работы. Результаты самостоятельной учебной работы каждого студента анализируются и оцениваются преподавателем.

Для организации самостоятельной работы студентов при обучении основам объектно-ориентированного программирования следует подбирать качественно новые учебные материалы, учитывающие специфику этого вида деятельности студентов. Одним из важных критериев отбора практических заданий для самостоятельной работы студентов по курсам является учет тех тем, которые выносятся на самостоятельную проработку.

Предлагаемая методика обучения также предусматривает наличие и выполнение дисциплинам «Программирование для начинающих и «Информатика и программирование» **контрольных мероприятий**:

1) тесты, которые каждый студент проходит при сдаче промежуточного контроля. Пример тестовых заданий приводится в приложении 7;

2) аудиторные контрольные работы – контрольная работа, которая проводится в аудитории с преподавателем в пределах 2 академических часов (пара). Пример такой работы приведен в приложении 8;

3) внеаудиторные контрольные работы – контрольная работа выдается

на выполнение студентам домой, устанавливаются сроки ее сдачи (приложение 9);

4) зачет – вид промежуточного контроля. Зачет можно получить автоматически, если студент выполняет все требования, которые ставит преподаватель в начале семестра.

5) экзамен – вид итогового контроля, который проводится в конце изучения дисциплины. Образец одного экзаменационного билета приведен в приложении 10. Оценку за экзамен можно получить автоматически, если студент выполняет все поставленные требования согласно требованиям рейтинговой системы.

Обобщая все представленные формы обучения и учитывая особенности коммуникативного взаимодействия между преподавателем и студентами, а также самими студентами, среди общих форм организации обучения мы считаем необходимым выделить фронтальные (коллективные), парные, индивидуальные и с переменным составом студентов [135].

Во время фронтального обучения все студенты, присутствующие на занятии, работают над одним и тем же заданием в едином темпе. В курсе «Программирование для начинающих» в начале изучения тем на лабораторных занятиях в основном используется именно эта форма организации обучения, поскольку уровень компетенции бакалавра прикладной информатики в области объектно-ориентированного программирования у студентов является относительно низким.

При этом, поскольку студенты, уже имеющие достаточные и высокие знания в области объектно-ориентированного программирования, имеют возможность выполнить задание лабораторных работ заранее (благодаря использованию Piazza, OpenClass), на лабораторных занятиях преподаватель может привлечь их к обучению слабых студентов. Реализация этой формы нашла свое отражение в использовании практических методов (выполнение лабораторных работ, практикумов, решение задач) в рамках дисциплины «Информатика и программирование». В процессе применения этих методов

студенты получают новые знания, умения и опыт деятельности, являющиеся основой целостной компетенции бакалавра прикладной информатики в области объектно-ориентированного программирования.

При обучении основам объектно-ориентированного программирования целесообразным также является использование парной формы организации обучения, при которой основное взаимодействие происходит между двумя студентами, которые работают вместе и могут обсуждать задачи, осуществлять взаимообучение или взаимоконтроль. Парное программирование заключается в том, что один студент работает над написанием кода, а другой сидит рядом и наблюдает за его работой, таким образом, контролируя его работу и представляя проект в целом. Использование данной формы обучения является предпочтительным на лабораторных занятиях, как в аудиторной работе, так и внеаудиторной.

Индивидуальная форма обучения предусматривает выполнение каждым студентом собственной задачи в личном темпе под руководством преподавателя. Особенностью лабораторных занятий в компьютерной аудитории является индивидуальный темп работы каждого студента, даже при изначально фронтальной форме организации занятия. В условиях компьютерного класса управлять индивидуальной деятельностью студентов достаточно сложно: ситуация с каждым компьютером практически уникальна. В этом случае целесообразным будет использование метода проблемного обучения.

Эффективной организации индивидуальной работы способствует также использование всех возможностей онлайн-платформ Piazza и OpenClass, позволяющих предлагать индивидуальные и тестовые задания, использовать чат и обмен файлами.

Групповая форма обучения используется во время работы над проектами, когда, в частности, студенты работают в группах. При объединении студентов в группы следует учитывать желание студентов, но руководство преподавателя необходимо. Групповая форма работы является

эффективной, если группы являются неоднородными как по уровню предметных компетенций, так и по познавательной активности. А использование средств обучения основам объектно-ориентированного программирования способствует оптимизации групповой работы.

Таким образом, нами описаны компоненты методики обучения основам объектно-ориентированного программирования. Схематично структура этих компонентов представлена на рисунке 7.

Рисунок 7. Компоненты методики обучения основам объектно-ориентированного программирования

2.2. Опытно-экспериментальная работа по оценке эффективности методики обучения основам объектно-ориентированного программирования бакалавров прикладной информатики

Целью организации опытно-экспериментальной работы является оценка эффективности методики обучения основам объектно-ориентированного программирования бакалавров прикладной информатики в процессе обучения студентов. Эксперимент проводился в ГБОУ ВО РК «Крымский инженерно-педагогический университет». Программа экспериментальной работы была разработана для изучения современного состояния проблемы обучения основам объектно-ориентированного программирования бакалавров прикладной информатики и уровня сформированности компетенции бакалавра прикладной информатики в области объектно-ориентированного программирования.

Согласно цели исследования, программа экспериментальной работы включала три взаимосвязанных этапа: констатирующий, поисковый и формирующий [9, 72].

Констатирующий этап опытно-экспериментальной работы (2007/2008;2008/2009 уч. гг.)

Первый этап опытно-экспериментальной работы – констатирующий. На данном этапе ставились такие цели:

– определить ориентировочный уровень знаний, умений и навыков студентов первого курса по программированию, необходимых для успешного освоения основ объектно-ориентированного программирования;

– уточнить содержание теоретического материала курса по программированию, исходя из целей обучения в высших учебных заведениях соответствующего направления подготовки;

– выявить существующие подходы к определению содержания лабораторного практикума по программированию и роль такого практикума в системе подготовки бакалавров прикладной информатики.

Констатирующий эксперимент проводится в начале исследования. В ходе констатирующего этапа педагогического эксперимента применялись пассивные методы исследования [59].

Студенты, участвовавшие в эксперименте, изучали курс программирования согласно учебной программе, соответствующей требованиям стандартов высшего образования. Никаких дополнительных задач, кроме запланированных учебной программой, перед студентами не ставились.

Основной массив данных констатирующего эксперимента был получен на основе анализа результатов промежуточной аттестации обучающихся первого курса направления «Прикладная информатика» по дисциплине «Информатика и программирования». Нами были проанализированы данные промежуточной аттестации обучающихся 1 курса 2007/2008, 2008/2009 учебных годов.

Задания для проверки уровня знаний приведены в приложении 11. Уровень подготовки обучающихся в области программирования определялся в соответствии с качественными показателями, представленными в таблице 13.

Таблица 13

Уровни подготовки обучающихся в области программирования

Уровень	**Качественная характеристика**
Высокий 81-100 баллов	Обучающийся свободно владеет основами теории и практики программирования и способен: – анализировать и специфицировать множество входных и выходных данных задачи; – выполнять функциональную декомпозицию задачи; – реализовывать поиск различных вариантов выполнения алгоритма; – оценивать эффективность алгоритма; – анализировать и моделировать предметную область задачи; – структурировать алгоритм и программное решение задачи; – выделить факторы, влияющие на состояние отдельного объекта и всей системы; – интегрировать разработанный алгоритм и программное решение с другими разработками; – объяснить алгоритм и программное решение другим студентам; – работать с абстрактными типами данных, самостоятельно разрабатывает алгоритмы и программно реализовывать их.

Уровень	Качественная характеристика
Средний 56-80 баллов	Обучающийся владеет основами теории и практики программирования и способен: – создавать программные продукты на изучаемом языке программирования на основе приобретенных знаний, понимания и умений; – анализировать и специфицировать множество входных и выходных данных задачи; – представить решение задачи через элементарные действия, включающие базовые алгоритмические структуры; – выполнить функциональную декомпозицию задачи; – анализировать различные варианты выполнения алгоритма; – разработать, отладить и тестировать программный код для типовых заданий на основе стандартных алгоритмов и имеющихся программных решений.
Достаточный 20-55 баллов	Обучающийся достаточно владеет основами теории и практики программирования и способен: – анализировать множество входных и выходных данных задачи; – представить решение задачи через элементарные действия, (базовые алгоритмические структуры); – разработать и устранить синтаксические ошибки в простой программе с базовыми алгоритмическими структурами и типами данных.
Низкий 0-19 баллов	Обучающийся недостаточно владеет основами теории и практики программирования и не способен: – представить решение задачи через базовые алгоритмические структуры; – разработать и устранить синтаксические ошибки в простой программе с базовыми алгоритмическими структурами и типами данных.

Результаты данной части констатирующего эксперимента представлены в таблице 14. Эти результаты показывают общий уровень подготовки студентов в области программирования на момент завершения изучения дисциплины «Информатика и программирование».

Таблица 14

Оценка уровня подготовки обучающихся в области программирования

Уч.год	Средний балл	Уровень знаний
2007/2008	35	Достаточный
2008/2009	34	Достаточный

Для получения данных этой таблицы средний балл показателя успеваемости студентов за соответствующий год обучения определялся как

среднее арифметическое суммы баллов всех опрошенных соответствующего года.

$$Q = \frac{\sum_{i=1}^{n} a_i}{n}$$

где Q – средний балл, a_i – балл одного опрошенного, n – общее количество опрошенных соответствующего года обучения.

Помимо оценки результатов промежуточной аттестации, на констатирующем этапе нами применялся и метод анкетирования, который позволил уточнить некоторые качественные характеристики исследуемой проблемы [76].

Анкета была разработана для студентов первого курса направления подготовки «Прикладная информатика» (Приложение 12). Результаты анкетирования приведены в таблице 15.

Таблица 15

Результаты анкетирования студентов первых курсов направления подготовки «Прикладная информатика»

Вопросы	Частота положительных ответов	Частота отрицательных ответов
1	0,134	0,866
2	0,585	0,415
3	0,110	0,890
4	0,244	0,756
5	0,439	0,561

Таким образом, в ходе констатирующего эксперимента было установлено:

1. Уровень подготовки обучающихся первого курса 2007/2008; 2008/2009 учебных годов в области программирования можно характеризовать лишь как достаточный, что подтверждается результатами промежуточной аттестации.

2. Большинство студентов бакалавриата прикладной информатики считает, что уровня их школьных знаний недостаточно для уверенного изучения программирования в вузе. При этом всего треть респондентов занималась дополнительно программированием в школе.

3. Студенты, принимавшие участие в анкетировании, в своем большинстве не демонстрируют достаточных знаний в области объектно-ориентированного программирования. Кроме того, респонденты показали пониженный уровень мотивации к изучению объектно-ориентированных языков программирования.

Полученные результаты позволили определить круг вопросов, имеющих значение для дальнейшего проведения опытно-экспериментальной работы. Большинство студентов демонстрируют ограниченность знаний о назначении и возможностях использования объектно-ориентированных языков программирования, не владеют навыками программирования с использованием объектно-ориентированного подхода, что указывает на недостаточный уровень сформированности их готовности к освоению основ объектно-ориентированного программирования.

Данные констатирующего эксперимента позволили обозначить следующие условия, определяющие не сформированность компетенции бакалавра прикладной информатики в области объектно-ориентированного программирования:

– фрагментарность знаний о возможностях использования объектно-ориентированного подхода для разработки программ различного назначения;

– отсутствие знаний о роли и месте объектно-ориентированной парадигмы в современной компьютерной науке.

Выявленные проблемы определили необходимость уточнения целей и содержания подготовки бакалавров прикладной информатики в области объектно-ориентированного программирования (что нами было сделано через описание компетенции бакалавров прикладной информатики в области

объектно-ориентированного программирования), а также адекватных этим целям и содержанию методов, средств и форм обучения.

Подобные результаты позволили также нам сделать вывод о том, что система подготовки бакалавра прикладной информатики должна предполагать наличие специального подготовительного курса, нацеленного на формирование системы знаний, умений, навыков и опыта деятельности, необходимых для успешного освоения курса программирования на профессиональном языке. Профессиональный курс в значительной степени должен ориентироваться на освоение объектно-ориентированного подхода при разработке компьютерных программ на изучаемом языке, а это означает, что основные понятия объектно-ориентированного подхода должны быть сформированы в рамках подготовительного курса. При этом, как подготовительный, так и профессиональный курс должны строиться на активной основе, предполагающей такую организацию учебного процесса, при которой повышается уровень активности студентов. Активное обучение на основе практических занятий на подготовительном этапе предполагает использование визуальных учебных сред, а на профессиональном – уже профессиональную среду.

Поисковый этап опытно-экспериментальной работы (2009/2010; 2010/2011; 2011/2012; 2012/2013 уч. гг.)

Второй этап опытно-экспериментальной работы – поисковый. На этом этапе осуществлялась апробация отдельных средств и методов разрабатываемой методики.

Основной акцент на этом этапе был сделан на проверке дидактических возможностей средств организации учебной деятельности студентов. С этой целью во время поискового этапа опытно-экспериментальной работы с 2009 г. по 2013 г. при обучении студентов 1 курса направления подготовки «Прикладная информатика» программированию использовались следующие средства организации учебной деятельности студентов:

1) педагогические программные средства – платформы Piazza, OpenClass;

2) инструментальные программные средства – инструменты для подготовки презентаций и портфолио Prezi и Power Point.

Эти инструменты описаны нами в п. 2.1 диссертационного исследования.

Так, использование онлайн-платформ Piazza и OpenClass при изучении курса «Информатика и программирование» для организации взаимодействия преподавателя со студентами предполагалось в 2009-2010 учебном году. При этом студентам при обучении программированию на платформах Piazza и OpenClass предлагались задачи для практической работы, выполнение которых сопровождалось подсказками на доске обсуждений в OpenClass.

В 2010-2011 учебном году в учебный процесс были внедрены инструментальные программные средства, предназначенные для подготовки презентационных материалов средствами Prezi, PowerPoint.

Данная работа была усилена в 2011-2012 и 2012-2013 учебных годах, когда при обучении программированию на платформах Piazza и OpenClass предлагались не только задачи для практической работы, но и вопросы, тестовые задания для самоконтроля, задания для промежуточного контроля знаний по программированию.

Для того чтобы оценить эффективность внедрения отдельных компонент разрабатываемой методики нами был продолжен анализ результатов промежуточной аттестации студентов первого курса по дисциплине «Информатика и программирование» по тем оценочным средствам и методике расчёта баллов, что и на констатирующем этапе проводимого эксперимента.

Результаты этого анализа, сопоставленные с перечнем внедряемых элементов методики, приводятся в таблице 16. Наглядно полученные данные представлены на рисунке 8.

Как видно из графика на рисунке 8, внедрение отдельных средств методики обучения основам объектно-ориентированного программирования бакалавров прикладной информатики на поисковом этапе опытно-экспериментальной работы, позволило повысить уровень подготовки студентов в области программирования (вырос средний балл показателя успеваемости студентов за соответствующий год обучения).

Таблица 16

Компоненты методики и уровень подготовки студентов в области программирования на поисковом этапе

Год	Средний балл	Уровень знаний	Педагогические программные средства	Инструментальные программные средства
2009/2010	45	Достаточный	Piazza, OpenClass	
2010/2011	53	Достаточный	Piazza, OpenClass	Prezi, Power Point
2011/2012	62	Достаточный	Piazza, OpenClass	Prezi, Power Point
2012/2013	68	Средний	Piazza, OpenClass	Prezi, Power Point

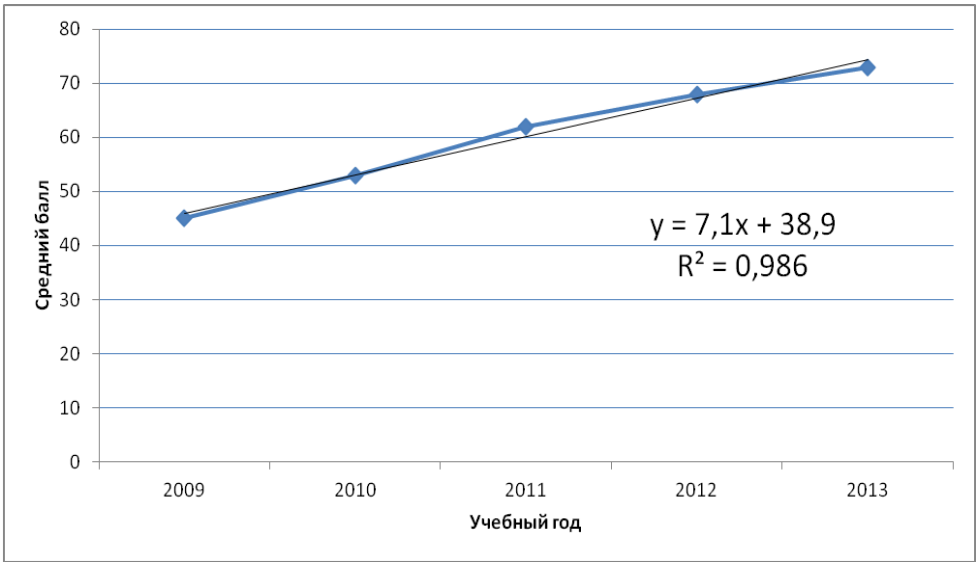


Рисунок 8. Показатели уровня знаний студентов в области программирования на поисковом этапе в 2009-13 г.г.

Полученные средствами MS Excel линия тренда, регрессионная модель и величина достоверности аппроксимации R^2, представленные на рисунке 8,

позволяют оценить тенденцию роста уровня знания студентов по программированию.

Обратим внимание, что величина достоверности аппроксимации (число от 0 до 1, отображающее степень соответствия ожидаемых значений для линии тренда фактическим данным) равна 0,9869, что свидетельствует о том, что на 98,69% расчетные параметры модели, то есть сама модель, объясняют зависимость и изменения изучаемого параметра – Y (уровень знаний – средний балл по программированию) от исследуемого фактора – X (использование компонентов методики обучения). Коэффициент детерминации является в некотором роде показателем качества модели: чем он выше, тем качественнее регрессионная модель. R^2 не может быть больше 1, когда R^2 выше 0,8 – регрессионная модель резонна, аппроксимация удовлетворительна.

Определить, есть ли взаимосвязь между показателями интегрированного коэффициента и уровнем знаний в области программирования у студентов 1 курса направления подготовки «Прикладная информатика», позволяет корреляционный анализ. Пакет анализа Excel позволяет рассчитать корреляционную матрицу (табл. 17).

Таблица 17

Корреляционная матрица

	Столбец 1	Столбец 2
Столбец 1	1	
Столбец 2	0,943752	1

Словесное описание величин коэффициента корреляции представлено в таблице 18.

Таблица 18

Словесное описание величин коэффициента корреляции

Значение коэффициента корреляции r	**Интерпретация**
0 < r <= 0,2	Очень слабая корреляция
0,2 < r <= 0,5	Слабая корреляция

0,5 < r <= 0,7	Средняя корреляция
0,7 < r <= 0,9	Сильная корреляция
0,9 < r <= 1	Очень сильная корреляция

По результатам расчета видно, что коэффициент корреляции равен 0,943752. Исходя из вышеприведенной таблицы, можно сделать следующие заключения: между показателями использования отдельных компонентов методики обучения и уровнем знаний в области программирования у студентов 1 курса направления подготовки «Прикладная информатика» существует очень сильная корреляция (заключение о силе зависимости), переменные коррелируют положительно (заключение о направлении зависимости).

В таблице «Дисперсионный анализ» (табл. 19) оценивается достоверность полученной модели, которая описывается регрессионным уравнением y = 7,1x + 38,9 и величиной достоверности аппроксимации R^2 = 0,9869 (представленной на рисунке 3.2) по уровню значимости критерия Фишера (строка Регрессия, столбец Значимость F) – она равна 0,000041, то есть p < 0,05 и модель значима).

Таблица 19

Дисперсионный анализ

	df	**SS**	**MS**	**F**	**Значимость F**
Регрессия	1	5,303	5,303	65,171	0,000041
Остаток	8	0,651	0,081		
Итого	9	5,954			

Результаты анализа показывают, что разница в уровне знаний в области программирования у студентов 1 курса направления подготовки «Прикладная информатика» в зависимости от интенсивности использования отдельных компонентов методики обучения, является значимой (существенной). Об этом свидетельствует значение F: 65,171 > 0,000041.

Следовательно, использование педагогических программных средств – платформы Piazza, OpenClass и инструментальных программных средств –

инструменты для подготовки презентаций и портофолио Prezi и PowerPoint при изучении программирования, влияет на уровень знаний в области программирования у студентов 1 курса направления подготовки «Прикладная информатика».

Таким образом, на этапе поискового эксперимента в результате использования отдельных компонентов методики обучения уровень знаний в области программирования у студентов 1 курса направления подготовки «Прикладная информатика» изменился с достаточного до среднего, а средний балл вырос с 45 до 73. Это подтверждает целесообразность и эффективность внедрения методики обучения в процесс подготовки бакалавров прикладной информатики.

Формирующий этап педагогического эксперимента (2013/2014; 2014/2015; 2015/2016 уч. гг.)

Третий этап проведенного эксперимента – формирующий. Целью формирующего этапа эксперимента была оценка эффективности разработанной методики обучения основам объектно-ориентированного программирования бакалавров прикладной информатики с использованием визуальных учебных сред. Для этого на базе кафедры прикладной информатики ГБОУ ВО РК «КИПУ» было организовано обучение контрольной и экспериментальной групп студентов. Обучение студентов в контрольной группе происходило по традиционной схеме, а в экспериментальной группе – согласно предлагаемой методике обучения основам объектно-ориентированного программирования бакалавров прикладной информатики с использованием визуальных учебных сред (в учебный процесс была введена дисциплина «Программирование для начинающих» в рамках которой студенты работали с визуальными учебными средами Alice и Scratch). Такой подход к организации формирующего эксперимента позволил объективно выявить количественные и качественные изменения в уровнях сформированности компетенции бакалавра прикладной

информатики в области объектно-ориентированного программирования по критериям и показателям сформированности этой компетенции.

Опишем содержание фактически проделанной нами работы и полученные результаты.

Методика проведения формирующего эксперимента предполагала следующее:

1. Разделение студентов на экспериментальную (ЭГ) и контрольную (КГ) группы (устанавливается совпадение характеристик экспериментальной и контрольной групп до начала эксперимента.).

2. Внедрение для экспериментальной группы разработанной методики обучения основам объектно-ориентированного программирования в учебный процесс (реализуется педагогическое воздействие на экспериментальную группу).

3. Проведение количественного и качественного анализа полученных результатов, их систематизацию и обобщение (устанавливается различие характеристик экспериментальной и контрольной групп после окончания эксперимента, устанавливается динамика изменений экспериментальной группы; проверяются аналогичные изменения в контрольной группе).

Схематично структура формирующего эксперимента представлена на рисунке 9.

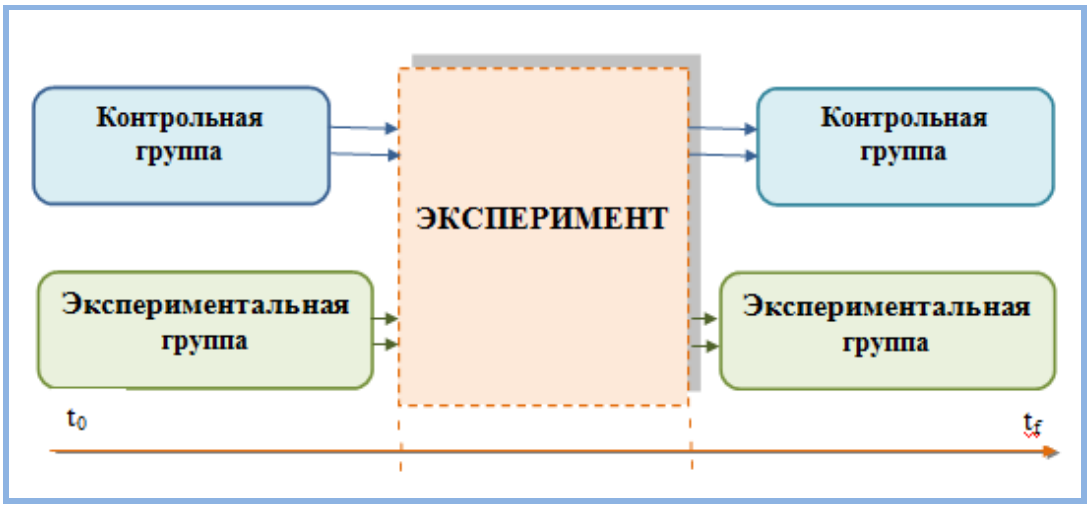


Рисунок 9. Структура формирующего эксперимента

В состав экспериментальной группы вошли студенты факультета экономики, менеджмента и информационных технологий ГБОУ ВО РК «КИПУ», обучающиеся по направлению «Прикладная информатика», в количестве 131 человек. В состав контрольной группы были включены 124 студента.

Для эксперимента подбирались академические группы, совпадающие по статистическим характеристикам.

Статистические характеристики определялись показателями средней оценки, полученной студентами первого курса по результатам контрольного среза знаний (приложение 13).

Для проверки однородности, предполагающей совпадение характеристик двух групп исследования, и достоверности эксперимента были использованы два критерия – Крамера-Уэлча [56] и Вилкоксона-Манна-Уитни [79] по данным таблицы 20.

Таблица 20

Успеваемость участников эксперимента, полученная студентами первого курса по результатам контрольного среза знаний

Оценка	Экспериментальная группа		Контрольная группа	
	Кол. чел.	%	Кол. чел.	%
Удовлетворительно	73	55,73	66	53,23
Хорошо	51	38,93	46	37,10
Отлично	7	5,34	12	9,68
Итого:	131	100,0	124	100,0

В исследовании использованы инструменты анализа данных для проверки равенства студентов КГ и ЭГ по уровням сформированности компетенции бакалавра прикладной информатики в области объектно-ориентированного программирования. На констатирующем этапе эксперимента получены следующие значения:

1) эмпирическое значение критерия Крамера-Уэлча, $T_{эмп}$ = 0,8597, критическое значение $T_{кр}$ = 1,96. Следовательно, характеристики сравниваемых выборок совпадают на уровне значимости 0,05 (99,95 %);

2) эмпирическое значение критерия Вилкоксона-Манна-Уитни, $W_{эмп} = 0,5912$, критическое $W_{кр} = 1,96$. Следовательно, характеристики сравниваемых выборок совпадают на уровне значимости 0,05 (99,95 %).

Таким образом, начальный (до начала эксперимента) уровень сформированности компетенции бакалавра прикладной информатики в области объектно-ориентированного программирования в ЭГ и КГ совпадают, следовательно, все участники эксперимента имеют потенциально одинаковые возможности для апробации предлагаемой методики обучения основам объектно-ориентированного программирования бакалавров прикладной информатики с использованием визуальных учебных сред.

Экспериментальная группа проходила обучение по предложенной методике обучения основам объектно-ориентированного программирования бакалавров прикладной информатики с использованием визуальных учебных сред, которая включала (рис. 10):

1) Преподавание дисциплины «Программирование для начинающих».

Начиная с 2013 года на занятиях по дисциплине «Программирование для начинающих» бакалавры прикладной информатики работают с визуальными средами программирования. Дисциплина «Программирование для начинающих» направлена на изучение основ объектно-ориентированного программирования, которое базируется в основном на моделировании некоторого реального процесса – объекта. Поэтому в рамках данной дисциплины обучение начиналось с таких концептуальных вопросов, как описание и идентификация объектов, классы, методы. Изучение основ объектно-ориентированного программирования осуществлялось посредством визуальных учебных сред Alice и Scratch, где объекты и их взаимодействия всегда представлены в центре внимания. Моделирование объектов и их взаимосвязей, реализовано включением элементов проектирования объектов с самого начала курса, а представление объектов в форме программного кода интегрировано в курс соответственно.

2) Использование визуальных учебных сред Alice, Scratch, предназначенных для обучения основам объектно-ориентированного программирования, формирование компетенции бакалавра прикладной информатики в области объектно-ориентированного программирования.

3) Использование педагогических программных средств, предназначенных для организации и компьютерной поддержки учебно-познавательной деятельности.

Параллельно с визуальными средами программирования при изучении дисциплин «Программирование для начинающих» и «Информатика и программирование» используются педагогические программные средства – платформы Piazza, OpenClass.

4) Использование инструментальных программных средств.

С помощью инструментальных программных средств – Prezi, PowerPoint осуществляется подготовка выполненных студентами проектов или генерирование учебно-методических и организационных материалов.

Обобщение фактического статистического материала осуществлялось в соответствии с основными требованиями репрезентативности и валидности проведения выборки педагогического эксперимента. При этом на этапе формирующего эксперимента мы использовали трехуровневую характеристику сформированности компетенции бакалавра прикладной информатики в области объектно-ориентированного программирования (достаточный, средний и высокий уровни), описанную нами в п. 1.2 (таблица 5). Обработка полученного статистического материала дала возможность проверить выдвинутую гипотезу исследования.

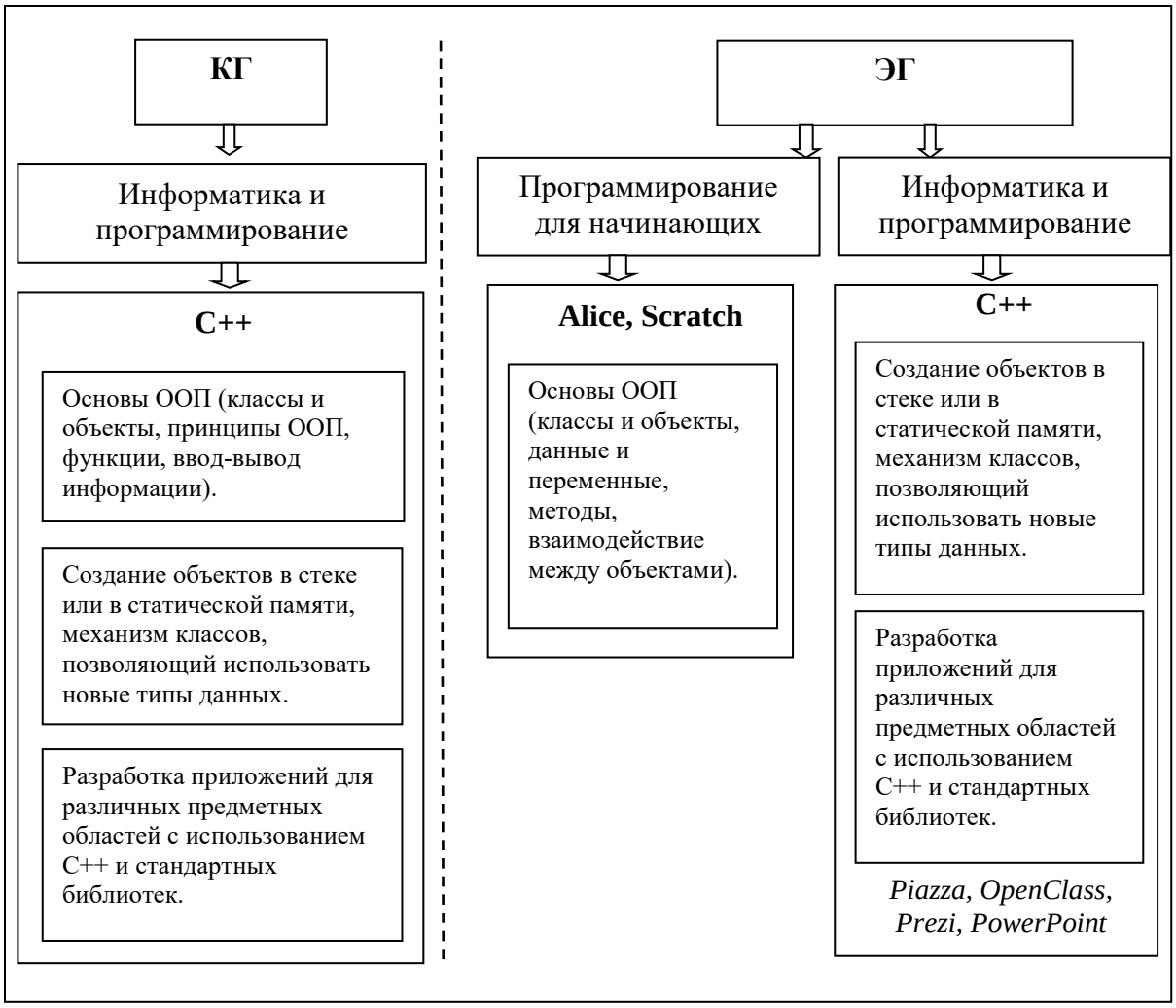


Рис. 10. Схема формирующего этапа

Анализ результатов исследования по формированию компетенции бакалавра прикладной информатики в области объектно-ориентированного программирования

Для оценивания результатов исследования количественные результаты эмпирически наблюдаемых показателей, соответствующих определенному уровню сформированности компетенции бакалавра прикладной информатики в области объектно-ориентированного программирования, переводились в количественные эквиваленты. Были введены следующие количественные эквиваленты:

- 3 балла – высокий уровень сформированности компетенции бакалавра прикладной информатики в области объектно-ориентированного программирования
- 2 балла – средний уровень сформированности компетенции бакалавра прикладной информатики в области объектно-ориентированного программирования
- 1 балл – слабая выраженность показателя, интерпретировалось как показатель наличия достаточного уровня сформированности компетенции бакалавра прикладной информатики в области объектно-ориентированного программирования

Общесредний уровень сформированности каждой компоненты компетенции бакалавра прикладной информатики в области объектно-ориентированного программирования будем вычислять следующим образом:

*(Д*1 балл + С*2 балла + В*3 балла)/*n,

где

Д – это количество студентов с достаточным уровнем сформированности компонента компетенции,

С – это количество студентов со средним уровнем сформированности компонента компетенции,

В – это количество студентов с высоким уровнем сформированности компонента компетенции,

n – это общее количество студентов в группе.

Для выявления общесреднего уровня сформированности компетенции бакалавра прикладной информатики в области объектно-ориентированного программирования изначально были обозначены границы баллов для каждого уровня (табл. 21). Так:

- *достаточный уровень* сформированности компетенции бакалавра прикладной информатики в области объектно-ориентированного

программирования характеризуется числовыми показателями в пределах 1–1,69;

- *средний уровень* сформированности компетенции бакалавра прикладной информатики в области объектно-ориентированного программирования характеризуется числовыми показателями в пределах 1,7–2,39;
- *высокий уровень* сформированности компетенции бакалавра прикладной информатики в области объектно-ориентированного программирования характеризуется числовыми показателями в пределах 2,4–3.

Таблица 21

Шкала диагностики уровней сформированности компетенции бакалавра прикладной информатики в области объектно-ориентированного программирования

Назв. / Баллы	Уровни сформированности компетенции бакалавра прикладной информатики в области объектно-ориентированного программирования		
	Достаточный	Средний	Высокий
Значения	1–1,69	1,7–2,39	2,4–3

Рассмотрим покритериально результаты анализа данных исследования на констатирующем и формирующем этапах педагогического эксперимента.

Диагностика сформированности мотивационно-ценностного компонента компетенции бакалавра прикладной информатики в области объектно-ориентированного программирования.

От сформированности мотивации и потребности к изучению теоретических основ объектно-ориентированной технологии зависят активность познавательной деятельности студентов, а также успешность освоения ими основ объектно-ориентированного программирования.

Статистический материал для диагностики уровня сформированности мотивационно-ценностного компонента сформированности компетенции

бакалавра прикладной информатики в области объектно-ориентированного программирования получен в результате анкетирования. Студентам было предложено дать ответ на вопрос: «Какие из предложенных мотивов могут побудить Вас использовать объектно-ориентированную парадигму программирования в будущей профессиональной деятельности?» (Приложение 14).

Каждый вопрос анкеты оценивался по трехбалльной шкале с учетом степени значимости мотивов, которые могут побудить анкетируемого изучать и использовать объектно-ориентированные технологии в будущей профессиональной деятельности (3 балла – максимальная степень значимости мотива, 2 балла – средняя степень значимости, 1 балл – минимальная степень значимости мотива). На основании проведенного исследования по диагностике мотивационной сферы, студенты были распределены на уровневые группы: достаточный, средний, высокий. Рассмотрим динамику изменений мотивационно-ценностного компонента до и после формирующего этапа эксперимента (табл. 22).

Таблица 22

Результаты уровня сформированности мотивационно-ценностного компонента компетенции бакалавра прикладной информатики в области объектно-ориентированного программирования

Уровень сформированности	Формирующий этап			
	ЭГ		КГ	
	Кол. чел.	%	Кол. чел.	%
Достаточный	43	33,14	37	30,06
Средний	56	42,12	65	52,26
Высокий	32	24,74	22	17,68
Итого:	131	100	124	100
Ср. знач., балл	1,92	1,92	1,88	1,88

На формирующем этапе эксперимента большинство студентов обладают устойчивыми мотивами и интересом к изучению объектно-ориентированного программирования. Это подтверждают результаты диагностики уровня сформированности компетенции бакалавра прикладной

информатики в области объектно-ориентированного программирования по мотивационно-ценностному критерию: в ЭГ 33,14% студентов, а в КГ – 30,06% показали достаточный уровень сформированности мотивационно-ценностного компонента рассматриваемой компетенции. При этом высокий уровень мотивации оказался большим в ЭГ почти на 7%.

Средние значения уровня сформированности мотивационно-ценностного компонента компетенции бакалавра прикладной информатики в области объектно-ориентированного программирования на формирующем этапе педагогического эксперимента представлены на рисунке 11.

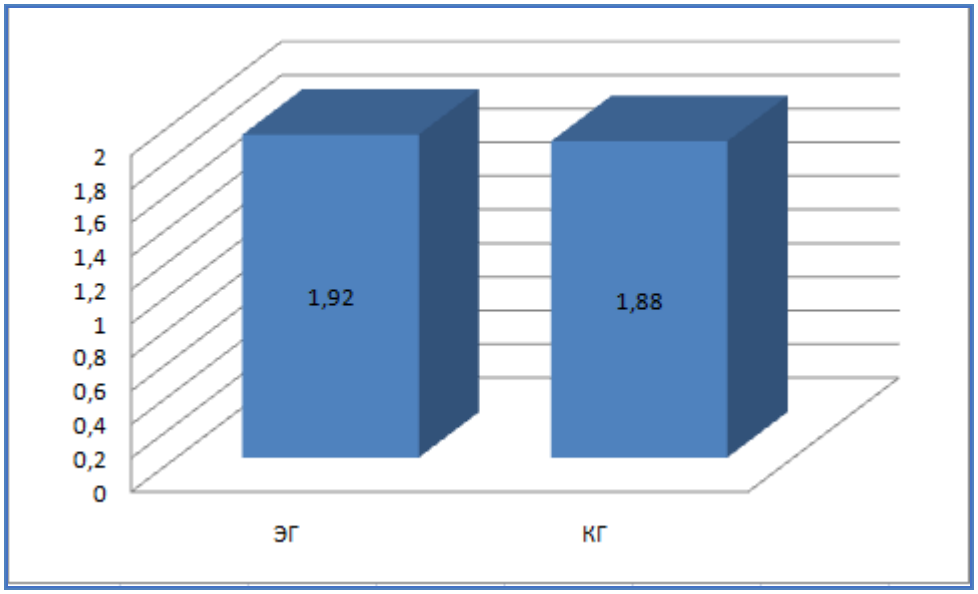


Рисунок 11. Средние значения уровня сформированности мотивационно-ценностного компонента компетенции бакалавра прикладной информатики в области объектно-ориентированного программирования (балл)

В результате проведенной экспериментальной работы по формированию мотивационно-ценностного компонента компетенции бакалавра прикладной информатики в области объектно-ориентированного программирования увеличение среднего значения в экспериментальной

группе по сравнению с контрольной составило 0,04 (1,92 и 1,88 баллов соответственно).

Для получения качественных характеристик мы также проводили анализ решения задач:

1. Используя документ Паспорт гражданина России, произвести его описание с помощью декомпозиции на простейшие объекты-атрибуты.

2. Рассмотрите объект Корабль из игры «Морской бой». Определите этот объект в терминах объектов-атрибутов.

Проведенный анализ показал, что у большинства студентов ЭГ высокий уровень сформированности компетенции бакалавра прикладной информатики в области объектно-ориентированного программирования по мотивационно-ценностному компоненту. А именно студентам присуща четкая мотивационная позиция в необходимости изучения объектно-ориентированного программирования для повышения профессионального уровня; приобретения ценностных ориентаций, мотивов адекватных целям и задачам деятельности.

Диагностика сформированности организационно-содержательного компонента компетенции бакалавра прикладной информатики в области объектно-ориентированного программирования.

С целью диагностики организационно-содержательного компонента компетенции бакалавра прикладной информатики в области объектно-ориентированного программирования в качестве показателя рассматривалось владение студентами знаниями основных понятий теории алгоритмизации и программирования.

Для определения уровня знаний в области алгоритмизации и программирования, для диагностики организационно-содержательного компонента компетенции бакалавра прикладной информатики в области объектно-ориентированного программирования использовался **специально разработанный тест**, включающий 15 вопросов и заданий различного типа сложности (Приложение 15).

1) общие информационные знания в области алгоритмизации и программирования;

2) специальные методологические и инженерные знания в области программирования, алгоритмов и структур данных.

Респондент получал за каждый правильный ответ по 1 баллу.

Результат тестирования в пределах 1-4 баллов интерпретировался как показатель наличия низкого уровня сформированности организационно-содержательного компонента компетенции бакалавра прикладной информатики в области объектно-ориентированного программирования; 5-11 баллов – среднего уровня; 12-15 баллов – высокого уровня.

Рассмотрим динамику изменений организационно-содержательного компонента компетенции бакалавра прикладной информатики в области объектно-ориентированного программирования на формирующем этапе эксперимента в ЭГ и КГ (табл. 22). Изменения количественных показателей организационно-содержательного компонента компетенции бакалавра прикладной информатики в области объектно-ориентированного программирования показали, что изменилась их динамика в сторону высоких оценок за счет снижения числа студентов, продемонстрировавших достаточный уровень знаний.

Результаты диагностики на формирующем этапе эксперимента указывают на то, что 56,14 % студентов ЭГ владеют достаточной системой знаний, относящихся к различным аспектам алгоритмизации и программирования, знают основные алгоритмы, структуры данных. В КГ этот показатель составляет 39,66% (табл. 23).

Таблица 23

Результаты уровня сформированности организационно-содержательного компонента компетенции бакалавра прикладной информатики в области объектно-ориентированного программирования

Уровень сформированности	Формирующий этап			
	ЭГ		КГ	
	Кол. чел.	%	Кол. чел.	%
Достаточный	39	29,82	66	53,44
Средний	74	56,14	49	39,66
Высокий	18	14,04	9	6,9
Итого:	131	100	124	100
Ср. знач., балл	1,84	1,84	1,53	1,53

Графическое представление результатов диагностики сформированности организационно-содержательного компонента компетенции бакалавра прикладной информатики в области объектно-ориентированного программирования (рис. 12), демонстрирует приращение среднего значения в экспериментальной группе. Этот сдвиг в ЭГ по сравнению с КГ составляет 0,31 балл.

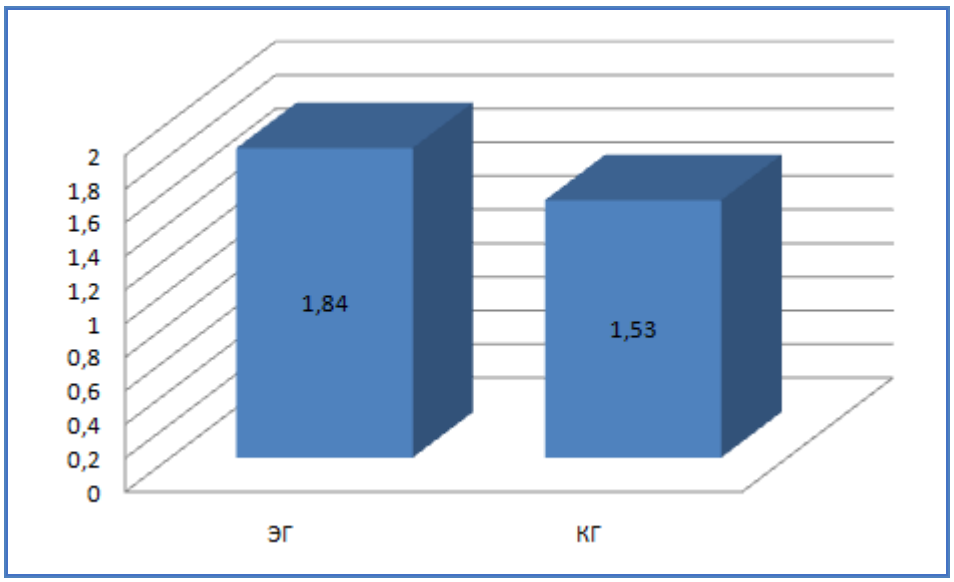


Рисунок 12. Средние значения уровня сформированности организационно-содержательного компонента компетенции бакалавра прикладной информатики в области объектно-ориентированного программирования (балл)

Сформированность организационно-содержательного компонента компетенции бакалавра прикладной информатики в области объектно-ориентированного программирования позволяет сделать вывод о том, что студенты владеют теоретическими знаниями в области методологии, базовых концепций, идей объектно-ориентированного программирования и проявляют познавательную активность, необходимых для изучения объектно-ориентированного программирования в дальнейшем.

На основании анализа выполненных проектов студентами мы получили и качественные характеристики. На данном этапе проектные работы студентов были связаны с выполнением таких заданий, как:

1. Создайте объектную модель информационной системы «Университет».
2. Создайте объектную модель информационной системы «Библиотека».
3. Создайте объектную модель информационной системы «Абитуриент».

Проведенный анализ выполнения этих заданий позволил определить уровень сформированности организационно-содержательного компонента бакалавра прикладной информатики в области объектно-ориентированного программирования. Этот уровень оказался средним – студенты ЭГ показали знания технологии программирования объектно-ориентированной парадигмы, а также некоторые знания и умения конкретных способов их реализации, в отличие от КГ.

Диагностика сформированности когнитивно-операционного компонента компетенции бакалавра прикладной информатики в области объектно-ориентированного программирования.

С целью диагностики когнитивно-операционного компонента компетенции бакалавра прикладной информатики в области объектно-ориентированного программирования в качестве показателя изучалась

степень освоения теории объектно-ориентированного программирования и способности использования этих знаний в профессиональной деятельности.

Количественный и качественный анализ диагностики сформированности когнитивно-операционного компонента у студентов контрольной и экспериментальной групп был проведен на основе результатов теста, включающего общие вопросы по теории ООП (Приложение 16).

Респондент получал за каждый правильный ответ по 1 баллу.

Результат тестирования в пределах 1-4 баллов интерпретировался как показатель наличия низкого уровня сформированности организационно-содержательного компонента компетенции бакалавра прикладной информатики в области объектно-ориентированного программирования; 5-11 баллов – среднего уровня; 12-15 баллов – высокого уровня.

Обобщенные данные такой оценки представлены в таблице 24 и на рисунке 13.

Таблица 24

Обобщенные результаты сформированности когнитивно-операционного компонента у студентов контрольной и экспериментальной групп

Уровень сформированности	Формирующий этап			
	ЭГ		КГ	
	Кол. чел.	%	Кол. чел.	%
Достаточный	41	31,30	71	54,20
Средний	70	53,44	48	36,64
Высокий	20	15,27	5	3,82
Итого:	131	100	124	100
Ср. знач., балл	1,84	1,84	1,47	1,47

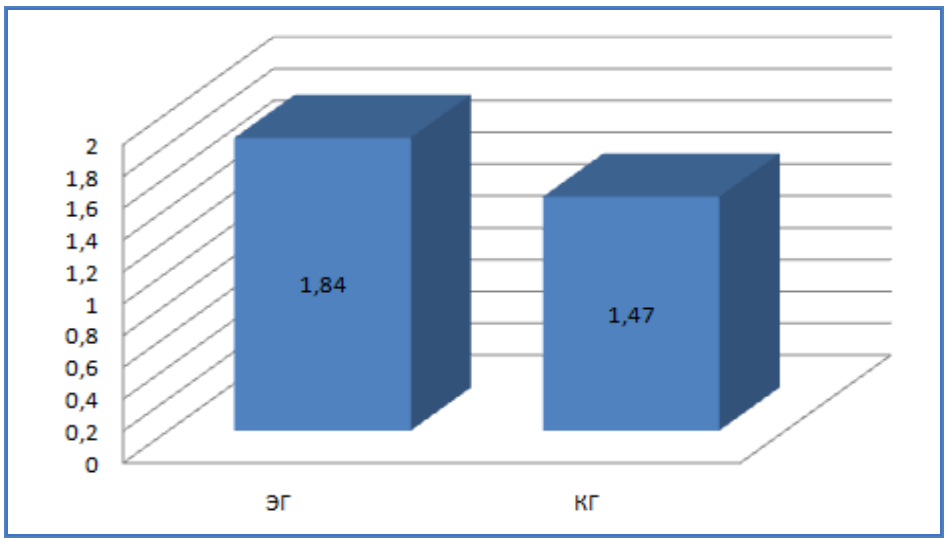


Рисунок 13. Средние значения уровня сформированности когнитивно-операционного компонента компетенции бакалавра прикладной информатики в области объектно-ориентированного программирования (балл)

Результаты диагностики на формирующем этапе эксперимента указывают на то, что студенты в экспериментальной группе имеют более высокую степень освоения теории объектно-ориентированного программирования, нежели в контрольной группе.

Для более точной оценки качественных изменений данного компонента, наряду с тестированием, нами также анализировались результаты проектов, выполненных студентами в рамках дисциплины «Информатика и программирование». Результаты анализировались на основе трех составляющих: знаний (ответы на теоретические вопросы), умений и навыков (выполнение проекта). Выполнение проекта на данном этапе является заключительным в ходе обучения основам объектно-ориентированного программирования.

Ниже приведен пример такого задания, которое было введено для того, чтобы студент мог еще глубже осмыслить суть объектно-ориентированного программирования:

1. Создайте класс для обработки записей базы данных «Библиотека». Разработайте программу, соблюдая следующие требования:

1) Разместите интерфейс класса в заголовочный файл, а определения функций и главную функцию программы – в двух отдельных файлах.

2) Предусмотрите возможность работы с произвольным числом записей, а также реализуйте отдельными функциями класса:

– конструкторы без параметров и с параметрами;

– добавление объектов;

– удаление объектов;

– вывод информации на экран;

– поиск определенной информации по конкретному признаку;

– редактирование записей;

– сортировка по различным полям.

3) При разработке программы осуществите защиту данных (описание модификатором *private*) для изоляции элементов-данных класса от подпрограмм, в которых этот класс используется.

4) Программа должна содержать меню для проверки всех методов класса. Для реализации меню разработайте отдельную функцию, которая возвращает номер выбранного пункта меню.

Проведенный анализ результатов выполнения этого задания позволил сделать выводы о том, что у студентов ЭГ выражено стремление к углубленному программированию в области объектно-ориентированной парадигмы, а также умение практически использовать полученные знания в условиях современного программирования.

Диагностика сформированности личностно-рефлексивного компонента компетенции бакалавра прикладной информатики в области объектно-ориентированного программирования

Для диагностики уровня сформированности личностно-рефлексивного компонента компетенции бакалавра прикладной информатики в области объектно-ориентированного программирования использовалась методика диагностики способности к саморазвитию и самообразованию, предложенная

В.И. Андреевым [158, с. 57]. Самодиагностика студентами проводилась с помощью тестирования (Приложение 17).

Диагностика личностно-рефлексивного компонента компетенции бакалавра прикладной информатики в области объектно-ориентированного программирования позволяет оценить уровень сформированности навыков самосовершенствования и самореализации.

На основании проведенного исследования по диагностике уровня сформированности личностно-рефлексивного компонента компетенции бакалавра прикладной информатики в области объектно-ориентированного программирования, студенты были распределены на уровневые группы: достаточный, средний, высокий (табл. 24).

Обобщенные результаты (табл. 25, рис. 14) уровней личностно-рефлексивного компонента компетенции бакалавра прикладной информатики в области объектно-ориентированного программирования на формирующем этапе эксперимента указывают на динамику количественных значений.

Так, количество студентов с высоким уровнем сформированности личностно-рефлексивного компонента *компетенции бакалавра прикладной информатики в области объектно-ориентированного программирования* в ЭГ больше на 14,89%, чем в КГ.

Таблица 25

Результаты уровня сформированности
личностно-рефлексивного компонента компетенции бакалавра прикладной информатики в области объектно-ориентированного программирования

Уровень	Формирующий этап			
	ЭГ		КГ	
	Кол. чел.	%	Кол. чел.	%
Достаточный	34	26,23	61	49,17
Средний	65	49,36	51	41,31
Высокий	32	24,41	12	9,52
Итого:	131	100	124	100
Ср. знач., балл	1,98	1,98	1,60	1,60

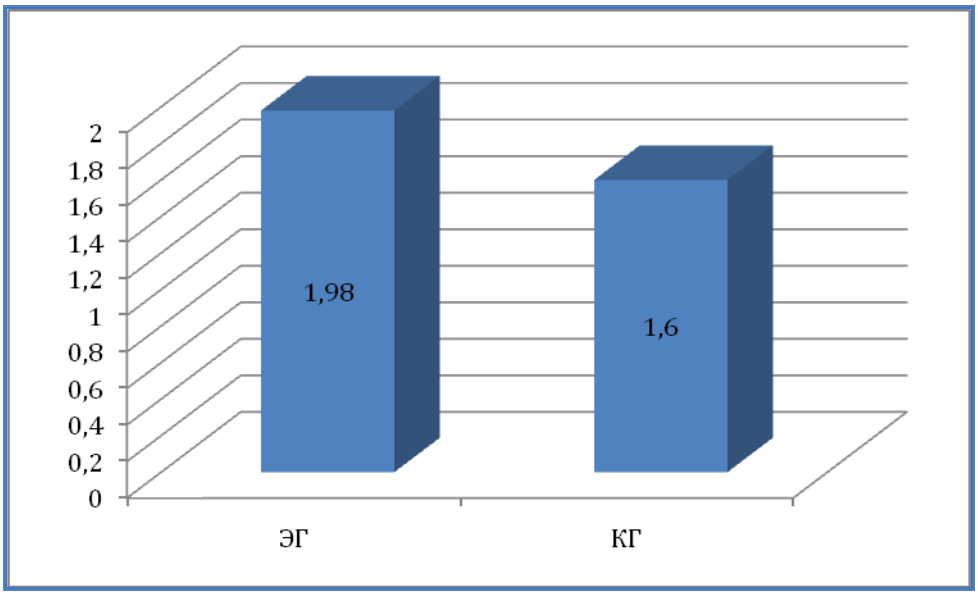


Рисунок 14. Средние значения уровня сформированности личностно-рефлексивного компонента компетенции бакалавра прикладной информатики в области объектно-ориентированного программирования (балл)

Для определения качественных характеристик уровня сформированности личностно-рефлексивного компонента компетенции бакалавра прикладной информатики в области объектно-ориентированного программирования был поведен анализ задач, выполненных студентами:

1. Написать программу, которая использует класс Book и выполняет следующие действия:

– вводит с клавиатуры массив данных SHOP, состоящий из N переменных типа BOOK;

– выводит на экран все книги, которые были напечатаны в заданном году.

В результате анализа, мы выявили, что студенты ЭГ имеют способность разносторонне подходить к анализу ситуаций при решении объектно-ориентированных задач в зависимости от целей и условий.

Анализ динамики сформированности компонентов компетенции бакалавра прикладной информатики в области объектно-ориентированного

программирования в экспериментальной и контрольной группах, а также сопоставление результатов в конце педагогического эксперимента позволили свести полученные ранее данные (табл. 21, 22, 23, 24) в обобщенную таблицу 26.

Таблица 26

Уровни сформированности компетенции бакалавра прикладной информатики в области объектно-ориентированного программирования на формирующем этапе

Компоненты компетенции	Экспериментальная группа				Контрольная группа			
	Уровни (%)			Среднее Значение, балл	Уровни (%)			Среднее Значение, балл
	достаточный	средний	высокий		достаточный	средний	высокий	
Мотивационно-ценностный	33,14	42,12	24,74	1,92	30,06	52,26	17,68	1,88
Организационно-содержательный	29,82	56,14	14,04	1,84	53,44	39,66	6,90	1,53
Когнитивно-операционный	31,30	53,44	15,27	1,84	54,20	36,64	3,82	1,47
Личностно-рефлексивный	26,23	49,36	24,41	1,98	49,17	41,31	9,52	1,60

На рисунке 15 показаны результаты (средние значения в баллах) сформированности компетенции бакалавра прикладной информатики в области объектно-ориентированного программирования всем компонентам в конце формирующего этапа эксперимента (табл. 24).

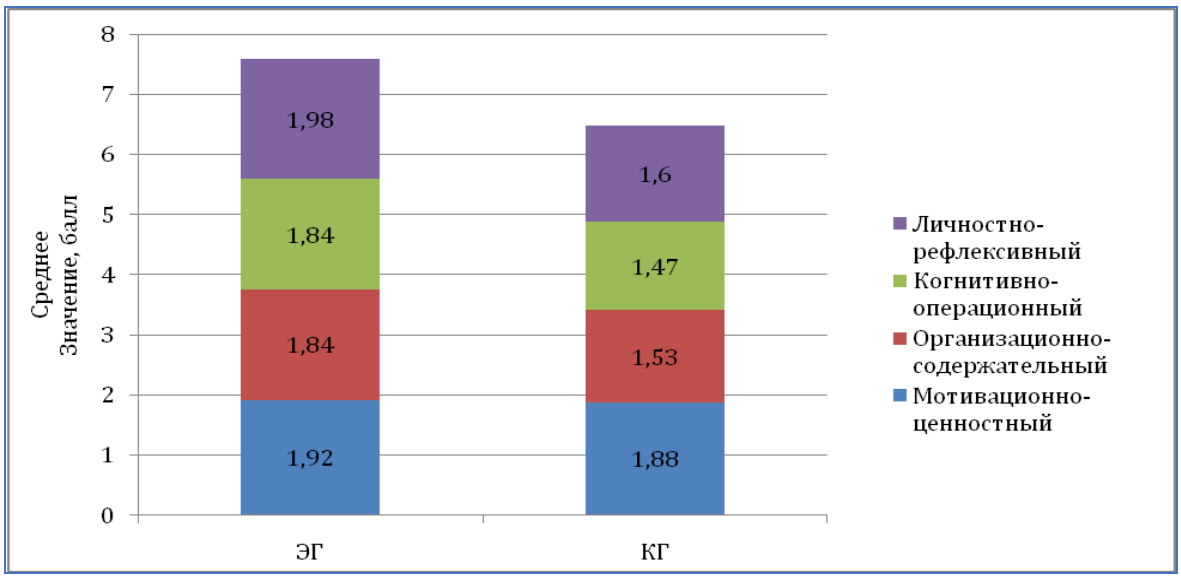


Рисунок 15. Сформированность компетенции бакалавра прикладной информатики в области объектно-ориентированного программирования на формирующем этапе по всем компонентам, (балл)

Анализ результатов исследования, приведенных в таблице 25, показывает, что после проведения работы по формированию компетенции бакалавра прикладной информатики в области объектно-ориентированного программирования в ЭГ наблюдались заметные сдвиги в уровнях ее сформированности, зато в КГ они незначительны по сравнению с ЭГ. Так, произошли изменения в ЭГ по мотивационно-ценностному критерию, возросла заинтересованность в приобретении новых знаний и умений, ярко выразились потребности и интерес к изучению объектно-ориентированного программирования. На это указывает большее число студентов с высоким уровнем сформированности компетенции бакалавра прикладной информатики в области объектно-ориентированного программирования по мотивационно-ценностному компоненту.

Изменения в уровнях сформированности организационно-содержательного и когнитивно-операционного компонентов показали, что повысилась значимость показателей в сторону высокого уровня за счет уменьшения числа студентов с низким уровнем рассматриваемых компонентов. Количество студентов в ЭГ высоком уровне сформированности

организационно-содержательного и когнитивно-операционного компонентов больше, чем в КГ на 7,14% и 11,45% соответственно. В целом можно отметить, что знания студентов, относящиеся к основам программирования и алгоритмизации, характеризуются полнотой и системностью.

Сопоставление данных диагностики сформированности компетенции бакалавра прикладной информатики в области объектно-ориентированного программирования по личностно-рефлексивному компоненту в ЭГ показало следующее число студентов с высоким и средним уровнем стремления к саморазвитию и самообразованию: 49,36% и 24,41% соответственно, что на 0,19% и 14,89% больше, чем в КГ.

Результаты анализа эмпирических данных в ЭК и КГ свидетельствует о том, что внедрение в учебный процесс разработанной методики обучения основам объектно-ориентированного программирования бакалавров прикладной информатики с использованием визуальных учебных сред повышает уровень профессиональной подготовки студентов.

Расчет среднего арифметического значения в баллах по результатам средних значений в ЭГ и КГ, полученных по четырем критериям (рис. 14), позволил определить общий уровень сформированности компетенции бакалавра прикладной информатики в области объектно-ориентированного программирования. В конце эксперимента такой показатель составил (рис. 16):

- в контрольной группе – **1,62**, т.е. уровень сформированности компетенции бакалавра прикладной информатики в области объектно-ориентированного программирования *достаточный*;
- в экспериментальной группе – **1,89**, что указывает на то, что сформированность компетенции бакалавра прикладной информатики в области объектно-ориентированного программирования у студентов достигла *среднего уровня*.

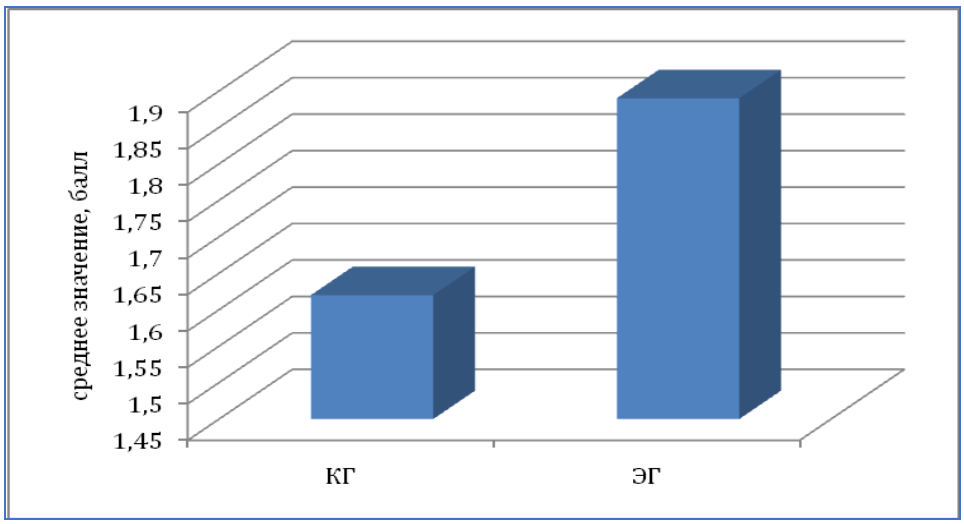


Рисунок 16. Динамика сформированности компетенции бакалавра прикладной информатики в области объектно-ориентированного программирования (среднее по четырем критериям, балл)

К количественным данным, полученным в результате экспериментального исследования, для доказательства чистоты эксперимента применялся многофункциональный математико-статистический критерий «χ^2-Пирсона». Использование критерия χ^2-Пирсона позволяет доказать наличие существенной положительной динамики в уровнях сформированности компетенции бакалавра прикладной информатики в области объектно-ориентированного программирования в экспериментальной группе по отношению к уровню сформированности компетенции в контрольной группе.

Предполагалось, что повышение уровня сформированности компетенции бакалавра прикладной информатики в области объектно-ориентированного программирования может произойти вследствие целенаправленных педагогических воздействий на обучающихся в учебном процессе с использованием методики обучения основам объектно-ориентированного программирования бакалавров прикладной информатики с использованием визуальных учебных сред.

Если разница в уровнях сформированности компетенции бакалавра прикладной информатики в области объектно-ориентированного

программирования экспериментальной и контрольной групп существенна, то есть $\chi^2_{набл} > \chi^2_{крит}$ при уровне значимости 0,05, то ее нельзя объяснить случайными факторами. То есть такие существенные изменения могут быть следствием применения специально организованных педагогических воздействий, что подтверждает эффективность разработанной методики обучения, позволяющей сформировать сформированности компетенции бакалавра прикладной информатики в области объектно-ориентированного программирования. Результаты вычисления критерия χ^2 отражены в таблице 27.

По полученным расчетным результатам в экспериментальной группе эмпирическое значение критерия χ^2 равно 23,776, критическое – 5,991, т.е. $\chi^2_{набл} > \chi^2_{крит}$ (23,776 > 5,991). Достоверность различий характеристик сравниваемых выборок составляет 95%. Поэтому, выдвинутая гипотеза принимается, поскольку достоверность различий характеристик сравниваемых выборок составляет 95%.

Таблица 27

Таблица расчета критерия χ^2

Контрольная группа						
Уровень сформированности компетенции бакалавра прикладной информатики в области объектно-ориентированного программирования	Количество студентов		$\lvert f_1 - f_2 \rvert$	$(f_1 - f_2)^2$	$\lvert f_1 + f_2 \rvert$	$\frac{(f_1 - f_2)^2}{f_1 + f_2}$
	До экспер.	После экспер.				
Достаточный	66	55	11	121	121	0,445
Средний	46	55	10	100	101	0,248
Высокий	12	14	2	4	26	0,200
Всего	124	124	$\chi^2_{крит} = 5,991$			$\chi^2_{набл} = 1,956$
Экспериментальная группа						
Уровень сформированности компетенции бакалавра прикладной информатики в области объектно-	Количество студентов		$\lvert f_1 - f_2 \rvert$	$(f_1 - f_2)^2$	$\lvert f_1 + f_2 \rvert$	$\frac{(f_1 - f_2)^2}{f_1 + f_2}$
	До экспер.	После экспер.				

ориентированного программирования						
Достаточный	73	39	35	1201,78	112	10,73
Средний	51	65	13	169,00	116	1,46
Высокий	7	27	21	441,00	34	12,97
Всего	131	131	$\chi^2_{крит} = 5,991$			$\chi^2_{набл} = 23,776$

Анализ результатов исследования подтверждает эффективность внедрения методики обучения основам объектно-ориентированного программирования бакалавров прикладной информатики с использованием визуальных учебных сред в учебный процесс, которая доказана сравнительным анализом полученных результатов на констатирующем и формирующем этапах эксперимента, достоверность которых проверена средствами математической статистики с помощью критерия χ^2-Пирсона.

Внедрение в учебный процесс разработанной методики обучения основам объектно-ориентированного программирования бакалавров прикладной информатики с использованием визуальных учебных сред обеспечило повышение уровня сформированности компетенции бакалавра прикладной информатики в области объектно-ориентированного программирования. Это подтверждает начальную гипотезу о том, что методика обучения основам объектно-ориентированного программирования бакалавров прикладной информатики с использованием визуальных учебных сред будет эффективной при условии поэтапной ее реализации с системным использованием визуальных учебных сред в процессе обучения бакалавров прикладной информатики, что позволит повысить уровень их профессиональной подготовки.

Результаты проведенного педагогического эксперимента полностью подтвердили гипотезу исследования о целесообразности построения методики обучения основам объектно-ориентированного программирования бакалавров прикладной информатики с использованием визуальных учебных сред на основе предлагаемого подхода и ее эффективность.

ВЫВОДЫ ПО ГЛАВЕ 2

В результате аналитической работы в рамках диссертационного исследования сделаны следующие выводы:

1. Методика обучения основам объектно-ориентированного программирования бакалавров прикладной информатики представлена в виде пяти компонентов: цели, содержание, средства, методы, формы.

Целью обучения бакалавров прикладной информатики основам объектно-ориентированного программирования является формирование целостной *компетенции бакалавра прикладной информатики в области объектно-ориентированного программирования*, которая конкретизируется в четырех аспектах – формирования мотивационно-ценностного, организационно-содержательного, когнитивно-операционного и личностно-рефлексивного компонентов.

Содержание разрабатываемой нами методики представлено на основании материала двух учебных дисциплин «Программирование для начинающих» и «Информатика и программирование» направленных на обучение основам объектно-ориентированного программирования бакалавров прикладной информатики. Содержание обучению конкретизируется через виды и типы учебных заданий, которые предлагаются студентам в дисциплинах «Программирование для начинающих» и «Информатика и программирование», с помощью которых будут сформированы соответствующие умения и навыки:

1) задания-упражнения, направленные на обеспечение усвоения учебного материала и формирования способов деятельности на уровне восприятия, понимания и запоминания;

2) задания на усвоение способов деятельности, направленные на обеспечение усвоения учебного материала и способов деятельности на уровне применения по образцу, в знакомой ситуации;

3) задания на составление тестовых вопросов по определенной теме учебного материала, направленных на качественное изучение учебного

материала курса, формирование таких качеств, как лаконичность и точность выражения мнений, ответственность за результаты труда;

4) задания-исследования, направленные на формирование у студентов навыков работы с учебной, научной литературой и с периодическими изданиями с целью поиска необходимых сведений для выполнения учебных задач, оценки значимости найденных сведений и т.д.;

5) задания по разработке программного продукта, направленные на формирование у студентов навыков и опыта разработки реальных программных продуктов с использованием технологий проектирования, а также развитие творческих способностей студентов.

К средствам учебно-методического обеспечения мы отнесли учебники, учебные пособия, а также сайты проектов Alice и Scratch, библиотеку MSDN, сайт национального открытого университета «Интуит» и др. В качестве технических и программных средств обучения были рассмотрены технические устройства, которые обеспечивают, прежде всего, представление учебной информации, контроль знаний, формирование практических умений и навыков, а также программно-аппаратные средства и устройства, функционирующие на базе компьютерной техники, современные средства и системы информационного обмена, которые обеспечивают операции по поиску, сбору, накоплению, хранению, обработке, представлению, передачи информации. Программные средства организации образовательного процесса отражают предметную область обучения объектно-ориентированному программированию, реализуют те или иные технологии обучения, обеспечивают условия для осуществления и компьютерной поддержки различных видов учебной деятельности:

– педагогические программные средства, предназначенных для организации и компьютерной поддержки учебно-познавательной деятельности (платформы Piazza и OpenClass);

– инструментальные программные средства, предназначенных для подготовки или генерирования учебно-методических и организационных материалов (подготовка презентационных материалов преподавателем средствами Prezi, PowerPoint).

В структуре разрабатываемой нами методики обучения основам объектно-ориентированного были предложены методы обучения – объяснительно-иллюстративный, репродуктивный, исследовательский, словесный, наглядный, практические методы, метод проектов, эвристические методы и метод проблемного обучения. Также в качестве методов обучения основам объектно-ориентированного программирования также были выделены метод целесообразно подобранных задач, метод демонстрационных примеров, предложенные, которые по-нашему мнению способствуют именно формированию профессиональной компетенции в области объектно-ориентированного программирования.

В качестве организационных форм обучения были рассмотрены как лекции, лабораторные работы, самостоятельная работа студентов, контрольные мероприятия.

2. Внедрение в учебный процесс разработанной методики обучения основам объектно-ориентированного программирования бакалавров прикладной информатики с использованием визуальных учебных сред обеспечило повышение уровня сформированности компетенции бакалавра прикладной информатики в области объектно-ориентированного программирования.

Основные результаты второй главы опубликованы в работах [178, 181, 182, 184, 185, 186, 188, 189, 191, 195, 196, 197].

ЗАКЛЮЧЕНИЕ

В диссертации осуществлено теоретическое обобщение и практическое решение проблемы обучения основам объектно-ориентированного программирования бакалавров прикладной информатики с использованием визуальных учебных сред. В процессе проведения диссертационного исследования были решены все поставленные задачи, гипотеза получила подтверждение. Согласно цели и поставленных задач в ходе теоретического поиска и экспериментальной работы получены следующие основные результаты:

– проанализированы роль и место обучения программированию в системе подготовки бакалавров прикладной информатики;

– рассмотрено содержание обучения в области объектно-ориентированного программирования бакалавров прикладной информатики;

– в ходе исследования была сформулирована и определена компетенции бакалавра прикладной информатики в области объектно-ориентированного программирования;

– выделены основные компоненты профессиональной компетентности бакалавра прикладной информатики в области объектно-ориентированного программирования;

– определены уровни сформированности компетенции в области объектно-ориентированного программирования;

– разработана и внедрена структура и содержание учебной дисциплины «Программирование для начинающих»;

– предложены визуальные учебные среды, которые могут быть использованы в процессе обучения основам объектно-ориентированного программирования;

– проведена опытно-экспериментальная работа по оценке эффективности методики обучения основам объектно-ориентированного программирования бакалавров прикладной информатики;

– разработаны учебные пособия «Программирование для начинающих: Alice», «Программирование для начинающих: Alice and Scratch».

Полученные результаты исследования позволяют сделать следующие **выводы**:

1. Обучение основам объектно-ориентированного программирования бакалавров прикладной информатики должно осуществляться на ранних курсах и в рамках самостоятельной дисциплины, взаимосвязанной с курсом технологий программирования на конкретном языке и предваряющей изучение всех других дисциплин профессионального цикла, посвященных разработке компьютерных программ.

2. Компетентность бакалавра прикладной информатики в области объектно-ориентированного программирования предполагает высокий уровень обобщенных профессиональных знаний, готовность к разработке компьютерных приложений в процессе решения профессиональных задач. Эта компетенция является составной частью ядра профессиональной компетентности бакалавра прикладной информатики, которое позволяет выпускнику вуза быть современным и конкурентоспособным на рынке труда

3. Методика обучения основам объектно-ориентированного программирования бакалавров прикладной информатики представлена в виде пяти компонентов: цели, содержание, средства, методы, формы.

4. Использование разработанной и научно обоснованной методики обучения позволяет обеспечить прирост в уровнях сформированности компетенции бакалавра прикладной информатики в области объектно-ориентированного программирования.

Проведённое исследование не претендует на окончательное решение проблемы обучения основам объектно-ориентированного программирования бакалавров прикладной информатики с использованием визуальных учебных сред. Анализ его результатов определяет направления дальнейших исследований, среди которых:

– исследование вопросов индивидуализации обучения объектно-ориентированному программированию бакалавров прикладной информатики;

– разработка эффективных методов оценки и мониторинга уровня сформированности компетенции бакалавра прикладной информатики в области объектно-ориентированного программирования на различных этапах обучения.

СПИСОК ЛИТЕРАТУРЫ

1. Авдеюк, О.А. Олимпиады по информатике как форма выявления и развития одаренности школьников и студентов в области программирования / О.А. Авдеюк, И.Г. Лемешкина, Е.С. Павлова, И.В. Приходькова // Современные наукоемкие технологии. – 2016. – № 5-2. – С. 306-309. URL: http://www.top-technologies.ru/ru/article/view?id=35904.
2. Авдеюк, О.А. Средства развития способностей школьников в области программирования / О.А. Авдеюк, И.Г. Лемешкина, Е.С. Павлова, И.В. Приходькова // Современные проблемы науки и образования. – 2015. – № 3. – С. 300. URL: http://www.science-education.ru/ru/article/view?id=17479
3. Адольф, В.А. Профессиональная компетентность современного учителя / В.А. Адольф. – Красноярск: КГУ, 1998. – 309 с.
4. Азимов, Э.Г. Новый словарь методических терминов и понятий (теория и практика обучения языкам) / Э.Г. Азимов, А.Н. Щукин. – М. : Издательство ИКАР, 2009. – 448 с.
5. Алдашева, А.А. Профессиональная компетентность: понятие и структура / А.А. Алдашева // Вестник Адыгейского государственного университета. Серия 3: Педагогика и психология. – 2012. – №4 (109). URL: http://cyberleninka.ru/article/n/professionalnaya-kompetentnost-ponyatie-i-struktura.
6. Андреев, В.Е. Проблема формирования готовности к профессиональной мобильности у будущих специалистов / В.Е. Андреев // Вестник Череповецкого государственного университета. – 2012. – №1 (42). – С.101-104.
7. Андреева, В.В. Проектирование и реализация системы многоуровневой подготовки специалистов в области

информационных технологий : дис. ... д-ра пед. наук : 13.00.08 / Андреева Валентина Владимировна. – Новгород, 2005. – 375 с.

8. Асламова, Т.В.Организация самостоятельной работы студентов как фактор формирования профессионально значимых компетенций / Т.В. Асламова // Материалы 77-й международной научно-технической конференции ААИ. – Секция 14. – С. 160–165. URL: http://mospolytech.ru/science/aai77/scientific/article/s14/s14_36.pdf
9. Афанасьев, В.В. Математическая статистика в педагогике / В.В. Афанасьев, М. А. Сивов. – ЯГПУ, 2010. – 78 с.
10. Ахо, А.В. Построение и анализ вычислительных алгоритмов / А.В. Ахо, Д.Э. Хопкрофт, Д.Д. Ульман [пер. с англ.]. – М. : Издательство «Мир», 1978. – 535 с.
11. Ахо, А.В. Структуры данных и алгоритмы / А.В. Ахо, Д.Э. Хопкрофт, Д.Д. Ульман [пер. с англ.]. – М. : Издательский дом «Вильямс», 2001. – 384 с.
12. Байденко, В.И. Выявление состава компетенций выпускников вузов как необходимый этап проектирования ГОС ВПО нового поколения (методическое пособие) / В.И. Байденко. – М. : Исследовательский центр проблем качества подготовки специалистов. Российский Новый Университет, 2006. – 55 с. URL: http://www.inpro.msu.ru/PDF/gos_vpo.pdf.
13. Байденко, В.И. Компетентностный подход к проектирования государственных образовательных стандартов высшего профессионального образования (методология и методические вопросы) / В. И. Байденко. – М. : Исследовательский центр проблем качества подготовки специалистов. Российский Новый Университет, 2005. – 144 с.
14. Баранова, Е.М. Формирование ключевых компетенций у студентов технических вузов в процессе обучения математике посредством активных методов обучения / Е.М. Баранова // Известия Пензенского

государственного педагогического университета им. В.Г. Белинского. – 2011. – № 24. – С. 544-549.

15. Барков, И.А. Преподавание дисциплины «Объектно-ориентированное программирование» / И.А. Барков // ОТО. – 2009. – №4. URL: https://cyberleninka.ru/article/n/prepodavanie-distsipliny-obektno-orientirovannoe-programmirovanie.
16. Басова, Н. В. Педагогика и практическая психология / Н.В. Басова. – Ростов н/Д: «Феникс», 1999. – 416 с.
17. Белов, М.П. Основы алгоритмизации в информационных системах: Учебное пособие / М.П. Белов. – СПб.: СЗТУ, 2003. – 85 с.
18. Бережнова, Е.В. Исследования в области образования: проблемы управления качеством: монография [Текст] / Е.В. Бережнова, В.В. Краевский. – Москва, РАО, 2007. – 150 с.
19. Бермус, А.Г. Проблемы и перспективы реализации компетентностного подхода в образовании: (Проблемы компетентностного подхода) / А.Г. Бермус // Интернет-журнал «Эйдос». – 2005. – 10 сентября. URL: http://www.eidos.ru /journal/2005/0910-12.htm.
20. Бобров, А.Н. Проблемы выбора языка программирования в школьном курсе информатики / А.Н. Бобров // Молодой ученый. – 2015. – №24. – С. 61–64. URL: https://moluch.ru/archive/104/24471/
21. Бодряков, В.Ю. Практический опыт формирования исследовательских компетенций студентов, обучающихся по направлению «01. 03. 02 – Прикладная математика и информатика» / В.Ю. Бодряков, Л.Р. Ушакова // Педагогическое образование в России. – 2015. – №7. URL: https://cyberleninka.ru/article/n/ prakticheskiy-opyt-formirovaniya-issledovatelskih-kompetentsiy-stud entov-obuchayuschihsya-po-napravleniyu-01-03-02-prikladnaya.

22. Болотов, В.А. Компетентносная модель: от идеи к образовательной программе / В.А. Болотов, В.В. Сериков // Педагогика. – 2003. – № 10. – С. 8–14.
23. Бордовская, Н.В. Педагогика. Учебник для вузов / Н.В. Бордовская, А.А. Реан. – Издательство «Питер», 2000. – 304 с.
24. Борисова, Е.А. Из опыта обучения программированию на занятиях по информатике в экономическом вузе / Е.А. Борисова // Проблемы и перспективы развития образования: материалы Междунар. науч. конф. (г. Пермь, апрель 2011 г.). Т. II. – Пермь: Меркурий, 2011. – С. 45-47.
25. Бороненко, Т.А. Методика обучения информатике (теоретические основы) [Текст]: учеб. пособие / Т.А. Вороненко. – СПб. : Изд-во РГПУ им. А. И. Герцена, 1997. – 99 с.
26. Бочкин, А.И. Методика преподавания информатики / А. И. Бочкин. – Минск: Вышейшая школа, 1998. – 431 с.
27. Брукс, Ф.П. Как проектируются и создаются программные комплексы / Брукс Ф.П. [пер. с англ.]. – М. : Наука, 1979. – 151 с.
28. Бурдин, А.О. О классификации задач / А.О. Бурдин // Совершенствование содержания и методов обучения естественно-математическим дисциплинам в средней школе. – М., 1981. – С.3–7.
29. Бутусова, Е.И. Опыт использования технологии проблемного обучения на занятиях Основ алгоритмизации и программирования / Е.И. Бутусова. URL: https://nsportal.ru/shkola/informatika-i-ikt/library/2015/04/23/statya-opyt-ispolzovaniya-tehnologii-problemnogo.
30. Быков, А.А. Особенности реализации компетентностного подхода при подготовке специалистов прикладной информатики и вычислительной техники на базе смоленских вузов / А.А. Быков // Электронный научно-практический журнал «Современная педагогика». – 2014. – №10. URL: http://pedagogika.snauka.ru/2014/10/2753.

31. Векслер, В.А. «Scratch» среда программирования для детей / В.А. Векслер // Современная педагогика. – 2015. – № 5. URL: http://pedagogika.snauka.ru/2015/05/4251.
32. Востроknутов И.Е., Калинина О.С. Применение метода программных фрагментов при обучении визуальному программированию на базовом уровне в средней школе / И.Е. Востроknутов, О.С. Калинина // Педагогическая информатика. – 2014. – № 1. – С. 22–31.
33. Востроknутов, И.Е. Основные характеристики визуальных сред программирования / И.Е. Востроknутов // Педагогическая информатика. – 2011. – № 5. – С. 32–36.
34. Востроknутов, И.Е. Проблема подготовки IT специалистов в отечественных вузах / И.Е. Востроknутов, Д.В. Шагбазян // Информатизация образования. – 2017. – С. 120–123.
35. Востроknутов, И.Е. Проблемы обучения информатике и информационным и коммуникационным технологиям в условиях пенитенциарной системы / И.Е. Востроknутов, Н.М. Токарев // Педагогическая информатика. – 2016. – № 1. – С. 13–18.
36. Востроknутов, И.Е. Программирование в предпрофильном обучении информатике и информационным и коммуникационным технологиям в условиях дополнительного образования / И.Е. Востроknутов, Н.Г. Саблукова // Электронное периодическое издание Информационная среда образования и науки. – 2012. – № 7. – С. 10–17. URL: http://www.iiorao.ru/iio/pages/izdat/ison/publication/ison_2012/num_7_2012/Vostroknutov_Sablukova.pdf.
37. Востроknутов, И.Е. Структура содержания обучения программированию в визуальных средах / И.Е. Востроknутов, Н.Г. Саблукова // Педагогическая информатика. –2011. – № 6. – С. 14–20.

38. Вузы России // Справочник высших учебных заведений. URL: http://www.edu.ru/vuz/.
39. Голант, Е.Я. О развитии самостоятельности и творческой активности учащихся в процессе обучения. – В кн.: Воспитание познавательной активности и самостоятельности учащихся / Е.Я. Голант. – Казань, 1969 – 152с.
40. Готская, И.Б. Маркетинговое проектирование методической системы обучения информатике студентов педвузов [Текст]: монография / И.Б. Готская. – СПб. : Изд-во РГПУ им. А. И. Герцена, 1999. – 114 с.
41. Гриншкун, В.В. Информационные технологии и дизайн: международный опыт применения исследовательских подходов к обучению информатике / В.В. Гриншкун // Вестник Российского университета дружбы народов. Серия: Информатизация образования. – 2016. – № 4. – С. 7-13.
42. Гриншкун, В.В. Обучение школьников алгоритмизации и программированию на основе использования игровой платформы Unity 3d / В.В. Гриншкун, И.Н Шегай // Математика и математическое образование сборник трудов по материалам VIII международной научной конференции «Математика. Образование. Культура» (к 240-летию Карла Фридриха Гаусса). – 2017. – С. 49-53.
43. Гриншкун, В.В. Особенности использования открытых электронных ресурсов и массовых учебных курсов в высшем образовании / В.В. Гриншкун, Г.А. Краснова, А. Нухулы // Вестник Московского городского педагогического университета. Серия: Информатика и информатизация образования. – 2017. – № 2 (40). – С. 8-17.
44. Гриценко, Л.И. Влияние типа познавательной деятельности учащихся старших классов на усвоение ими знаний : автореф. дис. … канд. пед. наук : 13.00.02 / Гриценко Людмила Ивановна. – Красноярск, 1972. – 28 с.

45. Громкова, М.Т. Педагогика высшей школы: учеб. пособие для студентов педагогических вузов / М.Т. Громкова. – М. : ЮНИТИ-ДАНА, 2012. - 447с.
46. Данилова, М.А. Дидактика средней школы / М.А. Данилова, М.Н. Скаткина. – М., 1975. – 256 с.
47. Данильчук, Е. В. Эволюция курса информатики в школе: поиск новой парадигмы подготовки будущего учителя информатики в педагогическом вузе / Е.В. Данильчук // Известия Волгоградского Государственного Педагогического Университета. – 2011. – №8 (62). – С. 62-68. URL: https://elibrary.ru/contents.asp?issueid=1005559.
48. Дейкстра, Э. Дисциплина программирования / Э. Дейкстра [пер. с англ]. – М. : Издательство «Мир», 1978. – 274 с.
49. Елагина, Л.В. Формирование культуры профессиональной деятельности будущего специалиста на основе компетентностного подхода: методология, теория, практика : дис. … доктора пед. наук : 13.00.08 / Елагина Людмила Васильевна. – Челябинск, 2009. – 464с.
50. Епишева, О.Б. Актуальные вопросы современного образования: монография / О.Б. Епишева (§ 1 главы первой), В.А. Игнатова (§ 2 главы второй), Н.А. Беззубцева и др. – Тюмень: ТюмГНГУ, 2010. – 404 с.
51. Еремина, А.В. Идентификация ключевых компетенций выпускников вузов / А.В. Еремина, И.В. Зороастрова, Е.О. Сучкова // Studia Humanitatis Международный электронный научный журнал. – 2015. – №4. URL: http://st-hum.ru/content/eremina-av-zoroastrova-iv-suchkova-eo-identifikaciya-klyuchevyh-kompetenciy-vypusknikov.
52. Ершов, А.П. Школьная информатика в СССР: от грамотности к культуре. Доклад на второй международной конференции «Дети в мире компьютеров» (Новосибирск, 1987) / А.П. Ершов // Архив академика А. П. Ершова. – 2000. URL: http://www.ershov.ras.ru /archive/eaindex.asp?did=3093.

53. Ершов, А.П. ЭВМ в классе. Школа на путях к реформе / А.П. Ершов // Архив академика А. П. Ершова. – 2000. URL: http://www.ershov.ras.ru/archive/eaindex.asp?did=18855.
54. Ершов, А.П. Информатизация: от компьютерной грамотности учащихся к информационной культуре общества / А.П. Ершов // Коммунист. – 1988. – № 2. – С. 82-92.
55. Ершов, А.П. Программирование – вторая грамотность / А.П. Ершов // Архив академика А.П. Ершова. – 1981. URL: http:// eshov.iis.nsksu/russian/second_literacy/article. html.
56. Жалдак, М.И. Теория вероятностей с элементами информатики. Практикум. Учебное пособие / М.И. Жалдак, А.Н. Квитко [под общ. ред. М.И. Ядренко]. – К.: Высшая школа, 1989. – 263 с.
57. Жужжалов, В.С. Совершенствование содержания обучения программированию на основе интеграции парадигм программирования : дис. … д-ра пед. наук : 13.00.02 / Жужжалов Валерий Евгеньевич. – Москва, 2004. – 274 с.
58. Жульковская, И.И. К выбору стратегии преподавания информатики для инженерных специальностей университетов / И.И. Жульковская, О.А. Жульковский // Сборник научных трудов Днепрдзержинского государственного технического университета. – 2013. – Вып. 1. – С. 171-176.
59. Загвязинский, В.И. Теория обучения: Современная интерпретация: Учебное пособие для вузов / В.И. Загвязинский. – М. : Академия, 2006. – 192 с.
60. Зайченко, И.В. Педагогика: учебное пособие для студ. высш. пед. учеб. заведений / И. В. Зайченко. – К. : Образованпие Украины, 2006. – 528 с.
61. Зверинцева, М.Е. Рекомендации по преподаванию информатики в университетах / М.Е. Зверинцева, Т.В. Зверинцева, Н.Ю.Курочка, А.А. Симановский, Д.А. Шапоренков. – СПб., 2002. –372 с.

62. Зимняя, И.А. Компетенция и компетентность в контексте компетентностного подхода в образовании / И.А. Зимняя [Электронный ресурс]. URL: http://www.rusreadorg.ru/ckeditor_assets/attachments/63/i_a_zymnaya_competency_and_competence.pdf.
63. Зимняя, И.А. Ключевые компетентности как результативно-целевая основа компетентностного подхода в образовании / И.А. Зимняя. – М. : Исследовательский центр проблем качества подготовки специалистов, 2004. – 213 с.
64. Иванова, Г.С. Основы программирования: Учебник для вузов / Г.С. Иванова. – М. : Изд-во МГТУ им. Н.Э. Баумана, 2002. – 416 с.
65. Истомина-Кастровская, Н.Н. Эволюция учебных заданий в связи с изменением содержания образования (на материале изучения математики в нач. кл.) : автореф. дис. ... канд. пед. наук: 13.00.01 / Истомина-Кастровская Наталия Борисовна. – М., 1973. – 20с.
66. Касторнова, В.А. Методика создания и использования прикладных программ на основе мультимедиа технологии в обучении информатике : дис. … канд пед наук : 13.00.02 / Касторнова Василиса Анатольевна. – Москва, 1998. – 193с.
67. Керниган, Б.В. Практика программирования / Б.В. Керниган, Р. Пайк [пер. с англ.]. – СПб. : «Невский диалект», 2001. – 381 с.
68. Керниган, Б.В. Язык программирования С. / Б. Керниган, Д. Ритчи [пер. с англ.]. – СПБ. : «Невский диалект», 2001. – 352 с.
69. Компетентности и компетентностный подход в современном образовании [Электронный ресурс]. URL: http://festival.1september.ru/articles/581708/.
70. Коннова, М.В. Психологический аспект компьютеризации образования / М.В. Коннова // Современные информационные технологии и инновационные методики обучения в подготовке специалистов: методология, теория, опыт, проблемы. –2000. – С.179–180.

71. Кормен, Т. Алгоритмы: построение и анализ / Т. Кормен, Ч. Лейзерсон, Р. Ривест, Р. Штайн. – М. Издательский дом «Вильямс», 2005. – 1296 с.
72. Кудаев, М.Р. Методология и методика педагогических исследований. Учебное пособие / М.Р. Кудаев. – Майкоп: Изд-во АГУ, 2003. – 168с.
73. Кудинов, Ю.В. Педагогические пути повышения эффективности процесса профессиональной подготовки студентов в высшей школе [Электронный ресурс]. URL: http://archive.nbuv.gov.ua/portal/soc_gum/znpkhist/2009_1/09kyvsvh.pdf.
74. Кудрина, Е.В. Школа-вуз: осуществление преемственности в обучении программированию / Е.В. Кудрина, М.В. Огнева // Научное обозрение. Педагогические науки. – 2017. – № 5. – С. 108-113.
75. Кузнецов, А.Б. Методика обучения учащихся классов с углубленным изучением информатики объектно-ориентированному проектированию программ : дис. … канд. пед. наук : 13.00.02 / Кузнецов Александр Борисович. – Екатеринбург, 1999. – 268 с.
76. Кузьмина, Н.В. Методы системного педагогического исследования: Учебное пособие / Н.В. Кузьмина. – Л.: ЛГУ, 1980. – 172 с.
77. Кузьминский А.И. Технология и техника школьного урока: Учебное пособие / А.И. Кузьминський, С.В.Омельяненко.– К., 2010. – 335 с.
78. Кулюкина, Е.С. Формирование компетенций бакалавров и магистров технических профилей с учетом международных стандартов : автореф. дис. ... канд. пед. наук : 13.00.08 / Кулюкина Евгения Сергеевна. – М., 2011. – 23 с.
79. Кыверялг, А.А. Методы исследования в профессиональной педагогике / А.А. Кыверляг. – Талин: Валгус, 1980. – 334 с.
80. Лазарева, Л.Ю. Модель методической системы формирования профессиональной компетентности будущих специалистов в условиях информатизации образования / Л.Ю. Лазарева, С.Н. Ларин // Личность, семья и общество: вопросы педагогики и психологии:

сб. ст. по матер. XII междунар. науч.-практ. конф. Часть II. – Новосибирск: СибАК. – 2012. URL: https://sibac.info/conf /pedagog/xii/26402.

81. Лапо, А. Методика построения систем задач по программированию в школьном курсе информатики / А. Лапо // Вести БДПУ. – 2014. – Серия 3. №4. – С. 50–56. URL: http://elib.bspu.by/handle/doc/8547.
82. Лаптев, В.В. Методическая теория обучения информатике. Аспекты фундаментальной подготовки / В.В. Лаптев, Н.И. Рыжова, М.В Швецкий. – СПб. : Изд-во С.-Петерб. ун-та, 2003. – 352 с.
83. Лапчик, М.П. ИКТ-компетентность бакалавров образования / М. П. Лапчик // Информатика и образование. – 2012. – № 2. – С. 29–33.
84. Лапчик, М.П. Информатика и информационные технологии в системе общего и педагогического образования: монография / М.П. Лапчик. – Омск: изд-во Омского гос. пед. ун-та, 1999. – 276 с.
85. Лапчик, М.П. Теория и методика обучения информатике: учебник / М.П. Лапчик, И.Г. Семакин, Е.К. Хеннер [под общей ред. М.П. Лапчика]. – М. : Академия, 2008. – 592 с.
86. Ларионова, И.А. Профессиональная подготовка специалистов социальной работы: компетентностный подход / И.А. Ларионова, В.А. Дегтерев // Фундаментальные исследования. – 2014. –№ 11 (часть 12) – С. 2734-2739. URL: https://www.fundamental-research.ru/ru/article/view?id=36055.
87. Левина, Л.М. Инновационные аспекты самостоятельной работы студентов в контексте болонского процесса и модернизации высшей школы / Л.М. Левина // Инновации в образовании. Вестник Нижегородского университета им. Н.И. Лобачевского. – 2010. – № 6. – С. 17–22. URL: http://www.unn.ru/pages/issues/vestnik /99999999_West_2010_6/2.pdf.

88. Макконелл, Дж. Анализ алгоритмов. Вводный курс / Дж. Макконелл. – Москва: Техносфера, 2002. – 304 с.
89. Машбиц, Е.И. Психолого–педагогические проблемы компьютеризации обучения / Е.И. Машбиц. – М.: Педагогика, 1988. – 192с.
90. Можаров, М.С. Использование современных технологий в области интерактивного обучения программированию: тенденции и перспективы / М.С. Можаров // Вестник ТГПУ. – 2017. №5 (182). – С. 134-140. URL: https://cyberleninka.ru/article/n/ispolzovanie-sovremennyh-tehnologiy-v-oblasti-interaktivnogo-obucheniya-programmirovaniyu-tendentsii-i-perspektivy.
91. Нельзина, О.Г. Проблемы обучения программированию по курсу информатики в системе «школа-вуз» / О.Г. Нельзина // RELGA – научно-культурологический журнал (№13 [135]. – 2006. URL: http://www.relga.ru/Environ/WebObjects/tgu-www.woa/wa/Main?textid=1087&level1=main&level2=articles.
92. Нехожина, Е.П. Моделирование технологии формирования профессиональной компетентности будущих инженеров в сфере информационных технологий / Е.П. Нехожина // Молодой ученый. — 2011. — №7. Т.2. — С. 110-112.
93. Нехожина, Е.П. Формирование профессиональной компетентности инженеров по программному обеспечению вычислительной техники и автоматизированных систем : дис. ... канд. пед. наук: 13.00.08 / Нехожина Евгения Петровна. – Димитровград, 2009. – 267 с.
94. Новиков, Д.А. Статистические методы в педагогических исследованиях (типовые случаи) / Д.А. Новиков. – М. : МЗ-Пресс, 2004. – 67 с.
95. Нурбекова, Ж.К. Теоретико-методологические основы обучения программированию [Текст]: монография / Ж.К. Нурбекова. – Павлодар, 2004. – 225 с.

96. Нуриев, Н.К. Проектирование дидактической системы инновационной подготовки специалистов в области программной инженерии : автореф. дис. ... д-ра пед. наук: 13.00.08/ Нуриев Наиль Кашапович. – Казань, 2008. – 192 с.
97. Овчинникова, Е.Н. К определению терминов «учебник» и «учебное пособие» / Е.Н. Овчинникова // Электронный научно-практический журнал «Гуманитарные научные исследования». URL: http://human.snauka.ru/2012/05/1189.
98. Огнева, М.В. Программирование на языке С++. Практический курс: Учебное пособие для бакалавриата и специалитета. / М.В. Огнева, Кудрина Е.В. – Москва, 2017. – 335 с.
99. Одарич, И.Н. Моделирование формирования профессиональной компетентности бакалавра / И.Н. Одарич // Вектор науки ТГУ. – 2015. – № 2 (32–1). – С. 186–189.
100. Одинцов, И.О. Профессиональное программирование. Системный подход / И.О. Одинцов. – СПб. : БХВ Петербург, 2002. – 512 с.
101. Онищук, В.А. Психолого-педагогические требования к заданиям и упражнениям в учебнике / В.А. Онищук // Проблемы школьного учебника. – Вып. 3. – 1975. – С. 130–137.
102. Орлов, А.А. Педагогика: концепция и учебная программа курса для студентов пед. вуза [Текст] / А.А. Орлов. – Тула: Изд-во ТГПУ им. Л.Н. Толстого, 2001. – 34 с.
103. Орлова, М.С. Система смешанного обучения программированию, ориентированная на формирование профессиональной коммуникативной компетентности : дис. … канд. пед. наук: 13.00.02 / Орлова Марина Сергеевна. – Москва, 2009. – 186 с.
104. Осмоловская, И.М. Наглядные методы обучения: учеб. пособие для студ. высш. учеб. заведений / И.М.Осмоловская. – М. : Издательский центр «Академия», 2009. – 192 с.

105. Осмоловская, И.М. Словесные методы обучения: учеб. пособие для студ. высш. учеб. заведений / И.М.Осмоловская. – М. : Издательский центр «Академия», 2008. – 172 с.
106. Основные понятия по дисциплине «Функциональное программирование и интеллектуальные системы». [Электронный ресурс]. – Екатеринбург 2015. – URL: edu.usfeu.ru/Uploads/MetodObespech/KursLekzii/3803052/3803052_52.pdf.
107. Официальный сайт Alice [Электронный ресурс]. URL: http://www.alice.org/
108. Официальный сайт Microsoft [Электронный ресурс]. URL:: www.microsoft.com.
109. Официальный сайт OpenClass [Электронный ресурс]. URL: https://www.pearsonhighered.com/products-and-services/institutional-services-and-solutions/openclass.html.
110. Официальный сайт Piazza The incredibly easy, completely free [Электронный ресурс]. URL: Q&A platform https://piazza.com/.
111. Павличенко, М.А. Проблема обучению программированию в школе / М.А. Павличенко // Социальная сеть работников образования nsportal.ru. – 2014. URL: https://nsportal.ru/shkola/obshchepedagogicheskie-tekhnologii/library/2014/10/19/problema-obucheniyu-programmirovaniyu-v
112. Павлова, Е.С. Методика использования систем задач как средства развития одаренности при подготовке школьников к олимпиадам по информатике : автореф. дис. … канд. пед. наук : 13.00.02 / Павлова Елена Станиславовна. – Волгоград, 2014. – 26с.
113. Павлова, Е.С. Методика развития способностей учащихся в области программирования / Е.С. Павлова, О.А. Авдеюк, И.Г. Лемешкина, И.В. Приходькова // Инновации на основе информационных и коммуникационных технологий. – 2015. – Т. 1. – С. 16–18.

114. Павлова, Е.С. Методика формирования одаренности при подготовке к олимпиадам по информатике / Е.С. Павлова // Фундаментальные исследования. – 2013. – № 10-6. – С. 1360-1362. URL: http://www.fundamental-research.ru/ru/article/view?id=32547.
115. Павлова, Е.С. Основные критерии одаренности учащихся в области программирования / Е.С. Павлова, О.А. Авдеюк, Д.Н. Авдеюк // Инновации на основе информационных и коммуникационных технологий. – 2015. – Т. 1. – С. 10–11.
116. Пелевин, В.Н. Формирование профессиональной компетентности будущих бакалавров по направлению «Информационные системы и технологии» : дис. ... канд. пед. наук :13.00.08 / Пелевин Владимир Николаевич. – Екатеринбург, 2010. – 189 с.
117. Петров, Алексей Николаевич Совершенствование методики обучения объектно-ориентированному программированию на основе объектно-ориентированного проектирования: на примере дисциплины «Программирование» для будущих учителей информатики : автореф. дис. ... канд. пед. наук : 13.00.02 / Петров Алексей Николаевич. – Москва, 2009.– 18 с.
118. Пешкова, В.Е. Педагогика. Часть 4. Теория обучения (дидактика) Курс лекций: (Учебное пособие) / В.Е. Пешкова. – Майкоп, 2010. – 148 с. URL: http://www.adygnet.ru/sites/default/files/pedagogik_ch4.pdf.
119. Пидкасистый, П.И. Педагогика. Учебник / П.И. Пидкасистый. – М. : 2006. – 608с.
120. Писоцкая, М.Э. Опережающие задания как способ формирования самостоятельности учащихся : автореф. дис. ... канд. пед. наук : 13.00.01 / Писоцкая Марина Эмильевна. – Харьков,1992. –17с.
121. Пичкова, Л.С. Организация самостоятельной работы студентов как фактор формирования профессионально значимых компетенций / Л.С. Пичкова // Пути повышения конкурентоспособности экономики России в условиях глобализации. – М. : МГИМО-Университет, 2008.

URL: https://mgimo.ru/upload/iblock/b39/b392841de7ffeb6159bc9c93d40de04f.pdf.

122. Полат, Е.С. Дидактические основы комплексного использования средств обучения в учебно-воспитательном процессе общеобразовательной школы / Е.С. Полат, З.И. Батюкова и др. – М. : НИИ СО и УК, 1991. – 137с.
123. Прата, С. Язык программирования С++. Лекции и упражнения. Учебник / С. Прата [пер. с англ.]. – Издательство «ДиаСофт», 2001. – 656 с.
124. Пратт, Т. Языки программирования: разработка и реализация / Т. Пратт, М. Зелковиц. – СПб. : Питер, 2002. – 688 с.
125. Прозорова, Г.В. Формирование профессиональных компетенций бакалавров-инженеров по направлению «информационные системы и технологии» в вузе : дис. … канд. пед. наук : 13.00.08 / Прозорова Галина Владимировна. – Тюмень, 2015. – 207 с.
126. Пышкало, А. М. Методическая система обучения геометрии в начальной школе [Текст]: авторский доклад по монографии «Методика обучения элементам геометрии в начальных классах», представленной на соискание ... д-ра пед. наук. / А. М. Пышкало. – М.: Академия пед. наук СССР, 1975. – 60 с.
127. Резник, Л.М. Отдельные аспекты совершенствования современной профессиональной подготовки будущих специалистов в высших учебных заведениях / Л.М. Резник // Вестник Черкасского университета. – Вып. 136. –2008. – С. 132-135.
128. Рейтинг языков программирования // Сообщество программистов DOU. – Режим доступу: http://www.tiobe.com/index.php/content/paperinfo/tpci/index.html.
129. Роберт, И.В. Теория и методика информатизации образования (психолого-педагогический и технологический аспекты). 2-е издание, дополненное / И.В. Роберт. – М. : ИИО РАО, 2008. – 274 с.

130. Рыжова, Н.И. Модель методики обучения учителей информатики использованию информационно-образовательных систем обучения в профессиональной деятельности / Рыжова, Н. И., Ляш А.А [Электронный ресурс]. URL: http://www.gup.ru/events /news/smi/rijova_inf.pdf.
131. Саблукова, Н.Г. Методические подходы к обучению программированию в визуальных средах в условиях дополнительного образования : автореф. дис. ... канд. пед. наук: 13.00.02 / Саблукова Наталья Геннадьевна. – Москва, 2012. – 19 с.
132. Саблукова, Н.Г. Разработка методики обучения программированию в предпрофильном обучении информатике и ИКТ в условиях дополнительного образования / Н.Г. Саблукова, И.Е. Востроknутов // Педагогическая информатика. – 2011. – № 1. – С. 28–33.
133. Садулаева, Б.С. Объектно-ориентированное программирование в обучении будущих бакалавров информатики / Б.С. Садулаева // Инновационная наука. – 2015. – №10-3. URL: http://cyberleninka.ru/ article/n/obektno-orientirovannoe-programmirovanie-v-obuchenii-buduschih-bakalavrov-informatiki.
134. Садулаева, Б.С. Формирование специальных компетенций будущих бакалавров профиля «Информатика» в процессе обучения математической информатике : дис. … канд. пед. наук : 13.00.02 / Садулаева Билянт Султановна, Челябинск, 2012. – 216с.
135. Сайт академика РАО Новикова А.М. Формы обучения в современных условиях [Электронный ресурс]. URL: http://www.anovikov.ru/artikle/forms.pdf.
136. Сайт компании Association for Computing Machinery: Advancing Computing as a Science & Profession [Электронный ресурс]. URL: http://www.acm.org/.

137. Сайт компании BT Group PLC [Электронный ресурс]. URL: http://home.bt.com/.
138. Сайт компании Cisco Systems [Электронный ресурс]. URL: http://www.cisco.com/.
139. Сайт компании IBM Europe [Электронный ресурс]. URL: http://www-03.ibm.com/able/europe/.
140. Сайт компании Intel [Электронный ресурс]. URL: www.intel.com/.
141. Сайт компании Microsoft Europe [Электронный ресурс]. URL: http://news.microsoft.com/europe/.
142. Сайт компании Nokia [Электронный ресурс]. URL: http://www.nokia.com/ru_int.
143. Сайт компании Philips Semiconductors [Электронный ресурс]. URL: http://www.mouser.com/philips-semiconductors.
144. Сайт компании Siemens AG [Электронный ресурс]. URL: http://www.siemens.com/.
145. Сайт компании Thales [Электронный ресурс]. URL: https http://www.thalesgroup.com.
146. Сайт корпорации DigitalEurope [Электронный ресурс]. URL: http://www.digitaleurope.org/.
147. Сакаева, А.Н. Повышение качества профессиональной подготовки студентов в процессе дистанционного обучения : автореф. дис. … канд. пед. наук : 13.00.08 / Сакаева Альфина Нигаматзяновна. – Республика Казахстан Караганды, 2009. – 29с.
148. Самерханова, Э.К. Формирование профессиональной рефлексии студентов в области прикладной информатики / Э.К. Самерханова, М.Г. Гайда // Проблемы современного педагогического образования. – 2016. – №53 (11). – С. 71–77.
149. Себеста, Р. Основные концепции языков программирования / Р. Себеста. – М.: Издательский дом «Вильямс», 2001. – 672 с.

150. Сейдаметова, З.С. Факторы, влияющие на IT-образование: рынок труда, образовательные стандарты, языки программирования / З.С. Сейдаметова, В.А. Темненко // Инженерия программного обеспечения. – 2010. — № 1. – С. 62 –70.
151. Сейтвелиева, С.Н. Программирование на языке С++: учебное пособие для студентов и преподавателей компьютерных специальностей высших учебных заведений / С.Н. Сейтвелиева, Ф.С. Ильясова. – Симферополь: ИП Хотеева Л.В, 2017. – 104 с.
152. Семакин, И.Г. Научно-методические основы построения базового курса информатики : дис.: … д-ра пед. наук : 13.00.02 / Семакин Игорь Геннадьевич. – Пермь, 2002. – 415с.
153. Сериков Г.Н. Управление достижением качества образования [Текст] / Г.Н. Сериков. – Челябинск, изд-во ЮУрГУ, 2009. – 265 с.
154. Синкина, Е.А. Моделирование процесса формирования профессиональных компетенций бакалавров через проектирование содержания общепрофессиональных дисциплин / Е.А. Синкина // Вестник ПНИПУ. Машиностроение, материаловедение. – 2013. – №1. URL: http://cyberleninka.ru/article/n/modelirovanie-protsessa-formirovaniya-professionalnyh-kompetentsiy-bakalavrov-cherez-proektirovanie-soderzhaniya.
155. Скворцова, С.А. Педагогические условия формирования компетентности будущих специалистов в процессе профессиональной подготовки / С.А. Скворцова // Вектор науки ТГУ. – 2011. – № 1(4). – С. 155–158.
156. Сластенин, В.А. Педагогика: Учеб. пособие для студ. высш. пед. учеб. заведений / В.А. Сластенин, И.Ф. Исаев, Е.Н. Шиянов [под ред. В.А. Сластенина]. – М.: Издательский центр «Академия», 2002. – 576 с.
157. Смульсон, М.Л. Психология развития интеллекта [Текст]: монография / М.Л. Смульсон. – К., 2001. – 276 с.

158. Соколова, И.Ю. От самопознания к самореализации и здоровье сбережению. Учебно-методическое пособие для студентов, магистрантов, аспирантов, кураторов, педагогов (электронный вариант) / И.Ю. Соколова, Л.Б. Гиль. – Томск: ТПУ, 2010. – 100 с. http://www.lib.tpu.ru/fulltext/m/2010/m32.pdf.
159. Сухомлин, В.А. Анализ международных образовательных стандартов в области информационных технологий / В.А.Сухомлин // Современные информационные технологии и ИТ-образование. – 2011. –С.16-46.
160. Тельнов, Ю.Ф. Разработка профильно-ориентированных основных образовательных программ ВПО по направлению «Прикладная информатика» [Электронный ресурс] / Ю.Ф. Тельнов // Материалы X Всероссийской конференции «Преподавание информационных технологий в Российской Федерации». – Саратов, 2011. URL: http://www.it-education.ru/2011/section/77/3814/index.html.
161. Толлингерова, Д.А. Опережающее управление учебной деятельностью: автореф. дис. … д-ра психол. наук: 19.00.01 / Толлингерова Дана Александровна. – М.,1981. – 35с.
162. Толлингерова, Д.А. Психология проектирования умственного развития детей / Д. Толлингерова, Д. Голоушова, Г. Канторкова. – Москва – Прага: Роспедагенство, 1994. – 48 с.
163. Традиционная иерархия мыслительных процессов [Электронный ресурс]. URL http://www.intel.ru/content/dam/www/program/education/emea/ru/ru/documents/project-design1/thinking-skills/bloom-taxonomy.pdf.
164. Уман, А.И. Учебные задания и процесс обучения / А.И. Уман. – М. : Педагогика, 1989. – 56с.
165. Фадеев, О.В. Характеристика и применение технических средств обучения / О.В. Фадеев, Д.В. Пешков // Проблемы и перспективы

развития образования: материалы II междунар. науч. конф. (г. Пермь, май 2012 г.). – Пермь: Меркурий, 2012. – С. 185-187.

166. Фатеева, И.А. Метод проектов как приоритетная инновационная технология в образовании / И.А. Фатеева, Т.Н. Канатникова // Молодой ученый. – 2013. – №1. – С. 376–378.
167. Федеральный закон «Об образовании в Российской Федерации» Федеральный закон от 29.12.2012 № 273-ФЗ «Об образовании в Российской Федерации» в редакции от 29.12.2017 года [Электронный ресурс]. URL: https://dokumenty24.ru/zakony-rf/zakon-ob-obrazovanii-v-rf.html
168. Феоктистова, О.А. Создание простых приложений iot средствами Scratch / О.А. Феоктистова, М.В. Храмова // Информационные технологии в образовании «ИТО-САРАТОВ-2017» Материалы IX Всероссийской научно-практической конференции. – 2017. – С. 81-84.
169. Фьюер, А. Задачи по языку Си / А. Фьюер. – М. : Финансы и статистика, 1985. URL: http://khpi-iip.mipk.kharkiv.edu/library/extent/prog/fuer/.
170. Хуторской, А.В. Ключевые компетенции и образовательные стандарты / А.В. Хуторской // Интерет-журнал «Эйдос». – 2002. – 23 апреля. URL: http://www.eidos.ru/journal/2002/0423.htm.
171. Цитаты великих и известных людей, афоризмы. Андрей Петрович Ершов [Электронный ресурс]. URL: http://citaty.info/man/andrei-petrovich-ershov.
172. Швецкий, М.В. Методическая система фундаментальной подготовки будущих учителей информатики в педагогическом вузе в условиях двухступенчатого образования : дис. ... до-ра. пед. наук : 13.00.02 / Швецкий Михаил Владимирович. – Санкт-Петербург, 1994. – 446 с.
173. Шибаева, А.С. Эвристический метод как средство активизации познавательной деятельности учащихся [Электронный ресурс] / А.С.

Шибаева, С.В. Корякина. – URL: // http://www.rae.ru /forum2010/43/550.

174. Шкарбан, Ф.В. Введение в ООП: использование специальных программных сред / Ф.В. Шкарбан, С.М Сейдаметова // Science and Education a New Dimension. Pedagogy and Psychology, III(32), Issue: 63. – 2015. – S. 48–52.
175. Шкарбан, Ф.В. Использование визуальной учебной среды Alice для обучения основам объектно-ориентированного программирования студентов вуза / Ф.В. Шкарбан // Электронный научно-образовательный журнал ВГСПУ «Грани познания». – 2017. – № 6(53). – URL: http://grani.vspu.ru/files/publics/1512651519.pdf.
176. Шкарбан, Ф.В. Классификация и систематизация процесса обучения программированию / Ф.В. Шкарбан // Информационно-компьютерные технологии в экономике, образовании и социальной сфере: тезисы докладов IX научно-практической конференции Симферополь. – 2014. – Вып. 9. – С. 95–98.
177. Шкарбан, Ф.В. Компоненты компетенции бакалавров прикладной информатики в области объектно-ориентированного программирования // Современные проблемы науки и образования. – 2018. – №1. – URL: http://www.science-education.ru/ article/view?id=27387.
178. Шкарбан, Ф.В. Методика преподавания дисциплины «Программирование для начинающих» / Ф.В. Шкарбан // Перспективы: сборник научных трудов молодых ученых. Выпуск 2. – 2011. – С. 292–302.
179. Шкарбан, Ф.В. Модель подготовки инженера-программиста / Ф.В. Шкарбан, Ф.С. Ильясова // Вестник Черниговского национального педагогического университета им. Т. Шевченко. – 2011. – Вып. 3. – С. 114–118.

180. Шкарбан, Ф.В. Некоторые аспекты обучения программированию / Ф.В. Шкарбан // Информационно-компьютерные технологии в экономике, образовании и социальной сфере: тезисы докладов VIII научно-практической конференции Симферополь. – 2013. – Вып. 8. – С. 160–161.
181. Шкарбан, Ф.В. «Низкий старт» в обучении программированию студентов компьютерных специальностей / Ф.В. Шкарбан, Л.М. Меджитова // Научный журнал НПУ им. Драгоманова. Серия №2. Компьютерно-ориентированные системы обучения. – 2012. – № 12 (19). – С. 76–80.
182. Шкарбан, Ф.В. Обучение бакалавров прикладной информатики основам объектно-ориентированного программирования: описание методики с использованием визуальных учебных сред / Ф.В. Шкарбан // Известия южного федерального университета. – 2018. – № 5. – С. 62–70.
183. Шкарбан, Ф.В. Обучение программированию бакалавров прикладной информатики: повышение качества подготовки на основе требований профессиональных стандартов / Ф.В. Шкарбан // Электронное обучение в непрерывном образовании 2018: сборник научных трудов V Международной научно-практической конференции (Россия, Ульяновск, 18–20 апреля 2018г.). – Ульяновск. – 2018. – С. 336–345.
184. Шкарбан, Ф.В. Система целей обучения программированию / Ф.В. Шкарбан // Информационно-компьютерные технологии в экономике, образовании и социальной сфере: сборник трудов научно-практической конференции. – Симферополь. – 2016. – Выпуск 10. – С. 107–110.
185. Шкарбан, Ф.В. Обучающая среда программирования Scratch / Ф.В. Шкарбан, Э. Уразалиева // Информационно-компьютерные технологии в экономике, образовании и социальной сфере: сборник

трудов научно-практической конференции. – Симферополь. – 2017. – Выпуск №1 (15). – С. 67–74.

186. Шкарбан, Ф.В. Обучающая среда программирования: Alice / Ф.В. Шкарбан, А.Э. Мевлют // Информационно-компьютерные технологии в экономике, образовании и социальной сфере: сборник трудов научно-практической конференции. – Симферополь. – 2017. – Выпуск №1(15). – С. 54–61.
187. Шкарбан, Ф.В. Обучение программированию в системе подготовки бакалавров прикладной информатики / Ф.В. Шкарбан // Исследования и разработки в перспективных научных областях: сборник материалов II Международной научно-практической конференции. – Новосибирск: Издательство ЦРНС, 2017. – С. 37–45.
188. Шкарбан, Ф.В. Обучение программированию средствами Scratch / Ф.В. Шкарбан, С.С. Танишева // Информационно-компьютерные технологии в экономике, образовании и социальной сфере: сборник трудов научно-практической конференции. – Симферополь. – 2017. – Выпуск №1(15). –С. 61–67.
189. Шкарбан, Ф.В. Объектно-ориентированное моделирование с помощью программы «Alice» / Ф.В. Шкарбан, Ф.С. Ильясова // Ежемесячный научно-педагогический журнал «Молодежь и рынок». – 2011. – №8 (79). – С. 147–150.
190. Шкарбан, Ф.В. ООП подход для студентов первокурсников компьютерных специальностей / Ф.В. Шкарбан // Научный журнал НПУ имени М.П. Драгоманова. Серия №2. Компьютерно-ориентированные системы обучения. – 2010. – №8 (15). – С. 151–154.
191. Шкарбан, Ф.В. Организация преподавания курса «Программирование для начинающих» / Ф.В. Шкарбан // Информационные технологии в образовании, науке и технике» (ИТОНТ 2010), Черкассы. – 2010. – Т.2. – С. 79.

192. Шкарбан, Ф.В. Особенности обучения дисциплин цикла компьютерных наук в вузах России и за рубежом / Ф.В. Шкарбан // Информационно-компьютерные технологии в экономике, образовании и социальной сфере: сборник трудов научно-практической конференции. – Симферополь. – 2016. – Выпуск №3(13). – С. 129–136.
193. Шкарбан, Ф.В. Особенности практической подготовки инженеров-программистов / Ф.В. Шкарбан // Информационно-компьютерные технологии в экономике, образовании и социальной сфере: сборник трудов научно-практической конференции. – Симферополь. – 2016. – Выпуск №2(12). – С. 89–104.
194. Шкарбан, Ф.В. Особенности преподавания основ программирования в контексте формирования базовых знаний направления подготовки «Информатика» / Ф.В. Шкарбан, Л.М. Меджитова // Высшая школа. – 2013. – №1. – С. 27–35.
195. Шкарбан, Ф.В. Программирование для начинающих: Alice and Scratch: Учебное пособие / Ф.В. Шкарбан. – Симферополь: ИП Хотеева Л.П., 2017. – 120с.
196. Шкарбан, Ф.В. Программирование для начинающих: ALICE. Учебное пособие / Ф.В. Шкарбан. – Симферополь: ФЛП Куртбединова Д.А., 2013. - 116с.
197. Шкарбан, Ф.В. Разработка методики обучения основам объектно-ориентированного программирования с использованием визуальных учебных сред / Ф.В. Шкарбан // Ученые записки Крымского федерального университета им. В.И. Вернадского. Социология.Педагогика. Психология. – Том 4 (70) №2. – 2018.– С.52-58.
198. Шкарбан, Ф.В. Содержание и этапы формирования компетенции бакалавров прикладной информатики в области объектно-ориентированного / Ф.В. Шкарбан // Известия Волгоградского

государственного педагогического университета. – 2018. – №1 (124). – С. 35–41.

199. Шкарбан, Ф.В. Уровни сформированности компетенции бакалавров прикладной информатики в области объектно-ориентированного программирования / Ф.В. Шкарбан // Приоритетные направления развития современного образования: сборник статей II Межрегиональной научно-практической конференции. – Астрахань: Астраханский государственный университет, Издательский дом «Астраханский университет», 2018. – С. 136–141.
200. Шкарбан, Ф.В. Формирование мотивации к программированию при подготовке студентов компьютерных специальностей / Ф.В. Шкарбан // Информационно-компьютерные технологии в экономике, образовании и социальной сфере: тезисы докладов VII научно-практической конференции Симферополь. – 2012. – Вып. 7. – С. 78–80.
201. Шкарбан, Ф.В. Формирование объектно-ориентированного мышления у студентов компьютерных специальностей / Ф.В. Шкарбан, Ф.С. Ильясова // Современные информационные технологии и инновационные методики обучения в подготовке специалистов: методология, теория, опыт, проблемы. – 2010. – С. 356–361.
202. Шнейдерман, Б. Психология программирования: Человеческие факторы в вычислительных и информационных системах / Б. Шнейдерман [пер. с англ.]. – М. : Радио и связь, 1984. – 304 с.
203. Эльконин, Б.Д. Понятие компетентности с позиции развивающего обучения / Б.Д. Эльконин // Современные походы к компетентностно-ориентированному образованию. – Красноярск, 2002. – С. 24–30.

204. Adams, J. Alice 3 in Action: Computing Through Animation / J. Adams [Electronic resource]. URL: http://www.alice.org/ index.php?page=documentation.
205. Classification of Instructional Programs (CIP 2016) // Computer and information sciences and support services [Electronic resource]. URL: http://www23.statcan.gc.ca/imdb/p3VD.pl?Function=getVD&TVD=299355&CVD=299356&CPV=11.&CST=01012016&CLV=1&MLV=3.
206. Classification of Instructional Programs (CIP) Canada // Statistics Canada – catalogue no. 12-590-X [Electronic resource]. URL: http://www.statcan.gc.ca/pub/12-590-x/12-590-x2012001-eng.pdf.
207. Compare Languages: Monthly Commits // Ohloh — public directory of Free and Open Source Software [Electronic resource]. URL: http://www.ohloh.net/languages/compare.
208. Computer Science Curriculum 2013. Curriculum Guidelines for Undergraduate Degree Programs in Computer Science [Electronic resource]. URL: https://www.acm.org/binaries/content/assets/education/ cs2013_web_final.pdf.
209. Computing Curricula 2016 [Electronic resource]. URL: https://www.computer.org/cms/Computer.org/professional-education /curricula/ComputerEngineeringCurricula2016.pdf.
210. Curriculum Development Guidelines New ICT curricula for the 21st century: designing tomorrow’s education [Electronic resource]. URL: http://people.ac.upc.edu/toni/papers/CurrITEng.PDF.
211. Cutts, Q. Expeditions through Alice / Q. Cutts, S. Esper, B. Simon [Electronic resource] URL: http://www.alice.org/index.php? page=documen/tation.
212. Dann, W. Learning to Program with Alice / W.Dann, St. Cooper, R. Pausch [Electronic resource]. URL: http://www.alice.org/ index.php?page=documentation.

213. Gaddis, T. Starting Out with Alice: A Visual Introduction to Programming, 2/E / T. Gaddis [Electronic resource]. URL: http://www.alice.org/index.php?page=documentation.
214. Hundt, R. Loop Recognition in C++/Java/Go/Scala / R. Hundt [Electronic resource]. URL: https://days2011.scala-lang.org/sites/days2011/files/ws3-1-Hundt.pdf.
215. Majed, M. Learn to Program with Scratch / M. Majed [Electronic resource]. San Francisco, California: No Starch Press. – 2014. – pp. 1–9, 13–15.
216. Maxwell, J.W. Tracing the Dynabook: A Study of Technocultural Transformations: PhD Dissertation / John W. Maxwell; The University of British Columbia. – Vancouver, 2006. – VIII+303 p.
217. Molina Gómez, J.D. Basic concepts of Prezi for making presentations/ J.D. Molina Gómez // FMC Formacion Medica Continuada en Atencion Primaria. – Issue 1. – 2014. P. 3 –16.
218. Moskal, B. Evaluating the effectiveness of a new instructional approach / B. Moskal, D. Lurie, S. B. Cooper //ACM SIGCSE Bulletin. – ACM, 2004. – T. 36. – №. 1. – P. 75–79. https://www.researchgate.net/publication/221538824_Evaluating_the_effectiveness_of_a_new_instructional_approach.
219. National Center for Education Statistics // Classification of Instructional Programs (CIP) [Electronic resource]. URL: https://nces.ed.gov/ipeds/cipcode/.
220. Resnick, M. Scratch: Programming for All / M. Resnick, J. Maloney, A. Monroy Hernandez, N. Rusk, E. Eastmond, K. Brennan, A.Millner, E. Rosenbaum, J. Silver, B. Silverman, Y. Kafai // Communications of the ACM. – 2009. – Vol. 52, No. 11. – P. 60–67.
221. Standardization training program E-40 discipline: Software Engineering [Electronic resource]. URL: http://www.ecss.nl/wp-

content/uploads/2016/09/ECSS-Training-L2-E40-Software-v7(2017-03-15).pdf.

222. Teaching, Learning, and Sharing: How Today's Higher Education Faculty Use Social Media [Electronic resource]. URL: http://eric.ed.gov/?id=ED535130.
223. The history of the European Union [Electronic resource]. URL: https://europa.eu/european-union/about-eu/history_en.
224. TIOBE Programming Community Index for July 2016[Electronic resource]. URL: http://www.tiobe.com/index.php/content/paperinfo/tpci/index.html.

ПРИЛОЖЕНИЯ

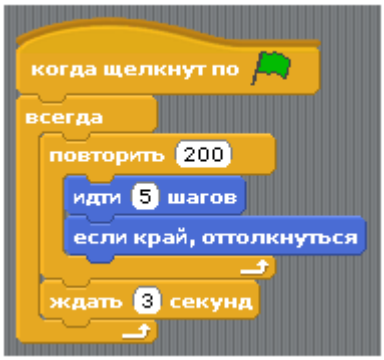
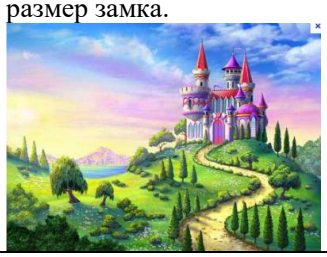
Приложение 1

Примеры заданий-упражнений для определения уровня усвоения учебного материала студентами во время лабораторных работ по дисциплинам «Программирование для начинающих» и «Информатика и программирование»

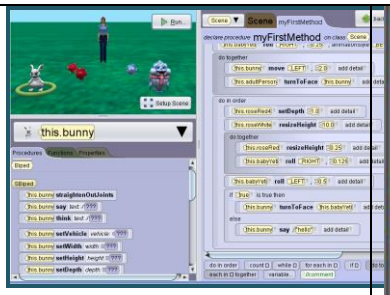
	Дисциплина «Программирование для начинающих»		Дисциплина «Информатика и программирование»
	Программная среда «Alice»	**Программная среда «Scratch»**	**Язык программирования С++**

1.	Напишите алгоритм в словесной форме и на псевдокоде для решения следующей задачи перехода пешехода через дорогу при отсутствии светофора.		Что будет выведено на экран в результате выполнения следующего кода:
			classABase {
			public:
			void f(inti) const { cout<< 1;}
			void f(charch) const { cout<< 2; }
			};
			
			classBBase {
			public:
			void f(double d) const { cout<< 3;}
			};
			classABBase : publicABase, public BBase {
			public:
			usingABase::f;
			usingBBase::f;
			void f(charch) const { cout<< 4; }
			};
			
			void g(ABBase&ab) {
			ab.f('c');
			ab.f(2.5);
			ab.f(4);
			}
			int main() {
			ABBaseab;
			g(ab);
			}

2.	1. Открыть пример, созданный в среде Alice (примеры см. на сайте http: // www.alice.org). 2. Ознакомиться с его структурой. 3. Подготовить отчет, в котором словесно описать структуру программы, в соответствии с выбранным примером. Разбить программу на основные составляющие и правильно их описать.	1. Открыть пример, созданный в среде Scratch (примеры см. на сайте (http://scratch.mit.edu). 2. Ознакомиться с его структурой. 3. Подготовить отчет, в котором словесно описать структуру программы, в соответствии с выбранным примером. Разбить программу на основные составляющие и правильно их описать.	Используя документ Паспорт гражданина России представить его описание с помощью декомпозиции на простейшие объекты-атрибуты.

3.	Что выполняет следующая конструкция: *DoInOrder (* *bunny. Move (Forward, 2),* *bunny. Move (Back, 2))*	1. Какие команды содержит цикл *повтори*? 2. Какие команды содержит цикл *всегда*?	Для реализации программного кода создать два класса: матрица и вектор. В каждом из классов задать конструкторы и деструкторы.
4.	*def Chase ():* *if Fish. DistanceTo (cat)> 2:* *DoInOrder (* *Fish. PointAt (cat),* *Fish. Move (Forward),* *Alice. SetAlarm (2, do (Chase)))* Этот код проверяет, находится объект «Рыба» достаточно близко к объекту «Кошка». Если нет, то объект «Рыба перемещается на 1 единицу к объекту «Кошка», и установленный SetAlarm повторит весь процесс. Найдите ошибки в фрагменте программы.	 Составить программу и объяснить как она работает.	Реализовать программный класс Person. В данном классе объявить закрытые переменные, которые сохраняют следующие данные: фамилия сотрудника, заработная плата, премии, ставка.
5.	Используя библиотеку объектов, их стили и свойства реализовать следующую картинку. Подготовить отчет о проделанной работе. 	Создать объект средствами любого графического редактора и экспортировать его в библиотеку Scratch.	Создать класс Student, содержащий следующие поля: - name - Фамилия и инициалы; - year - год рождения: - bal - оценки из 4 предметов (массив из 4 элементов)
6.	В представленном сюжете сделать так, чтобы деревья в определенный интервал времени провалились под землю и вернулись назад. С помощью соответствующей процедуры увеличить размер замка. 	Программно реализовать алгоритм перехода объекта из комнаты 1 (кухня) в комнату 2 (спальня).	Имеется прототип конструктора: myclass(int, char, char*); Объявить массив из четырех объектов.
7.	Используя функции к внутренним составным частям объектов, заставьте «объект» моргать глазами (левый глаз, потом правый глаз)..	1. Создать собственный объект, нарисуйте к нему несколько костюмов. 2. Составить скрипт с использованием смены костюмов.	Программно реализовать конструктор следующего класса таким образом, чтобы можно было создать неинициализированные объекты. При этом, создавая объекты присвоить переменным x и y

			значение 0:
			class myclass { int x,y;
			public:
			myclass(int i, int j) {
			x = i;
			y = j;
			}
			... // другие функции класса
			};

8.	Реализовать программу «Аквариум».		Создать класс для динамической структуры, которую называют *очередь*. Структура содержит два информационных поля: фамилия студента и экзаменационный балл по информатике. Класс будет иметь такие методы: конструктор, приобщение элемента в очереди, изъятия элемента из очереди, просмотра элементов очереди.
9.	 Есть ли в данной программе разветвления?	 Есть ли в данной программе цикл?	Составить программу с использованием класса *Smallobj* в консольном режиме.
10.	Реализовать программу, используя все рассмотренные конструкции: Do in order, Do Together, If / Else, While, // Comment.	Составить программу роста цветов (скорость роста у всех цветов разная).	1. Реализовать программно класс, реализующий работу стека. 2. Разработать и реализовать программу, использующую класс стека для моделирования Т-образного сортировочного узла на железной дороге: 2.1. В программе следует реализовать разделение поезда, который состоит из вагонов двух типов, на два направления (на каждое направление формируется поезд из вагонов одного типа). 2.2. Реализовать в программе возможности формирования железнодорожного состава пользователем с клавиатуры и случайным образом к предыдущей программе.

Приложение 2

Примеры заданий, направленных на усвоение учебного материала по дисциплинам «Программирование для начинающих» и «Информатика и программирование»

Дисциплина «Программирование для начинающих»			Дисциплина «Информатика и программирование»
№	**Программная среда «Alice»**	**Программная среда «Scratch»**	**Язык программирования С++**
1.	Составить программу перехода пешехода через дорогу при отсутствии светофора.	Создать программу – анимацию танцующего мальчика под музыку.	Заданы сведения о книгах: автор, название, издательство, год издания, количество страниц, шифр. Напишите программу, которая по команде пользователя будет выполнять одно из следующих действий: 1. Вводить сведения о новой книге. 2. Выводить в файл сведения о заданной книге (по шифру). 3. Сортировать множество книг по шифру. 4. Проверять и, если необходимо, редактировать записи так, чтобы автор и название книги были написано с большой буквы. 5. Выводить на экран сведения о первых n книгах.
2.	Создать мир, и программно реализовать: дельфин (Dolphin) плавает в океане. На некотором расстоянии летает птица (Falcon). Через некоторое время птица возвращается к дельфину, а затем летит от него и снова к нему (на некотором расстоянии).	Реализовать программно следующий алгоритм: Объект Кот издалека подходит к мячу, а затем ударяет по нему и мяч улетает. Объект Кот кричит: «Yes!». Применить эффекты приближения и удаления.	Реализовать иерархию классов Стек. Описать абстрактный класс Стек, определив операции Добавить, Удалить, Пустой как виртуальные.
3.	Составить программу, которая реализует элементы фигурного катания. При создании программы сделать объяснения отдельных фрагментов программы в виде комментариев.	Составьте программу, которая спрашивала бы у пользователя, на сколько процентов увеличить или уменьшить размеры произвольного объекта. После чего изменяла бы размер объекта на холсте.	Создать класс, который описывает и обеспечивает действия над данными параметризованного массива, размерность которого определяется во время работы программы. Все вычисления и преобразования реализовать в виде функций-членов класса. В массиве вычислить: номер элемента массива, ближайшего к среднеарифметического его значений; разницу элементов массива, расположенных между

			первым отрицательным и вторым положительным элементами. Преобразовать массив так, чтобы в его первой половине располагались элементы, расположенные в четных позициях, а во второй - элементы, расположенные в нечетных позициях.
4.	Создайте проект, в котором девочка под музыку выполняет некоторые танцевальные движения. Подключение звукового сопровождения – процедура PlaySound.	Написать программу: Компьютер случайным образом формирует число в пределах от 0 до 100, а игрок угадывает его. На предложения игрока компьютер сообщает: «Мало», «Много» или «Вы угадали !!!» в зависимости от взаимного расположения числа-догадки и случайного числа.	С клавиатуры вводится текстовая строка. Составить программу, которая подсчитывает количество слов, которые имеют нечетную длину; выводит на экран частоту каждой буквы.
5.	Составить программу «Спортсмен прыгает на скакалке 10 раз».	Добавьте новый объект в проект. Данный объект появляется в центре холста, после того как предыдущий закончит выполнять определенные команды. Новый объект должен сначала постепенно уменьшаться, а затем постепенно увеличиваться до прежних размеров.	Вводится с клавиатуры время в формате мин: сек, например 12: 11. Записать настоящее время текстом: Двенадцать минут одиннадцать секунд. Перед выводом результат поместить в объект типа string.
6.	Составить программу реализации трех видов циклического алгоритма.	Составить программу реализации трех видов циклического алгоритма.	С текстового файла считать натуральные числа в список. В очередь записать квадраты чисел, которые больше по первое число. Вывести на экран образовавшуюся очередь.
7.	 Подключить к рабочей сцене объект Camera	Составить программу, при выполнении которой пользователь может управлять объектом с помощью стрелок на клавиатуре (вверх, вниз), а при нажатии клавиши пробел, объект «говорит» фразу «I am tired».	Реализовать класс BOOK, который содержит следующие поля: - Name - название книги; - Avtor - авторы книги; - Data -дата печати; - Cost - стоимость книги. 2. Написать программу, которая использует данный класс и выполняет следующие действия - вводит с клавиатуры массив

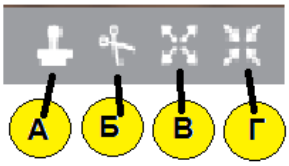
			данных SHOP, состоящий из N переменных типа BOOK; - выводит на экран все книги, которые были напечатаны в заданном году. 3. Программу создать в трех файлах: заготовочные файл * .h с описанием класса, файл с функциями класса * .cpp, и главная функция main.cpp. Доступ к полям класса выполнить через методы класса.
8.	Составить программу, в которой будут происходить определенные действия (по выбору студента) и подключить три разных звуковых сопровождения.	Составить программу «Оркестр». Подключить соответствующее звуковое сопровождение	Реализовать программно иерархию классов для работы со счетом вкладчика банка. Для этого следует выполнить: 1. Реализовать программно базовый класс Rahunok. 2. Реализовать программно базовый класс Vkladnyk. 3. Реализовать программно производный класс RahunokVkladnyka. 4. В главной части программы продемонстрировать работу созданной иерархии классов.
9.	Составить программу: Запуск сцены начинается со звукового сопровождения и произвольных действий объекта Camera. Далее на сцене появляется объект Bird. Программно реализовать природный полет данного объекта.	Реализовать программу, в которой будут использованы случайные числа.	Реализовать программное приложение начисления заработной платы сотрудникам фирмы. Для хранения данных о сотрудниках использовать массивы объектов, а для доступа к функциям – указатели.
10.	Студент формулирует условие задачи согласно следующим этапам: • разработка сценария (постановка задачи); • проектирование ролика; • моделирование сцен; • анимация сцен; • визуализация отдельных эпизодов; • конечный аудио- и видеомонтаж ролика. Объем задания рассчитан на 2 академических часа.		Каждый студент формулирует условие задачи для своего сокурсника. Студенты обмениваются заданиями и выполняют их. Условие задачи должно включать использование структурного типа, указателей и определение функций. Исходный код должен быть распределен как минимум на три файла: основной, файл с определениями функций. Заголовочный файл. Объем задания рассчитан на 2 академических часа.

Приложение 3

Примеры тестовых вопросов, предложенных студентами

1. Программа в среде Alice представляет собой:
   а) специальный набор команд (действий) для решения определенной задачи;
   б) один из компонентов программного обеспечения;
   в) предварительное описание будущих событий или действий
2. Программа в Alice состоит из:
   а) функций, переменных, параметров и рекурсий;
   б) variables;
   в) procedures
3. Организация цикла в Alice осуществляется посредством:
   а) одной из двух операций: loop или while;
   б) if / else;
   в) procedures

1. В среде Scratch можно создать программу:
   а) Изменив готовый проект;
   б) Создав новый проект
2. Чтобы сохранить проект в среде Scratch с новым именем нужно выбрать:
   а) Файл - Импортировать проект;
   б) Файл - Сохранить как;
   в) Файл – Новый
3. Увеличить размер изображения в среде Scratch можно с помощью инструмента:



1. Класс (class) – это сложный тип данных, который определяется пользователем и состоит:
   а) из элементов-данных (переменных) и функциональных данных (функций, методов), которые реализуют операции с ними;
   б) переменных;
   в) функций и методов
2. Объект – это
   а) переменная типа «класс»;
   б) функция;
   в) сложный тип данных
3. Для создания объекта используется имя класса как спецификатор типа данных:
   а) имя_класса список_объектов; ;
   б) список_объектов имя_класса; ;
   в) имя_класса{список_объектов}

Приложение 4

Примеры заданий-исследований на подготовку докладов по дисциплинам «Программирование для начинающих» и «Информатика и программирование»

Дисциплина «Программирование для начинающих»		**Дисциплина «Информатика и программирование»**
Программная среда «Alice»	**Программная среда «Scratch»**	**Язык программирования С++**
Подготовьте доклад по предложенной теме. Примерные темы докладов: 1. Методы (процедуры и функции) в Alice. 2.Встроенные процедуры в Alice. 3. Представление классов в Alice.	Подготовьте доклад по предложенной теме. Примерные темы докладов: 1. Элементы объектно-ориентированного программирования в Scratch. 2. Объекты в Scratch. 3. основные понятия Scratch.	Подготовьте доклад по предложенной теме. Примерные темы докладов: 1. Объектно-ориентированное программирование. 2. Использование классов при разработке программ. 3. Перезагрузка операций.

Приложение 5

Примеры заданий по разработке программного продукта по дисциплинам «Программирование для начинающих» и «Информатика и программирование»

Дисциплина «Программирование для начинающих»		**Дисциплина «Информатика и программирование»**
Программная среда «Alice»	**Программная среда «Scratch»**	**Язык программирования С++**
Разработка программного продукта		
Создать программу – проект, с помощью которой реализуется фрагмент сказки «Алиса в стране чудес»	Создать программу – проект, с помощью которого реализуется игра «Арканоид».	Создать программу – проект, с помощью которого реализуется игра «Судоку».
Создать программу – проект, с помощью которой реализуется фрагмент любимой сказки.	Создать программу – проект, с помощью которого реализуется игра «Лабиринт».	Создать программу – проект, с помощью которого реализуется игра «Лабиринт».
Создать программу – проект, с помощью которого реализуется игра «Марио»	Создать программу – проект, с помощью которого реализуется игра «Марио».	Создать программу – проект, с помощью которого реализуется игра «Марио».

Приложение 6

Примеры лабораторных работ

ПЛАН
ЛАБОРАТОРНОЙ РАБОТЫ № 2

По дисциплине: «Программирование для начинающих»

Тема работы: Знакомство с интерфейсом Alice

Цель работы: научиться запускать программу Alice, ознакомиться с интерфейсом и основными принципами использования среды «Alice».

Количество часов – 2часа.

Содержание работы:

1. Ознакомиться со структурой программной среды Alice.
2. Рассмотреть и проанализировать использование:

встроенных команд-действий;

инструкций;

функций;

результатов;

рекурсии / цикла;

событий / взаимодействий;

программы и языковых конструкций.

Теоретические сведения:

Рассмотрим принципы использования среды «Alice» в процессе знакомства с объектно-ориентированным программированием. Alice – это свободная и открытая объектно-ориентированная среда программирования, направленная на обучение с интегрированной средой разработки (IDE), что позволяет создавать трехмерную анимацию. Она реализована средствами Java и Python. Alice использует методы drag-and-drop для создания компьютерной анимации с использованием 3D-моделей [5].

Мир программной среды Alice состоит из объектов, а те в свою очередь делятся на составляющие объекта (рис. 2.1).

Рис. 2.1. Библиотека объектов

В процессе работы необходимо собрать из представленных блоков трехмерный мир, который «заселяют» объекты (животные, люди, растения и т.д.). Программа Alice состоит из функций, переменных, параметров и рекурсий, которые записываются в единую программу (рис. 2.2).

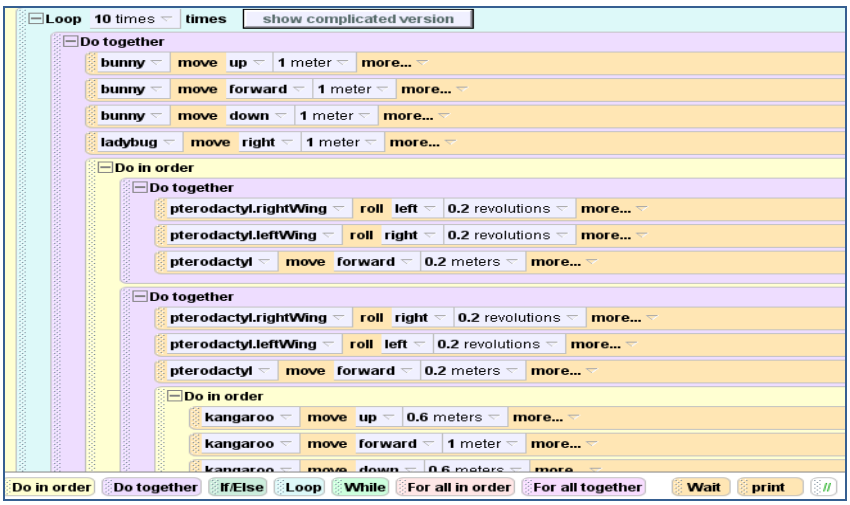


Рис.2.2. Структура программы в Alice

Работа в данной среде позволяет развивать интуитивное понимание основных концепций программирования. Таким образом, в процессе работы с «Alice» можно сразу увидеть, как их анимационные программы работают, а выполненные действия позволяют понять фактическое функционирование различных программ (рис.2.3).

Рис.2.3. Окно просмотра результата (запуск программы)

Как и любой программный продукт Alice имеет свою структуру [6]:

1. Команды-действия:

Alice содержит несколько встроенных команд-действий. Действия делятся на категории:

изменение движений объекта (например, Move, Turn и Roll, PointAt).

изменение физической природы объекта (например, Destroy, AddObject, Resize, Hide и Show).

2. Инструкции:

Alice позволяет использовать определенные последовательности инструкций. Инструкции весьма похожи на процедуры многих языков программирования. Следующая инструкция позволяет «заставить» объект выполнять определенное действие:

DoInOrder (Hop, Hop)

3. Функции:

В среде Alice функции поддерживаются с помощью базового языка Python. Функции используются для реализации рекурсии, цикла, а также выполнения событий и обработки вычислений:

def howmany (dist, diam):

*return dist / (3.14 * diam)*

4. Результаты:

Как и функции, результаты получаются с помощью базового языка Python. В связи с визуальной обратной связью в Alice, студенты имеют возможность сразу увидеть полученный результат:

if cat. DistanceTo (Fish) <1.0:
DoInOrder (
cat. Move (Up, 0.5),
cat. Move (Down, 0.5))

5. Рекурсия /Разветвления / Цикл:

Alice обеспечивает поддержку повторений через инструкцию-петлю (Loop). Например, если записать инструкцию Loop (Hop, 5), то получим результат - объект «Кролик» будет прыгать пять раз.

Также в Alice является возможность построения традиционного цикла с помощью инструкции - Do (действия, EachFrame) конструкции)).

def Chase ():
if Fish. DistanceTo (cat)> 2:
DoInOrder (
Fish. PointAt (cat),
Fish. Move (Forward, 1),
Alice. SetAlarm (2, do (Chase)))

Этот код проверяет, находится ли объект Fish достаточно близко к объекту Cat. Если нет, то объект Fish перемещается на 1 единицу к объекту Cat, а установленный SetAlarm повторяет весь процесс.

6. События / Взаимодействия:

Alice обеспечивает поддержку обработки событий для создания графических интерфейсов панели управления, списков, флажков. События позволяют пользователю взаимодействовать с анимированным миром.

7. Программы и языковые конструкции:

Alice содержит встроенные конструкции DoTogether и DoInOrder, которые позволяют начинающему программисту выбрать одновременное или последовательное действие объекта.

Например, использование следующей конструкции позволит выполнить действие последовательно (сначала вперед, потом назад):

DoInOrder (
bunny. Move (Forward, 2),
bunny. Move (Back, 2))

Для записи программы в область кода программы перетаскиваем встроенные программные элементы, в частности, условные ветвления, циклы – do ... while и for, а также операторы ожидания wait, вывода текста print и комментарии comment. Можно даже задавать сложные конструкции параллельного программирования, используя элемент Do Together.

Строя программирование на таком простом и интересном уровне Alice помогает ознакомиться со структурой и логикой программирования, избавляя от путаницы в синтаксисе. В процессе составления кода программы больше внимания уделяется объектам и другим возможностям объектно-ориентированного программирования.

Alice делает процесс программирования захватывающим, во многом благодаря сокрытию синтаксиса. С помощью дополнительных настроек в Alice можно изменить режим отображения кода программы с языка Alice на Java (рис. 2.4).

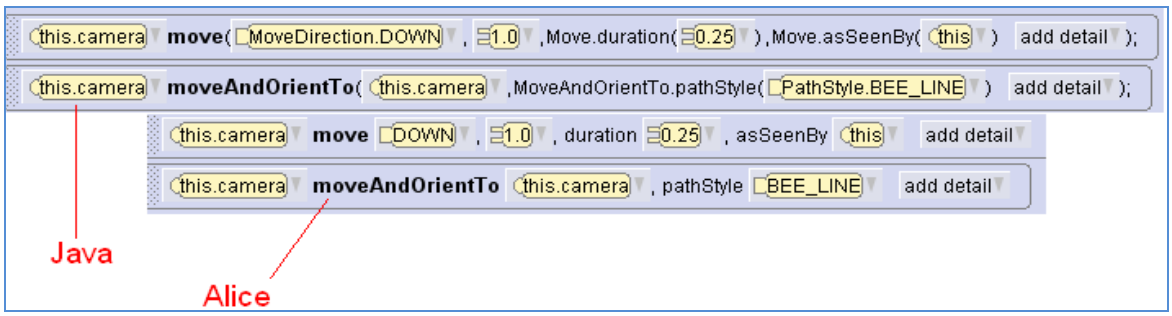


Рис. 2.4. Изменение отображения кода программы

Задание:

1. Открыть пример, созданный в среде Alice (примеры см. на сайте http: // www.alice.org).

2. Ознакомиться с его структурой.

3. Подготовить отчет, в котором представить и описать структуру программы, в соответствии с выбранным примером. Разбить программу на основные составляющие и соответственно их описать.

Отчет должен содержать:

1. Тему и цель лабораторной работы.

2. Перечень этапов выполнения заданий.

3. Краткое описание составляющих программы в среде Alice.

4. Выводы.

Рекомендуемая литература:

Сайт проекта Alice. URL: http:// www.alice.org (дата обращения 22.10 2016).

Professor Developed Alice Animation Tool to Teach Computer Programming. URL: http://www.webwire.com/ViewPressRel.asp?aId=54742
(дата обращения 22.10.2016).

Dann W.P., Cooper S., Pausch R. Learning To Program with Alice.- Edition 2nd. Prentice Hall, 2009. 384 p.

Контрольные вопросы:

1. Что такое встроенные команды-действия?

2. Как в программной среде Alice просмотреть результат собственной работы? (Ответ объяснить).

3. Можно ли в Alice реализовать рекурсию / цикл?

4. Что позволяет реализовать следующая конструкция?

DoInOrder (
bunny. Move (Forward, 2),
bunny. Move (Back 2))

5. Обоснуйте структуру программы, реализованную средствами Alice.

Самостоятельная работа:

Рассмотреть и проанализировать использование главного меню и его составляющих: File, Edit, Project, Run, Window, Help.

ПЛАН
ЛАБОРАТОРНОЙ РАБОТЫ № 3

По дисциплине: «Информатика и программирование»

Тема занятия: Проектирование классов и создание объектов. Реализация программы.

Цель занятия: усвоение понятия классов и объектов; приобретение практических навыков их объявления и использования.

Количество часов: 2

Содержание работы (Задание, Задачи):

1. Классы и объекты Механизм, который объединяет данные и код, манипулирует этими данными, и защищает то и другое от внешнего втручанння или неправильного использования, называется инкапсуляцией (incapsulation). В С ++ инкапсуляция поддерживается, в частности, благодаря использованию классов и объектов. Класс (class) - это сложный тип данных, определяется пользователем. Он может состоять как из элементов-данных (переменных), так и функциональных данных (функций, методов), которые реализуют операции с ними. Функции и переменные, объявленные внутри класса, члены (members) этого класса. Функции, объявленные внутри класса, называют функциями-членами (member functions) или методами. Члены класса могут быть закрытыми (private) или открытыми (public). Для объявления закрытых членов класса используется ключевое слово private. Закрытые члены класса доступны для других членов этого класса, но не достижимые за этим классом. Для объявления открытых членов класса используется ключевое слово public. К открытым членам класса имеют доступ как члены этого класса, так и любая другая часть программы. По умолчанию (by default) члены класса являются закрытыми.

Синтаксис описания класса (по умолчанию члены класса являются закрытыми, поэтому ключевое слово private необязательно в объявлении класса; списку_обьектив тоже может не быть или он будет состоять из одного или нескольких объектов):

```
class имя_класса
{
  private:
  ... //объявление закрытых функций и  переменных класса
  public:
  ... // объявление открытых функций и переменных класса
 } список_объектов;
```

Здесь имя_класса определяется согласно правилам написания идентификаторов в С ++: имя может начинаться с символов A..Z, a..z или со знака подчеркивания и содержать любые буквы и знак подчеркивания. Точка с запятой является обязательным в конце объявления класса. Например, объявим класс myclass:

```
class myclass {
  int x, y; //объявление закрытых  переменных класса
  public: // объявление открытых функций класса
  void show();
  int getx();
  int gety();
 }; //конец объявления класса
```

Для определения функций-членов класса следует связать имя этого класса с именем этого метода (функция может быть без параметров или иметь один или несколько параметров):

```
тип_данных_что_возвращается_функцией  имя_класса::
                        имя_функции (параметры_функции)
{
... // тело функции
}
```

Двойной двоеточием определяется операция расширения диапазона видимости (scope resolution operator).

Объект - это переменная типа "класс". объявление класса определяет только тип объекта, который будет создан во время его фактического объявления. Для создания объекта используется имя класса как спецификатор типа данных:

имя_класса список_обьектов;

Здесь список_объектов может состоять как из одного имени объекта, так и из нескольких, разделенных запятыми.

Например, для класса myclass, объекты ob1, ob2 будут объявлены так:

myclass ob1, ob2;

Объявление класса является логическим абстракцией, которая задает новый тип данных для объекта, а объявления объекта создает физическую сущность объекта такого типа (то есть в случае объявления объекта выделяется память, а в случае объявления класса - не выделяется).

Доступ к открытым членам класса можно получить только через объект этого класса. Для доступа к открытым членам класса используют операцию доступа "точка" (.):

имя_объекта.имя_функции (параметри_функции)

Для инициализации объектов класса используют конструкторы.

2. Конструкторы и деструкторы

Конструктор (constructor function) - функция-член класса, которая включается в описание класса и имеет то же имя, что и класс. Конструктор создает значение типа "класс" (то есть объект класса). Этот процесс включает инициализацию переменных класса и часто распределение свободной памяти. Конструктор класса вызывается автоматически каждый раз при создании объекта. Конструктор не может иметь возвращаемого значения. Объявления конструктора:

```
class имя_класса
{   ... // объявление закрытых членов класса
  public:
    ... // объявление открытых членов класса
    имя_класса (список_параметров); //конструктор
 };
```

Для глобальных объектов конструктор вызывается на начала выполнения программы. Для локальных объектов конструктор вызывается каждый раз при объявлении объекта. Функция, обратная к конструктору, является деструктором (destructor). Эта функция вызывается в случае уничтожения объектов. Локальные объекты уничтожаются тогда, когда они выходят из диапазона видимости. Глобальные объекты уничтожаются по завершению программы. Деструктор задается символом ~ (Тильда) с таким именем класса:

```
class имя_класса
{     ... // объявление закрыых членов класса
  public:
```

```
    ... // объявление открытых членов класса
    ~ имя_класса (); //деструктор
};
```

Невозможно получить указатели на конструктор и деструктор. Конструктору можно передавать аргументы. Для этого просто добавляются необходимые параметры в объявление и определение конструктора. Затем во время объявления объекта конструктору следует передать аргументы:

имя_класса имя_объекта (список_аргументов);

Фактически синтаксис передачи аргументов конструктора с параметрами являются сокращенной формой записи более длинных выражения:

имя_класса имя_объекта = имя_класса (список_параметров)

В отличие от конструктора деструктор не может иметь параметров. Содержание этого понять достаточно просто: Нет механизма передачи аргументов объекта, который уничтожается.

Задание:

1. Создать класс TRAIN, содержащий следующие поля:
- Nazv – Название
- Numer - номер поезда;
- Date - дата отправления
- Time - время отправления поезда.

2. Написать программу, которая использует данный класс, а также предусмотреть возможность работы с произвольным числом записей, а с помощью отдельных функций класса реализовать:
- конструкторы без параметров и с параметрами;
- добавление объектов;
- уничтожение объектов;
- вывод информации на экран;
- поиск нужной информации по конкретному признаку;
- редактирование записей;
- сортировка по различным полям.

Отчет должен содержать:

1. Тему и цель лабораторной работы.
2. Перечень этапов выполнения заданий.
3. Листинг программы.
4. Выводы.

Рекомендуемая литература:

1. Элджер Д. C++: библиотека программиста. — СПб.: Питер, 2000 — 320 с.
2. Буч Г. Объектно-ориентированный анализ и проектирование с примерами на C++. - М.: Бином, 1998 - 560 с.
3. Страуструп Б. Язык программирования C++. -- СПб.: Бином, 1999 —.991 с.

Пособия и инструменты:

1. Microsoft Visual Studio 2012
2. Microsoft Visual Studio 2015
3. Microsoft Visual Studio Express

Вопросы для защиты лабораторной работы:

1. Понятие *класс, объект*
2. Синтаксис описания класса
3. Конструкторы и деструкторы

Приложение 7

Комплект тестовых заданий

дисциплина «Программирование для начинающих»

1. Уровневую систему можно разрабатывать как:
    1. Совокупность задач.
    2. Упорядоченную множество виртуальных пространств.
    3. Иерархическую абстракцию.
    4. Множество параллельных моделей.
2. Подсистемы нижнего уровня называют:
    1. Объектами.
    2. Классами.
    3. Моделями.
    4. Модулями.
3. При выборе алгоритма можно корректироваться:
    1. Вычислительной сложностью.
    2. Понятностью и легкостью реализации.
    3. Гибкостью.
    4. Структурой.
4. Топология программной системы определяется:
    1. Совокупностью объектов.
    2. Множеством моделей.
    3. Совокупностью потоков информации в системе.
    4. Централизованным управлением.
5. Для определения асинхронности объектов используют:
    1. Объектную модели.
    2. Динамическую модель.
    3. Аппаратные устройства.
    4. Программное обеспечение.

Комплект тестовых заданий

дисциплина «Информатика и программирование»

ЧАСТЬ 1

В заданиях с 1 по 11 выберите один или несколько правильных вариантов ответа. В заданиях с 12 по 15 напишите расширенный ответ.

1. Определите значение суммы двоичных чисел 101+110

(a) 1111; (c) 0111;

(b) 1011; (d) 1010;

2. Если двоичное число 1011 перевести в десятичную систему счисления, то получится число …

(a) 11; (c) 13;

(b) 14; (d) 6;

3. Если десятичное число 33 перевести в двоичную систему счисления, то получится число …

(a) 100001; (c) 100010;

(b) 101001; (d) 100011;

4. Какие идентификаторы из ниже перечисленных недопустимы в C++?

(a) int double = 3.456; (c) int number = 3;

(b) char 1_or_2 = ‘1’; (d) double real_num = 4.123;

5. Какие из приведенных ниже определений переменных содержат синтаксические ошибки?

(a) int car = 345, auto = 34.6; (c) cin>>int tmp;

(b) double mobile = 23.4; (d) chat alpha = ‘!’;

6. Укажите правильные операторы присваивания

(a) int i; (c) int i;

i = 3.5; i ==3;

(b) int i; (d) int i;

i = 5; i = 3%5;

7. Укажите правильно записанные инструкции вывода:

(a) cout<<”This is the string and ”<<x<<”is the number”; (d) cout<<”This is the string and ”>>x<<”is the number”;

(b) cout<<”This is the string and ”x”is the number”; (e) cin<<”This is the string and ”<<x<<”is the number”;

(c) cin>>’This is the string and’>>x<<’is the number’; (f) cout<<”This is the string and \n”<<x<<”\n is the number”;

8. Укажите правильно записанные инструкции ввода:

(a) cout<<value; (d) cout<< ‘value’;

(b) cout>> value; (e) cin>> ‘value’;

(c) cin>> value; (f) cout<<value>>;

9. Какое значение будет иметь переменная val в результате выполнения данного фрагмента кода?

```
int val=3, i=7;
        val=val*i+(++i);
```

(a) 32; (b) 28;

(c) 29;

10. Какое значение будет иметь переменная val в результате выполнения данного фрагмента кода?

```
int val=3, i=7;
        val=val*i+++i;
```

(a) 32; (b) 28;

(c) 29;

11. Укажите определение переменной str, так чтобы приведенный ниже код не вызывал ошибок:

```
char str[…] =  “My string ”;
str[9] = ‘1’;
cout<<str;
```

(a) char str[12] = “My string ”;
(b) char str[11] = “My string ”;
(c) char str[10] = “My string ”;
(d) char str[9] = “My string ”;

12. Дайте определение понятия алгоритм. Перечислите и поясните свойства алгоритма.

13. Дайте определение понятия идентификатор. Перечислите правила его написания.

14. Как инициализация, так и присваивание обозначаются одним и тем же знаком «=». В чем отличие этих двух операций?

15. Как вы думаете, почему язык программирования назван C++, а не ++C?

16. Поясните понятие динамической типизации.

ЧАСТЬ 2

1. Напишите алгоритм в словесной форме и на псевдокоде для решения следующей задачи перехода пешехода через дорогу при отсутствии светофора.

2. Заполните пропущенные фрагменты кода и при необходимости откорректируйте арифметическое выражение, так чтобы в результате переменная value принимала значение 2

```
… value = 2,i;
    … j = 2;
    i = 5%value - sqrt(2*j);
    value = (value*i+1)*value+++i;
```

Приложение 8

Аудиторная контрольная работа по дисциплине «Программирование для начинающих»

Alice содержит набор объектно-ориентированных классов (AnObject, ACamera, ALight и т. д.), которые представляют доступ к управлению анимацией, движению и вращению, а также контролю над вложением родительского или дочернего отношения между объектами. Создать проект, используя данные ООП классы. Например, класс ALight реализует ту часть API-интерфейса Alice, которая управляет методами, специфичными для света, которые отличаются от AnObject: (например, TurnOn, TurnOff, SetBrightness). Alice поддерживает несколько видов света: Spotlights, PointLights, FillLights и SunLight. Тип создаваемого света выбирается с помощью параметра конструктора, а не создания четырех отдельных подклассов для пользователей:

```
My_light = ALight ()
                # the default is to create a PointLight
                # though you can create others kinds too…
My_fill_light = ALight (FillLight)
My_spotlight= ALight (SpotLight)
```

Аудиторная контрольная работа по дисциплине «Информатика и программирование»

1. Программно реализовать класс STUDENT, содержащий следующие поля:

– Name – Фамилия и инициалы;

– Year – год рождения;

– Bal – оценки из 4 предметов (массив из 4 элементов).

2. Написать программу, которая использует данный класс и вводит с клавиатуры массив данных Group, состоящий из N переменных типа STUDENT;

– выводит на экран фамилии и год рождения студентов средний балл которых> 4.0;

3. Программу создать в трех файлах: заготовочный файл *.h с описанием класса, файл с функциями класса *.cpp, и главная функция main.cpp. Доступ к полям класса выполнить через методы класса.

Приложение 9

Внеаудиторная контрольная работа

по дисциплине «Программирование для начинающих»

Задание №____

для контрольной работы
по дисциплине «Программирование для начинающих

Целью является: сформировать фундамент основных понятий в области объектно-ориентированного программирования, обеспечить устойчивые навыки работы в области программирования; а также сформировать у студентов мотивацию к дальнейшему программированию в области ООП.

Контрольная работа состоит из двух модулей:

1. Работа в визуальной учебной среде программирования Alice (приложение А).
2. Работа в визуальной учебной среде программирования Scratch (приложение Б).

Дата выдачи _____ Студент _____

. Преподаватель _____

Дата защиты КР «____»_____20__г.

Оценка _____

Руководитель_____.ст.преп. Шкарбан Ф.В.

Приложение А

Описание сценария

Утром прекрасного солнечного дня несколько видеооператоров решили сделать документальные съемки перекрестка. Съемки велись и на земле, и в воздухе.

Сначала зрителю представляется вид сверху, откуда можно разглядеть прекрасные пейзажи, разные пейзажи (зеленые газоны, деревья, кусты), современные постройки (обитаемые дома, супермаркет, административные заведения), высокотехнологичную дорогу с качественным покрытием и разметкой, отделенную забором от тротуара. Постепенно зритель оказывается на перепутье четырех дорог. С одной стороны можно

увидеть, как искусно водитель желтого автомобиля заезжает на парковку, которая отвечает евростандартам. Потом появляется двор одного из прилегающих к перекрестку обитаемых секторов и показано величие тамошнего небоскреба. Плавный переход и дорожный знак указывает на местонахождение парковки, на которой находятся шедевры современного машиностроения. Новинка автосалона приближается к перекрестку и останавливается ради предоставления возможности движения другим транспортным средствам.

Неожиданно зритель оказывается в центре растительного разнообразия; парк оживляется с помощью местного фонтана сказочной красоты. Неподалеку от него можно увидеть площадки для отдыха, оснащенные скамьями и ограниченных заборчиками, чтобы рассмотреть отдельные его детали.

После этого перед нами предстает супермаркет во всей своей красоте, если смотреть со стороны парковки. И наконец, с высоты птичьего полета можно увидеть, как водители, пропуская друг друга, разъезжаются в разные стороны.

В конце фильма зрителю представляется возможность еще раз увидеть перекресток, который стал основой для создания фильма.

Приложение Б

Описание игры Mario

Главными героями игры являются водопроводчик Марио Цель игры — пройти через Грибное королевство, ускользая или уничтожая солдат черепашьего Короля Купы, чтобы спасти захваченную им в плен Принцессу.

Марио атакует противников, прыгая на них сверху (такой способ убивает грибов Гумба и временно нейтрализует черепах Купа, заставляя их прятаться в свои панцири) или ударяя по платформе, на которой находится противник, снизу. «Испугавшихся» черепах можно использовать как оружие против других врагов, подтолкнув панцирь впереди себя: разогнавшись, он сметает всех на своем пути, но когда встречает преграду, меняет направление и может ударить самого Марио. На некоторых врагов, например дикобразов Спини, нельзя запрыгивать, так как это ранит Марио. Их можно убить, выстрелив огненными шарами, пнув в них черепаху или ударом по платформе снизу. То же самое касается хищных растений, регулярно показывающихся из торчащих в земле труб. Некоторые из этих труб являются проходом в бонусный уровень, подземелье, в котором можно найти небольшое количество монет, а заодно сократить путь до выхода с уровня. Также встречаются ростки, по которым Марио поднимается на облака, бонусный уровень, на котором большое количество монет и нет противников.

По пути Марио собирает монеты и бонусы, ударяя по блокам со знаком вопроса, а также выискивая секретные хранилища монет в кирпичных стенах. При наборе ста монет Марио получает дополнительную «жизнь», изначально у Марио есть три «жизни». За поверженных врагов начисляются очки, которые не приносят какой-либо практической пользы, а служат лишь для отражения мастерства игрока. При наборе 1 миллиона очков шестизначное табло увеличивается до семизначного. При последовательном поражении врагов одним черепашьим панцирем очки начисляются по возрастающей за каждого убитого: 500, 800, 1000, 2000, 4000, 5000, 8000 очков, а затем (при поражении восьмого персонажа) Марио начисляется дополнительная «жизнь».

Среди бонусов может встретиться оранжевый гриб, взяв который, персонаж увеличивается в размерах, превращаясь в Супер Марио. Если взять после этого цветок, то Супер Марио становится Огненным Марио и получает возможность стрелять огненными шарами, то есть поражать противника на расстоянии.

Если Супер Марио или Огненный Марио дотрагиваются до противника, то они возвращаются в стадию обычного Марио. Если же противник ранит Марио или кончается время, выделенное для прохождения уровня, то игрок теряет одну «жизнь» и игра начинается заново (либо с начала уровня, либо с его середины). В уровнях, которые расположены в за́мках, погибший Марио начинает игру всегда с начала уровня.

Ещё один встречающийся в игре бонус — звезда. Она не «вырастает» из блока как гриб, а выскакивает из него и начинает перемещаться по синусоидальной траектории вперёд по направлению движения. Заполучив её, Марио становится на некоторое время неуязвимым для врагов: противники погибают от одного его прикосновения. На некоторых уровнях встречаются также бонусы в виде зелёного гриба «1-up», который добавляет Марио одну «жизнь» (гриб «1-up» движется, и для того, чтобы его поймать, игроку приходится проявить сноровку).

Игра состоит из восьми миров по четыре уровня в каждом. В первом уровне во всех мирах Марио находится на земле, во втором чаще всего под землёй, в канализации или под водой, в третьем уровне Марио ходит по гигантским грибам и прыгает по платформам, висящим в воздухе, в четвёртом уровне Марио попадает в замок. Во всех мирах, кроме третьего и шестого, Марио путешествует днём. Восьмой мир Марио проходит полностью на земле. В конце каждого четвёртого уровня, на мосту через озеро с лавой, Марио встречается с Королём Купой (англ. *King Koopa*) — огромным черепаховидным драконом. Король Купа дышит огнём, а начиная с шестого мира — ещё и метает огромные молоты. Победить Купу можно двумя способами: дотронуться до топора, находящегося за Купой (тот перерубит трос и Король Купа упадёт в лаву), либо поразить короля Купу с безопасного расстояния огненными шарами (для этого нужно чтобы Марио находился в стадии Огненного Марио и весь путь через замок прошёл без потерь). В случае победы над Купой в первых семи мирах с помощью огня, выясняется, что это на самом деле переодетый простой враг, как черепаха, Гумба и т. д. После этого Марио встречает Тоада — жителя Грибного королевства, который говорит ему знаменитую фразу: «Thank you Mario! But our princess is in another castle!» (рус.*Спасибо, Марио, но наша принцесса в другом замке!*)

Внеаудиторная контрольная работа по дисциплине «Информатика и программирование»

Контрольная работа по дисциплине «**Информатика и программирование**» позволяет освоить принципы и методы объектно-ориентированного стиля программирования, а также является заключительным этапом изучения теоретических основ и обеспечивает закрепление лекционного материала путем более глубокого изучения основных разделов дисциплины. Целью является формирование навыков самостоятельной творческой работы и закрепление теоретических знаний, полученных при изучении курса.

Задание

Используя средства языка программирования C++ реализовать программно игру «Тетрис».

Приложение 10

Пример экзаменационного билета по дисциплине «Информатика и программирование»

ГБОУВО РК КИПУ

Образовательно-квалификационный уровень Бакалавр
Направление подготовки 09.03.03 Прикладная информатика
Профиль Прикладная информатика в информационной сфере
Семестр 3
Учебная дисциплина Информатика и программирование

Первый вопрос экзаменационного билета проверяет сформированность компетенции ПК-2, второй – ПК-12, третий – ПК-8

ЭКЗАМЕНАЦИОННЫЙ БИЛЕТ № 1

1. Принципы объектно-ориентированного программирования. Инкапсуляция. Полиморфизм. Наследование.
2. Понятия класса и объекта. Общее и различие в понятиях класс и объект.
3. Задача.

Создать класс для динамической структуры, которую называют *очередь*. Структура содержит два информационных поля: фамилия студента и экзаменационный балл по информатике. Класс будет иметь такие методы: конструктор, приобщение элемента в очереди, изъятия элемента из очереди, просмотра элементов очереди.

Приложение 11

Проверка уровня знаний обучающихся в области программирования

1. Дайте определение понятию идентификатор. Перечислите правила его написания.

2. Дан массив целых чисел. Написать функцию, определяющую, является ли *k*-тый элемент массива положительным числом (*k* ввести с клавиатуры).

3. Даны значения длин катетов прямоугольного треугольника. Определить функцию, вычисляющую его гипотенузу.

4. Если десятичное число 33 перевести в двоичную систему счисления, то получится число …

(a) 100001;
(b) 101001;
(c) 100010;
(d) 100011.

Приложение 12

Анкета

Для студентов 1-го курса факультета экономики, менеджмента и информационных технологий направления подготовки 09.03.03 Прикладная информатика

1. Удовлетворяет ли Ваш уровень школьной подготовки по программированию требованиям высшей школы?

☐ да

☐ нет

. 2. Занимались ли Вы в школе программированием дополнительно?

☐ да

☐ нет

3. Знаете ли Вы разницу между функциональной и объектно-ориентированной парадигмами программирования?

☐ да

☐ нет

4. Можете ли Вы привести пример языка программирования, поддерживающего объектно-ориентированный подход?

☐ да

☐ нет

5. Хотелось бы изучить объектно-ориентированные языки программирования более подробно?

☐ да

☐ нет

Дата_____ группа_____ _____

подпись

Приложение 13

Контрольный срез знаний

1. Дайте определение понятию алгоритм. Перечислите и поясните свойства алгоритма.

2. Дан массив целых чисел. Написать программу для работы с массивом: инициализировать массив, вывести его на экран, и вычислить квадратный корень из любого элемента массива, позиция которого введена с клавиатуры.

3. Вычислить значение *y=max(2a-b, b)*. Для этого написать определение функции *max(a,b)*, возвращающей максимальное из чисел *x* и *y*.

4. Если двоичное число 1011 перевести в десятичную систему счисления, то получится число …

(a) 11; (c) 13;

(b) 14; (d) 6.

Приложение 14

Анкета для диагностики уровня сформированности компетенции бакалавра прикладной информатики в области объектно-ориентированного программирования

Какие из предложенных мотивов могут побудить Вас использовать объектно-ориентированную парадигму программирования в будущей профессиональной деятельности?

Оцените по 3-балльной шкале степень значимости мотивов, которые могут побудить Вас использовать облачные технологии в будущей профессиональной деятельности (3 – максимальная степень значимости, 2 – средняя степень значимости, 1 – минимальная степень значимости).

1. Потребность в значимой работе для общества.
2. Потребность оказывать влияние на развитие технологий в области компьютерной науки.
3. Стремление к саморазвитию и самосовершенствованию.
4. Потребность быть конкурентоспособным.
5. Стремление повысить эффективность своей профессиональной деятельности.
6. Потребность в карьерном росте.
7. Стремление повысить свою квалификацию.
8. Потребность заниматься самообразованием в области программирования.
9. Потребность утвердиться среди коллег.
10. Стремление к развитию познавательных способностей

Приложение 15

Тест для диагностики уровня сформированности организационно-содержательного компонента компетенции бакалавра прикладной информатики в области объектно-ориентированного программирования

Какой из перечисленных типов данных не является типом данных в C++?
A) Double
B) Real
C) Float
D) int
Цикл с постусловием задается инструкцией:
A) for
B) while
C) do while
Какую функцию должны содержать все программы на C++?
A) system()
B) program()
C) main()
D) start()
Какой из ниже перечисленных операторов, не является циклом в C++?
A) repeat until
B) do while
C) while
D) for
Какой из следующих операторов – оператор сравнения двух переменных?
A):=
B) = =
C) =
Цикл с предусловием задается инструкцией:
A) for
B) while
C) do while
Какой из следующих операторов – оператор присваивания?
A):=
B) = =
C) =
Цикл с параметром задается инструкцией:
A) for
B) while
C) do while
Унарный оператор инкремента ++ увеличивает значение переменной на:
A) 1
B) 2
C) не увеличивает значение

Запись вида x+=a позволяет

A) увеличить значение a на x

B) увеличить значение x на a

C) присвоить x значение a

В примере:

for(i=0;i<3;++i)

cout<<i<<endl;

A) на экран выводится сумма чисел 1, 2 и 3

B) на экран выводятся числа 1, 2 и 3

C) на экран выводятся числа 0, 1 и 2

Бесконечный цикл можно задать следующим образом:

A) for(;i<3;++i)

B) for(; ;)

C) for(i=0;i<3;++i)

Цикл while с предусловием позволяет выполнить одну и ту же последовательность действий пока проверяемое условие:

A) истинно

B) ложно

C) отсутствует

Если в массиве M 7 элементов, каким будет правильное обращение к первому элементу массива?

A) M [7]

B) M [0]

C) M [1]

D) M [6]

Если в процессе компиляции программы возникла ошибка то:

A) будет создан исполняемый файл

B) компилятор выдаст сообщение об ошибке и создаст исполняемый файл

C) компилятор выдаст сообщение об ошибке, с возможным указанием её места

Ключи к тестам

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
вар	B)	C)	C)	A)	B)	B)	C)	A)	A)	B)	C)	B)	A)	B)	C)

Приложение 16

Тест для диагностики уровня сформированности когнитивно-операционного компонента компетенции бакалавра прикладной информатики в области объектно-ориентированного программирования

1. Объект представляет собой:
    1. Элемент данных, включая собственные методы.
    2. Предмет, который имеет имя.
    3. Экономическая категория сложной системы.
    4. Производственные отношения в процессе анализа элементов системы.
2. Атрибут объекта – это:
    1. Взаимообмен между классами.
    2. Реализация собственных диалогов.
    3. Значение, характеризующее объект в его классе.
    4. Действие обработки ошибки.
3. Операция является полиморфной, если:
    1. Одна операция может применяться к объектам разных классов.
    2. Добавляются два или более объектов одного класса.
    3. Аргументы функции являются указателями.
    4. Существует значение по умолчанию.
4. Имена полей необходимо указывать, если:
    1. Квалификаторы указываются в схемах.
    2. Существует несколько сторон трипарной зависимости.
    3. Пользователь имеет свою директорию.
    4. Зависимость устанавливается между объектами одного класса.
5. Важным свойством отношения агрегации являются:
    1. Идентификация.
    2. Инкапсуляция.
    3. Транзитивность.
    4. Полиморфизм.
6. Правила наследования необходимо соблюдать тогда, когда:
    1. Операции-запросы должны наследоваться всеми подклассами.
    2. Операции, изменяющие значения атрибутов, должны наследоваться во всех их расширениях.
    3. Операции, изменяющие значение ограниченных атрибутов или атрибутов определенных зависимостей, должны блокироваться во всех их расширениях.
    4. Наследованные операции можно уточнять, добавлять к ним дополнительные действия.
7. Абстрактный класс не может иметь:
    1. Операции.
    2. Методов.
    3. Объектов.
    4. Подклассов.
8. Делегирование – это:
    1. Данные, специфичные для потока.
    2. Механизм реализации пересылки операций от одного объекта к другому.
    3. Механизм получения индекса обработки.
    4. Выбор собственного родительского класса.
9. Возможным ключом называют:
    1. Минимальный набор атрибутов, которые однозначно идентифицируют объект или связь.

2. Метасимволы пакета регулярных выражений.
3. Методы регулирования взаимодействий между объектами.
4. Механизм функционирования объектов.

10. Динамическая модель состоит из:
1. Связанных объектов.
2. Наследования событий.
3. диаграмм состояний ее объектов и подсистем.
4. Операций между атрибутами объектов.

11. Условие – это:
1. Действие над объектами.
2. Логическая функция от значений объектов.
3. Операция над атрибутами объектов.
4. Диаграмма состояний.

12. Разработчик системы должен:
1. Оценить производительность и необходимые ресурсы.
2. Выбрать способ реализации подсистем (аппаратный или программный).
3. Распределить программные подсистемы по процессорам.
4. Построить математическую модель.

13. При разработке системы с интерактивным интерфейсом необходимо:
1. Выделить объекты, формирующие интерфейс.
2. Использовать готовые объекты для организации взаимодействий.
3. Определить по динамической модели структуру программы.
4. Из множества событий выделить аппаратные, простые.

14. При выборе алгоритма следует учитывать:
1. Вычислительную сложность.
2. Ясность и легкость реализации.
3. Гибкость.
4. Структуру.

15. При разработке системы непрерывной обработки необходимо выполнить:
1. Построение диаграммы потока данных.
2. Определить классы промежуточных объектов.
3. Представление каждой фазы в виде последовательности изменений значений элементов исходной структуры данных.
4. Численное моделирование.

Приложение 17

Диагностика уровня сформированности личностно-рефлексивного компонента компетенции бакалавра прикладной информатики в области объектно-ориентированного программирования

(использовалась методика диагностики способности к саморазвитию и самообразованию, предложенная В.И. Андреевым [http://www.lib.tpu.ru/fulltext/m/2010/m32.pdf стр. 57 и ответы на стр.76]).

1. За что Вас ценят Ваши друзья?
   а) За то, что преданный и верный друг;
   б) Сильный и готов в трудную минуту за них постоять;
   в) Эрудированный и интересный собеседник.
2. На основе сравнительной самооценки выберете, какая характеристика Вам более всего подходит?
   а) Целеустремленный;
   б) Трудолюбивый;
   в) Отзывчивый.
3. Как Вы относитесь к идее ведения личного ежедневника, к планированию своей работы за год, месяц, ближайшую неделю, день?
   а) Думаю, что чаще это пустая трата времени;
   б) Я пытался это делать, но нерегулярно;
   в) Положительно, так как я давно это делаю.
4. Что Вам больше всего мешает профессионально самосовершенствоваться, лучше учиться?
   а) Нет достаточно времени;
   б) Нет подходящей литературы и условий;
   в) Не всегда хватает силы воли и настойчивости.
5. Каковы типичные причины Ваших ошибок и промахов?
   а) Невнимательный;
   б) Переоцениваю свои способности;
   в) точно не знаю.
6. На основе сравнительной оценки выберите, какая характеристика Вам более всего подходит?
   а) Настойчивый;
   б) Усидчивый;
   в) Доброжелательный.
7. На основе сравнительной оценки выберите, какая характеристика вам более подходит?
   а) Решительный;
   б) Любознательный;
   в) Справедливый.
8. На основе сравнительной оценки выберите, какая характеристика вам более подходит?
   а) Генератор идей;
   б) Критик;
   в) Организатор.
9. На основе сравнительной оценки выберите, какие качества у Вас развиты в большей степени?
   а) Сила воли;
   б) Память;
   в) Обязательность.
10. Что чаще всего делаете, когда у вас появляется свободное время?
    а) Занимаюсь любимым делом, у меня есть хобби;
    б) Читаю художественную литературу;
    в) Провожу время с друзьями либо в кругу семьи.
11. Что из ниже приведенных сфер для Вас в последнее время представляет познавательный интерес?
    а) научная фантастика;
    б) Религия;
    в) Психология.

12. Кем бы Вы могли себя реализовать?

а) Спортсменом;

б) Ученым;

в) Художником.

13. Каким чаще всего считают или считали Вас учителя?

а) Трудолюбивым;

б) Сообразительным;

в) Дисциплинированным.

14. Какой из трех принципов Вам ближе всего и которого Вы придерживаетесь чаще всего?

а) Живи и наслаждайся жизнью;

б) Жить, чтобы больше знать;

в) Жизнь прожить не поле перейти.

15. Кто ближе всего к вашему идеалу?

а) Человек здоровый, сильный духом;

б) Человек, много знающий и умеющий;

в) Человек независимый и уверенный в себе.

16. Удастся ли Вам в жизни добиться того, о чем Вы мечтаете, в профессиональном и личном плане?

а) Думаю, что да;

б) Скорее всего да;

в) Как повезет.

17. Какие фильмы Вам больше всего нравятся?

а) Приключенческо-романтические;

б) Комедийно-развлекательные;

в) Философские.

18. Представьте себе, что Вы заработали миллион. Куда бы Вы предпочли его истратить?

а) Путешествовал бы и посмотрел мир;

б) Поехал бы учиться за границу или вложил деньги в любимое дело;

в) Купил бы коттедж с бассейном, мебель, шикарную машину и жил бы в свое удовольствие.

В.И. Андреев. Оценка способности к саморазвитию, самообразованию

Ваши ответы на вопросы теста оцениваются следующим образом:

Вопрос	Оценочные баллы ответов	Вопрос	Оценочные баллы ответов
1	а) 2 б) 1 в) 3	10	а) 2 б) 3 в) 1
2	а) 3 б) 2 в) 1	11	а) 1 б) 2 в) 3
3	а) 1 б) 2 в) 3	12	а) 1 б) 3 в) 2
4	а) 3 б) 2 в) 1	13	а) 3 б) 2 в) 1
5	а) 2 б) 3 в) 1	14	а) 1 б) 3 в) 2
6	а) 3 б) 2 в) 1	15	а) 1 б) 3 в) 2
7	а) 2 б) 3 в) 1	16	а) 3 б) 2 в) 1
8	а) 3 б) 2 в) 1	17	а) 2 б) 1 в) 3
9	а) 2 б) 3 в) 1	18	а) 2 б) 3 в) 1

По результатам тестирования Вы можете определить уровень Вашей способности к саморазвитию и самообразованию

Суммарное число баллов	Уровень способностей к саморазвитию и самообразованию	Суммарное число баллов	Уровень способностей к саморазвитию и самообразованию
18 – 25	1 – очень низкий	38 – 40	6 – чуть выше среднего
26 – 28	2 - низкий	41 – 43	7 – выше среднего
29 – 31	3 – ниже среднего	44 – 46	8 – высокий
32 – 34	4 – чуть ниже среднего	47-54	9 – очень высокий
35 – 37	5 - средний		