Автономная некоммерческая организация высшего образования
«Поволжский православный институт имени Святителя Алексия, митрополита Московского»

Кафедра педагогики и психологии

Направление подготовки 44.03.01 Педагогическое образование
Направленность (профиль) «Информатика и информационные технологии»

БАКАЛАВРСКАЯ РАБОТА

на тему:

«Программно-методическое обеспечение раздела «Объектно-ориентированное программирование» профильного курса информатики»

Выполнила студентка
4 курса группы ИТ-401
очной формы обучения
Артамонова Екатерина
Евгеньевна

_____
(подпись)

Научный руководитель
к. п. н., доцент Дудина И. П.

_____
(подпись)

Допустить к защите:
Заведующий кафедрой
педагогики и психологии _____ Е. А. Денисова
(подпись) *(И.О.Ф..)*

«___»_____20___г.

Тольятти
2020

ОГЛАВЛЕНИЕ

ВВЕДЕНИЕ

Обучение программированию, как одно из важнейших направлений деятельности человека в информационной среде, рассматривается на сегодняшний день как важнейший компонент образования. В наши дни программирование является востребованным практически во всех сферах нашей жизни, таких как строительство, бизнес и экономика, медицина, биология и физика. Большой процент физического труда в промышленности заменен на машинный и роботизированный труд, который управляется посредством программного обеспечения, что обеспечивает существенный прирост скорости, точности операций и эффективности производства. Такое богатство разнообразия применений обеспечивается солидным выбором языков программирования и типов, к которым они относятся: процедурное, структурное, объектно-ориентированное, декларативное, логическое программирование. Более подробно рассмотрим объектно-ориентированное программирование.

Объектно-ориентированное программирование (ООП) – самая эффективная и перспективная на сегодняшний день парадигма программирования, главными понятиями которой являются объект и класс. Она реализована в многочисленных языках программирования, таких как C, C#, C++, Object Pascal, Delphi, Java и др.

Высокоуровневый язык C# – разработка корпорации Майкрософт, является одним из самых востребованных и популярных современных языков программирования объектно-ориентированной парадигмы. Он используется в качестве средства разработки приложений во многих странах. На нем ведется работа по созданию многофункциональных программных проектов, сложных облачных веб-сервисов, мобильных приложений.

Достижения в области программирования актуализируют вопросы подготовки специалистов в области написания программ различной сложности.

Изучение современных языков и технологий программирования способствует формированию операционального, алгоритмического и объектного стилей мышления. А также влияет на формирование научного мировоззрения будущего IT-специалиста, на представление о возможности двойственного взгляда на окружающую действительность - с точки зрения процессов (структурное программирование) и с точки зрения объектов (объектно-ориентированное программирование).

Обучением объектно-ориентированному программированию в школьном курсе информатики занимались такие педагоги и ученые, как К. Ю. Поляков [31, 32], И. Г. Семакин, Е. К. Хеннер [37], Н. В. Макарова [22, 23], Н. Д. Угринович [40] и другие.

Комплексный анализ состояния методики обучения учащихся средних школ объектно-ориентированному программированию, а также современные исследования в указанной области позволили выделить противоречие между необходимостью создания учебных пособий и методических рекомендаций, позволяющих организовать процесс обучения объектно-ориентированному программированию, и недостаточной разработанностью научно-практических рекомендаций в этой области.

Актуальность темы исследования определяется отсутствием необходимых теоретически обоснованных учебно-методических материалов по этому разделу школьного курса информатики.

На основе анализа актуальности и выявленных противоречий сформулирована проблема исследования, заключающаяся в необходимости разработки и обосновании эффективности методики обучения объектно-ориентированному программированию учащихся профильных классов средних школ.

Теоретико-методологическую основу исследования составляют концепции в области алгоритмического мышления (С. Л. Рубинштейн [35], П. Я. Гальперин [8], А.П. Ершов, Г.А. Звенигородский, Ю.А. Первин [12], А. В. Копаев [15] и др.); педагогических и информационных технологий (И. С.

Якиманская [42], Т. А. Жданко [13], М. М. Шубович [41] и др.); теории и методики обучения информатике и программированию (Кузнецов В. В. [19], М. П. Лапчик, М. И. Рагулина [20], С. В. Зыков [14], К. Ю. Поляков [31, 32], И. Г. Семакин, Хеннер Е. К., Шестакова Л. В. [37], Угринович Н. Д. [40], Н. В. Макарова [22, 23] и др.); дидактические аспекты использования информационных технологий (Гафурова Н.В., Чурилова Е.Ю. [9], Е.И. Машбиц [25] и др.).

Актуальность выявленной проблемы и ее недостаточная разработанность в частной дидактике определила выбор темы исследования.

Объект исследования – процесс обучения объектно-ориентированному программированию в профильном курсе информатики средней школы.

Предмет исследования – программно-методическое обеспечение по обучению объектно-ориентированному программированию учащихся профильных классов средней школы.

Цель исследования – разработка, обоснование структуры, содержания и методов реализации программно-методического обеспечения по обучению объектно-ориентированному программированию учащихся профильных классов.

Задачи исследования:

1. Провести анализ психолого-педагогической, научно-методической и учебной литературы, связанной с проблемой введения профильного обучения на старшей ступени общего образования.
2. Выявить основные проблемы обучения разделу «Объектно-ориентированное программирование» учащихся профильных классов средней школы.
3. Определить роль и место раздела «Объектно-ориентированное программирование» в профильном обучении информатике и ИКТ и сформулировать основные требования к его структуре, содержанию, методам, средствам и формам обучения.

4. Выделить и обосновать этапы разработки и использования программно-методического обеспечения обучения объектно-ориентированному программированию учащихся профильных классов средней школы.

5. Выполнить программную реализацию электронных образовательных ресурсов для обучения разделу «Объектно-ориентированное программирование»: разработать интерфейс пользователя; функционал компонентов на базе созданных сценариев и режимов использования.

6. Разработать методические рекомендации для учащихся и преподавателей.

Гипотеза исследования: если использовать предложенное программно-методическое обеспечение обучения школьников объектно-ориентированному программированию, регламентируемое личностно-ориентированным и компетентностным подходами, а также дидактическими принципами наглядности, научности и индивидуальности, это позволит обеспечить достижение предметных и метапредметных результатов обучения в соответствии с образовательным стандартом общего среднего образования.

Методы исследования:

1. Теоретические: системный анализ отечественной и зарубежной психолого-педагогической, научно-методической литературы по педагогике, психологии, информатике; изучение и анализ нормативных документов в сфере общего образования, критический анализ существующих подходов к обучению объектно-ориентированному программированию, а также использованию электронных ресурсов по рассматриваемой проблеме.

2. Эмпирические: обобщение опыта преподавания информатики; анализ содержания учебных программ, планов, пособий, материалов конференций по вопросам обучения объектно-ориентированному программированию в школе; наблюдение, беседа, анкетирование, тестирование учащихся с целью выяснения целесообразности использования предложенной методики и ее эффективности в области развития познавательного и творческого потенциала школьников.

Первая глава бакалаврской работы отражает теоретическое обоснование проводимого исследования, а именно научно-педагогические и методические основы и особенности изучения объектно-ориентированного программирования в профильных классах средней школы, дидактические и технологические требования к электронному учебно-методическому комплексу по разделу «Объектно-ориентированное программирование».

Вторая глава представляет собой практическую проектную часть работы: результаты проектирования, поэтапной разработки и использования электронного учебно-методического комплекса по теме исследования.

Третья глава представляет собой оценку эффективности использования разработанного ЭУМК в практике обучения средней школы и результаты проведенного педагогического эксперимента.

В заключении подводятся итоги проделанной работы.

Библиографический список содержит перечень источников информации, использованных при выполнении бакалаврской работы.

Приложения включают разработанные практические задания и демонстрационные примеры их решения.

Глава 1 Теоретические предпосылки разработки методики обучения по разделу «Объектно-ориентированное программирование» в профильном курсе информатики

1.1 Научно-педагогические и методические основы реализации раздела «Объектно-ориентированное программирование» в школьном курсе информатики

Объектно-ориентированное программирование – один из ведущих разделов информатики и одно из важнейших направлений деятельности человека в современной информационной среде.

Изучение современных языков и технологий программирования способствует формированию операционального, алгоритмического и объектного стилей мышления, а также влияет на формирование научного мировоззрения будущего IT-специалиста, на представление о возможности двойственного взгляда на окружающую действительность - с точки зрения процессов (структурное программирование) и с точки зрения объектов (объектно-ориентированное программирование).

Термин «алгоритмический стиль мышления» широко используется в научной и методической литературе, посвященной обучению информатике и ИКТ. Это связано с тем, что одной из основных задач изучения данной дисциплины является формирование и развитие алгоритмического мышления учащихся. По мнению А.В. Копаева [15], ключевой проблемой при выяснении целей обучения и содержания модуля «Алгоритмизация и программирование» является уточнение понятия «алгоритмический стиль мышления»

С.Л. Рубинштейн считал, что мышление есть «движение мысли, раскрывающее связь, которая ведет от отдельного к общему и от общего к отдельному» [35].

Понятие «алгоритмическое мышление» А.П. Ершовым, Г.А. Звенигородским, Ю.А. Первиным [12] определяется как «умение планировать структуру действий, необходимых для достижения цели, при помощи

фиксированного набора средств»; «умение строить информационные модели для описания объектов и систем»; «умение организовывать поиск информации, необходимой для компьютерного решения поставленной задачи».

В работе П. Я. Гальперина [8] определено понятие – «логико-алгоритмическое мышление». Логико-алгоритмическое мышление проявляется в умении строить логические утверждения о свойствах данных и запросы к поисковым системам, мыслить индуктивно и дедуктивно при анализе затруднений в работе с персональным компьютером, формализовать собственные намерения вплоть до записи на некотором алгоритмическом языке. Наличие развитого алгоритмического мышления является необходимым условием способности к составлению программ для электронно-вычислительной машины.

Учебное пособие Рыжова В. Н. «Методика преподавания информатики» [36] описывает цели, принципы и методы обучения информатики в школе. Содержание пособия разделено на несколько частей, одна из них посвящена общим вопросам методики преподавания информатики и ИКТ в школе, другая – методике преподавания базового и профильного курсов, третья – методике обучения школьников с применением информационных технологий. В данном пособии автор говорит об актуальности объектно-ориентированного программирования: «Оно занимает ведущее место в разработке современных программных средств и поэтому ознакомление с ним необходимо для учащихся, ориентированных на профессию программиста», а также описывает варианты курсов с тематическим планированием по объектно-ориентированному программированию.

В методическом пособии обучения информатике М. П. Лапчик, М. И. Рагулина, И. Г. Семакин, Е. К. Хеннер пишут, что объектно-ориентированное программирование в настоящее время занимает ведущее место в разработке профессиональных программных средств. В ходе изучения ООП решаются следующие задачи: освоение методологии объектно-ориентированного программирования; овладение техникой объектно-ориентированного

программирования на одном из языков; введение учащихся в проблематику, адекватную данному подходу, расширение общего кругозора (общеобразовательный компонент). Вместе с тем, авторы методического пособия полагают, что методика изучения в школе любых видов объектного программирования разработана совершенно недостаточно и что этот процесс в настоящее время, по существу, находится на начальной стадии [20].

Зыков С. В. в пособии «Введение в теорию программирования. Объектно-ориентированный подход» [14] рассматривает особенности объектно-ориентированного подхода к программированию в сравнении с остальными подходами. Автор выделяет одно из преимуществ ООП – высокий процент повторного использования уже разработанного программного кода, но при этом данный подход требует глубокого понимания основных принципов, на которых он базируется. Учебное пособие содержит 18 лекций, направленных на изучение базовых понятий и принципов ООП, а так же основных конструкций программирования.

Таким образом, изучение объектно-ориентированного программирования в школе – один из важнейших компонентов современного образования. Данный раздел представлен в Федеральном государственном образовательном стандарте среднего образования (ФГОС СОО) [1] по предметной области «Информатика».

ФГОС СОО основан на системно-деятельностном подходе, который обеспечивает формирование готовности у обучающихся к непрерывному образованию и саморазвитию, проектирование и конструирование развивающей образовательной среды организации, осуществляющей образовательную деятельность, активную познавательную деятельность обучающихся, построение образовательной деятельности, учитывая индивидуальные, возрастные, психологические, физиологические особенности и здоровье обучающихся.

В статье «Системно-деятельностный подход: сущностная характеристика и принципы реализации» Т. А. Жданко [13] определяет понятие «системно-

деятельностный подход» - это организация учебного процесса, в котором главное место отводится активной и разносторонней, в максимальной степени самостоятельной познавательной деятельности школьника. Ключевыми моментами деятельностного подхода является постепенный уход от информационного репродуктивного знания к знанию действия.

Концепция развития образования РФ до 2020 года [5] указывает, что общеобразовательная школа должна формировать систему универсальных знаний, умений, навыков, а также опыт самостоятельной деятельности и личной ответственности учащихся, то есть современные ключевые компетенции в условиях личностно-ориентированной парадигмы современного отечественного образования. По словам Якиманской И. С. [42], под личностно-ориентированной парадигмой образования понимается исходная концептуальная модель постановки и решения проблем образования, где признаются уникальная сущность каждого ученика и индивидуальность его образовательной траектории. Основная функция личностно-ориентированного образования – обеспечивать и отражать становление системы личностных образовательных смыслов ученика.

Процесс поиска и обретения смыслов в ходе обучения предполагает следующие этапы: личностное творчество ученика по отношению к изучаемым объектам; самоосознание личного опыта, знаний и ценностных отношений ученика, обнаружившихся в процессе познания образовательных объектов и общекультурных сведений о них; позиция и соответствующая деятельность по отношению к фундаментальным достижениям человечества, связанным с изучаемыми объектами (отношение к общекультурным знаниям и социальному опыту). Это позволяет ученику выделить в образовании личностно-значимую основу.

Развиваемые в ходе реализации комплекса перечисленных элементов образовательной деятельности качества ученика называются образовательными компетенциями. Именно компетенции связывают воедино личностный и социальный смысл образования. Введение этого понятия в нормативную и

практическую составляющие образования позволяет решать проблему, когда ученики могут хорошо овладеть набором теоретических знаний, но испытывают значительные трудности в деятельности, требующей использования этих знаний для решения конкретных задач или проблемных ситуаций. Образовательная компетенция предполагает, что ученик не усваивает отдельные друг от друга знания и умения, а овладевает комплексной процедурой, в которой для каждого выделенного направления присутствует соответствующая совокупность образовательных компонентов, имеющих личностно-деятельностный характер. Выделяются следующие ключевые компетенции для общего среднего образования: ценностно-смысловые; общекультурные; учебно-познавательные; информационные; компетенции личностного самосовершенствования; коммуникативные; социально-трудовые.

М.М. Шубович в своей статье «Компетентностный подход как идеология современного личностно-ориентированного образования» [41] пишет, что компетентностный подход направлен к тому, чтобы практически реализовать знания, умения и навыки, развить способности, позволяющие эффективно действовать за пределами ситуаций и сюжетов, которые изучаются в школе, обогащать субъектный опыт и осмыслять собственные жизнедеятельность и бытие в мире.

Согласно ФГОС СОО изучение раздела «Алгоритмизация и программирование» должно обеспечить сформированные основы логического, алгоритмического и математического мышления, сформированные умения по применению полученных знаний при решении различных задач, сформированные представления о влиянии информационных технологий на жизнь человека в социуме. ФГОС СОО также определяет требования к изучению алгоритмизации и программирования на двух уровнях: базовом и профильном.

Предметные результаты изучения раздела «Алгоритмизация и программирование» на базовом уровне должны отражать владение навыками алгоритмического мышления и осознание необходимости формального

описания алгоритмов, владение умением понимать программы, написанные на универсальном алгоритмическом языке высокого уровня, выбранном для изучения, знанием основных конструкций программирования, умением анализировать алгоритмы с использованием таблиц, владение основными приемами написания программы на алгоритмическом языке для решения стандартной задачи с использованием основных конструкций программирования и отладки таких программ, использование готовых прикладных компьютерных программ по выбранной специализации.

Предметные результаты изучения раздела «Алгоритмизация и программирование» на углубленном уровне должны включать требования к результатам освоения базового курса и дополнительно отражать овладение понятием сложности алгоритма, знание базовых алгоритмов обработки текстовой и числовой информации, алгоритмов сортировки и поиска, владение универсальным языком программирования высокого уровня, представлениями об основных типах данных и структурах данных, умением применять стандартные управляющие конструкции, владение навыками и опытом создания программ в выбранной среде программирования, а также тестирование и отладка программ, владение начальными навыками формализации практической задачи и документирования программ.

Примерная основная образовательная программа среднего общего образования [2] уточняет требования ФГОС СОО к изучению алгоритмизации и программирования на двух уровнях: базовом и углубленном.

Согласно примерной основной образовательной программе среднего общего образования в результате изучения учебного предмета "Информатика" на уровне среднего общего образования выпускник на базовом уровне должен гарантированно получить следующие результаты:

1) определять результат выполнения алгоритма с учетом исходных данных, узнавать рассмотренные алгоритмы обработки чисел;

2) разрабатывать с их помощью простые программы анализа данных;

3) понимать несложные программы, созданные на универсальном алгоритмическом языке высокого уровня, выбранном для изучения;

4) пошагово выполнять элементарные алгоритмы управления исполнителями и анализа текстовых и числовых данных;

5) разрабатывать программы для решения стандартных задач базового уровня из разных предметных областей, используя основные алгоритмические конструкции;

6) использовать уже готовые прикладные компьютерные программы в соответствии с типом поставленных задач и по выбранной специализации.

Дополнительно выпускникам предоставляется возможность научиться применять навыки и опыт создания программ в выбранной среде программирования, включая тестирование и отладку программ, применять стандартные управляющие конструкции последовательного программирования и библиотеки прикладных программ, выполнять созданные программы.

На углубленном уровне примерная образовательная программа определяет следующие требования для выпускника:

1) должен уметь анализировать представленный алгоритм;

2) создавать, анализировать и реализовывать стандартные алгоритмы в виде программ, которые связаны с анализом простейших функций, записью чисел в позиционной системе счисления, делимостью целых чисел, линейной обработкой массивов чисел и последовательностей, анализом строк, а также рекурсивные алгоритмы;

3) применять метод динамического программирования для разработки полиномиальных (не переборных) алгоритмов решения разнообразных задач, для решения прикладных задач, основываясь на изученных алгоритмах и методах;

4) разрабатывать собственные алгоритмы, применять структуры данных (списки, словари, очереди, деревья) при решении задач;

5) при составлении алгоритмов применять стандартные операции со структурами данных;

6) также использовать основные понятия, структуры данных и конструкции последовательного программирования, применять в программах данные всевозможных типов;

7) использовать базовые и собственные подпрограммы для обработки символьных строк;

8) обрабатывать данные, хранящиеся в виде массивов разной размерности, использовать различные типы цикла в зависимости от решаемой подзадачи;

9) выполнять стандартные операции с двоичными и текстовыми файлами, выделять подзадачи, решение которых обязательно для полноценного решения предложенной задачи;

10) выполнять решения подзадач в виде подпрограмм, объединять в единую программу разработанные подпрограммы;

11) применять модульный принцип написания программ, обращаться к библиотекам стандартных подпрограмм.

Наряду с тем выпускник должен уметь использовать алгоритмы сортировки и поиска для решения типовых задач, осуществлять объектно-ориентированный анализ данных задач: вычленять объекты, описывать их свойства и методы; приводить в исполнение объектно-ориентированный подход для решения задач на изученном языке программирования, тестировать и отлаживать программы в выбранной среде программирования; обращаться при создании программ к стандартным библиотекам языка программирования и внешним библиотекам программ, разрабатывать многокомпонентные программные продукты в среде программирования, использовать навыки формализации задачи, описывать программы, создавать инструкции по их использованию и отчеты по осуществленным проектным работам.

Согласно федеральному базисному учебному плану и примерным учебным планам для образовательных учреждений Российской Федерации, реализующих программы общего образования по информатике и ИКТ [3], учебному предмету «Информатика и ИКТ» за 2 года обучения на базовом уровне отводится 70 часов (по одному часу в неделю в течение двух лет), на

профильном уровне – 280 часов (по 4 часа в неделю в течение двух лет).

В различных программах ведущих авторов школьных учебников по информатике на изучение раздела «Алгоритмизация и программирование» отводится от 18 до 63 учебных часов.

В учебнике Семакина И. Г. «Информатика. Углубленный уровень» для 11 класса [37] на этот раздел предполагается 63 часа. Учебный материал по алгоритмизации и программированию представлен 3-мя разделами, которые включают в себя 22 параграфа, из которых 5 параграфов отведено на изучение объектно-ориентированного программирования, рассчитанных на 10 учебных часов. В данном учебнике объектно-ориентированное программирование изучается на языке Delphi.

В учебнике Полякова К. Ю. «Информатика. Углубленный уровень» для 11 класса [32] на раздел «Алгоритмизация и программирование» при изучении информатики по полному углубленному курсу (4 часа в неделю) отводится 45 часов, из которых 12 часов выделено для объектно-ориентированного программирования, а на изучение раздела по сокращённому углублённому курсу (2 часа в неделю) отводится 18 часов без внедрения объектно-ориентированного программирования. Всего в учебнике 2 раздела, включающих 18 параграфов по алгоритмизации и программированию. Для обучения объектно-ориентированному программированию используется язык Lazarus, который во многом аналогичен Delphi.

В учебнике Угринович Н. Д. Информатика и информационные технологии. Учебник для 10-11 классов [40]. Раздел «Алгоритмизация и программирование» изучается в 10 классе, на него выделено 10 учебных часов, из которых 8,5 часов отводится для изучения объектно-ориентированного программирования. В учебнике рассматриваются языки: Visual Basic .NET, Visual C#, Lazarus.

В практикуме по программированию для 10-11 классов Макаровой Н.В. [22] объектно-ориентированное программирование изучается в среде Visual Basic, данный практикум рассчитан на 20 учебных часов. В практикуме

подробно рассматривается работа с графическим интерфейсом, основные элементы управления формами, а также понятия подпрограмм, циклов и списков.

Курс «Основы объектно-ориентированного программирования в Delphi», разработанный В. В. Кузнецовым [19], рассчитан на 34 часа и рассматривается разработчиком как продолжение курса «Программирование на Паскале». В данном курсе рассматриваются основные вопросы объектно-ориентированного программирования: наследование, полиморфизм, иерархия классов. Курс может быть реализован в школе с углубленным изучением информатики.

Таким образом, можно сделать вывод, что раздел «Объектно-ориентированное программирование» широко представлен в школьных учебниках информатики. Но во многих из них изучается язык Delphi, который значительно уступает по большинству показателей языку C#, поскольку одной из первых сред быстрой разработки стала среда Delphi, разработанная фирмой Borland в 1994 году. Сейчас же самая известная современная профессиональная система объектно-ориентированного программирования – Microsoft Visual Studio, поддерживающая несколько языков программирования, включая C#.

К примеру, в C# есть возможность описания переменных в коде программы и одновременная инициализация их, тогда как в Delphi такое недопустимо. Также возможна передача в метод переменного количества параметров. В Delphi отсутствие переменного количества параметров можно частично компенсировать либо использованием значений параметров по умолчанию, либо передачей в качестве параметра открытого массива, однако в первом случае все равно количество параметров ограничено и должно быть определено при написании соответствующей программы, во втором случае хоть и наблюдается сходство с C#, есть существенные ограничения: нельзя передать пустой массив; передаваемый массив должен быть создан и заполнен до вызова подпрограммы. Объекты Delphi не поддерживают механизм автоматического удаления, тогда как в C# не нужно думать об удалении списка объектов и его элементов, сам же список объектов создается и заполняется в

одном месте. А также в Delphi нет полей классов, индексаторов, делегатов и цикла foreach, что значительно затрудняет работу программиста и увеличивает код программы. Интереснейшая возможность C#, отсутствующая как в Delphi, так и в других наиболее популярных языках программирования (C++, Java) – это атрибуты, значения которых устанавливаются на стадии компиляции и в процессе выполнения программы могут быть только считаны, поле деятельности атрибутов различно: от отслеживания версий алгоритмов до контроля за совместимостью программ.

1.2 Методические особенности изучения раздела «Объектно-ориентированное программирование» в профильном курсе информатики старшей школы

В курсе информатики 10-11 классов раздел «Объектно-ориентированное программирование» подразумевает изучение основных понятий и принципов ООП, создание объектов, свойств, а также знакомство с различными объектно-ориентированными языками программирования и приобретение навыков разработки и отладки программ на изучаемом языке.

В учебнике Семакина И. Г. «Информатика. Углубленный уровень» для 11 класса [37] на раздел «Объектно-ориентированное программирование» предполагается 10 учебных часов. Программирование изучается на языке Delphi. Поурочное планирование данного раздела представлено на рисунке 1.

Тема	Всего часов	Теория (раздел учебника)	Задачи и опорные задания (подготовка к ЕГЭ)	Практикум, часть 2
6. Объектно-ориентированное программирование (ООП)				
6.1. Базовые понятия ООП	2	§ 2.4.1	Задачи к § 2.4.1	Раздел 16. Программирование. Работа 16.9
6.2. Система программирования Delphi	1	§ 2.4.2		
6.3. Этапы программирования на Delphi	2	§ 2.4.3	Задачи к § 2.4.3	Раздел 16. Программирование. Работа 16.10
6.4. Программирование метода статистических испытаний	2	§ 2.4.4	Задачи к § 2.4.4	Раздел 16. Программирование (ч. 2). Работа 16.10
6.5. Построение графика функции	3	§ 2.4.5	Задачи к § 2.4.5	Раздел 16. Программирование (ч. 2). Работа 16.11

Рисунок 1 - Поурочное планирование раздела «ООП» в учебнике И. Г. Семакина «Информатика. Углубленный уровень» для 11 класса

Основные понятия ООП, представленные в учебном материале: класс – тип данных, описываемый программистом и категория объектов, обладающих одинаковым, набором свойств и методов воздействия на объекты; объект – экземпляр определенного класса с конкретными значениями свойств. Элементы класса – поля и методы. Поля – переменные величины (простые или структурные), методы – процедуры и функции, работающие с полями этого класса. Процесс объединения в единую структуру данных (полей) и действий над этими данными (методов) называется объектно-ориентированной декомпозицией. Реализация полей и методов объекта скрыта от других объектов, взаимодействующих с ним. Этот принцип ООП называется инкапсуляцией. Инкапсуляция исключает возможность изменения полей другими способами, кроме методов данного класса. Инкапсуляция – первый базовый принцип ООП. Далее приводится пример программы вычисления длины отрезка прямой на языке Object Pascal, в которой используется тип данных «класс».

Вторым базовым принципом ООП является наследование. Наследование – способ создания новых классов в качестве наследников уже существующих. Данный принцип иллюстрируется на примере базового класса четырехугольников и класса-потомка квадратов, где квадрат является частным случаем четырехугольника.

Третий базовый принцип ООП – полиморфизм (многообразие). Элементы классов с одинаковым интерфейсом могут иметь разную реализацию. В приведенном в учебнике примере полиморфизм проявляется в том, что функция с одним и тем же именем определена дважды, по-разному: сначала в родительском классе, а затем в классе-потомке. В результате площадь произвольного четырехугольника и площадь квадрата будут вычисляться по разным алгоритмам.

Практическое решение задач автор предлагает выполнять в системе программирования Delphi, предназначенной для создания объектно-ориентированных приложений Windows путем использования визуальной технологии программирования. Визуальная среда Delphi относится к средам категории RAD (Rapid Application Development – среда быстрой разработки приложений). Основной подход к разработке программного обеспечения в подобных средах заключается в использовании стандартных визуальных компонентов – заранее подготовленных классов, которые программист подключает к программе, помещая их в свой проект. Следующими описываются основные компоненты среды программирования Delphi: строка заголовка, строка главного меню, окно конструктора форм, окно программного кода, окно элементов управления, окно инспектора объектов, окно менеджера проектов и окно дерева объектов.

Проектом на Delphi называется весь комплекс модулей и ресурсов, из которого создается исполняемая программа. Проект включает программные модули, описания экранных форм, графических ресурсов, общих параметров создания программы и т.д. Форма – базовый графический объект Delphi, на основе которого создается программный проект. Элементы управления –

классы объектов графического интерфейса проекта, вкладываемых в форму (метки, кнопки, окна редактирования и пр.). Обладают набором свойств и событий. Методы – процедуры обработки событий, которые составляются программистом. Программы, создаваемые в системе Delphi, называются событийно-управляемыми. Это значит, что выполнение различных задач, решаемых программой, инициируется определенными событиями.

Практические задания учебника предполагают разработку учащимися не только консольных приложений без графического интерфейса, но и оконных – событийно управляемых приложений Delphi с графическим интерфейсом. Это требует от обучаемых выполнения этапов проектирования, конструирования интерфейса и программирования обработки событий.

Метод статических испытаний – численный метод, использующий моделирование случайных величин и получение статических оценок искомых величин. Одно из применений этого метода – вычисление площадей фигур и объемов тел. Пример метода статических испытаний продемонстрирован на программе вычисления числа π.

В демонстрационном примере построения графика функции для получения графических изображений используется объект Canvas (холст) – свойство формы, реализованное как объект. Рисование на холсте производится с помощью команд изображения графических примитивов или закрашиванием точек графической сетки посредством установки значения свойства Pixels [x, y] := z, где x, y – координаты пикселя, z – цвет пикселя. Для рисования графика функции строятся оси системы координат, осуществляется разметка осей, изменяется с некоторым шагом значение аргумента и вычисляется соответствующее значение функции, пересчитываются значения x и y из системы координат в экранную систему, рисуется точка на экране в соответствующей позиции.

В учебнике Полякова К. Ю. «Информатика. Углубленный уровень» для 11 класса [32] на изучение объектно-ориентированного программирования выделено 12 часов. Для обучения используется язык Lazarus, который во

многом аналогичен Delphi. Поурочное планирование данного раздела представлено на рисунке 2.

Номер урока	Тема урока	Параграф учебника (номер, название)	Работы компьютерного практикума (источник, номер, название)	Кол-во часов
82.	Введение в объектно-ориентированное программирование	§ 42. Введение		1
83.	Создание объектов в программе	§ 43. Создание объектов в программе	ПР № 58. Движение по дороге	1
84.	Скрытие внутреннего устройства	§ 44. Скрытие внутреннего устройства	ПР № 59. Скрытие внутреннего устройства	1
85.	Иерархия классов	§ 45. Иерархия классов		1
86.	Классы логических элементов	§ 45. Иерархия классов	ПР № 60. Классы логических элементов	1
87.	Программы с графическим интерфейсом	§ 46. Программы с графическим интерфейсом		1
88.	Графический интерфейс: основы	§ 47. Графический интерфейс: основы	ПР № 61. Работа с формой	1
89.	Использование компонентов (виджетов)	§ 48. Использование компонентов (виджетов)	ПР № 62. Просмотр рисунков	1
90.	Ввод данных	§ 48. Использование компонентов (виджетов)	ПР № 63. Ввод данных	1
91.	Совершенствование компонентов	§ 49. Совершенствование компонентов	ПР № 64. Совершенствование компонентов	1
92.	Модель и представление	§ 50. Модель и представление		1
93.	Вычисление арифметических выражений	§ 50. Модель и представление	ПР № 65. Калькулятор	1

Рисунок 2 - Поурочное планирование раздела «ООП» учебника К. Ю. Полякова «Информатика. Углубленный уровень» для 11 класса

Как одно из основных при введении в ООП вводится понятие абстракции – выделение существенных характеристик объекта, отличающих его от других объектов. Программирование, основанное на моделировании задачи реального мира как множества взаимодействующих объектов, принято называть объектно-ориентированным программированием.

Для того чтобы построить объектную модель, нужно выделить взаимодействующие объекты, с помощью которых можно достаточно полно

описать поведение моделируемой системы; определить свойства объектов, существенные в данной задаче; описать поведение (возможные действия) объектов, т.е. команды, которые объекты могут выполнить. Этап разработки модели, на котором решаются перечисленные выше задачи, называется объектно-ориентированным анализом (ООА).

Основные понятия ООП представлены в учебнике следующими определениями: объект – это процесс или явление, имеющий четкие границы и обладающий состоянием и поведением; класс – множество объектов, имеющих общую структуру и общее поведение. Данные понятия демонстрируются на примере наглядной образной задачи, где изучается движение автомобилей на шоссе для определения его пропускной способности.

Понятие «метод» рассмотрено как процедура или функция, принадлежащая классу объектов; поле – переменная, принадлежащая объекту. Значения полей описывают состояние объекта, а методы – его поведение. Конструктор – метод класса, который вызывается для создания объекта этого класса. Начальные значения полей можно задавать прямо при создании объекта. Для этого нужно добавить в описание класса новый конструктор. Таким образом, класс выполняет роль «фабрики», которая «выпускает» (создает) объекты «по чертежу» (описанию класса) при вызове конструктора.

Скрытие внутреннего устройства объектов называют инкапсуляцией. Такой подход позволяет обезопасить внутренние данные (поля) объекта от изменений со стороны других объектов; проверять данные, поступающие от других объектов, на корректность, тем самым повышая надежность программы; переделывать внутреннюю структуру и код объекта любым способом, не меняя его внешние характеристики (интерфейс), при этом никакой переделки других объектов не требуется.

В теме «Скрытие внутреннего устройства» подробно рассмотрены приемы использования спецификаторов доступности. По умолчанию все члены класса (поля и методы) – открытые (public), общедоступные. Те элементы, которые нужно скрыть, в описании класса помещают в «частный» раздел

(private). Также на примере демонстрируется применение методов аксессоров get и set. Свойство – способ доступа к внутреннему состоянию объекта, имитирующий обращение к его внутренней переменной. Таким образом, доступ к внутренним данным объекта возможен, как правило, только с помощью методов. Применение свойств очень удобно, потому что позволяет использовать ту же форму записи, что и при работе с общедоступной переменной объекта.

Тема «Иерархия классов» содержит большое количество примеров для иллюстрации одного из важнейших принципов ООП – наследование. Суть принципа раскрывается через приемы классификации – раздел изучаемых объектов на группы (классы), объединенные общими признаками, а также с подклассы – классы-наследники и базовые классы – классы-предки.

Парадигма объектно-ориентированного программирования описана автором как подход к программированию, при котором программа представляет собой множество взаимодействующих объектов, каждый из которых является экземпляром определенного класса, а классы образуют иерархию наследования.

Принцип полиморфизма – возможность классов-наследников по-разному реализовать метод, описанный для класса-предка, также представлен в учебнике на большом количестве демонстрационных примеров. Абстрактный метод – это метод класса, который объявляется, но не реализуется в классе. Абстрактный класс – это класс, содержащий хотя бы один абстрактный метод. Виртуальный метод – метод базового класса, который могут переопределить классы-наследники, при этом конкретный адрес вызываемого метода определяется только при выполнении программы.

При выявлении особенностей современных прикладных программ автор учебника вводит понятия; «сообщение» – блок данных определенной структуры, который используется для обмена информацией между объектами, и «событие» – переход какого-либо объекта из одного состояния в другое. Большое внимание уделяется RAD-средам для разработки программ. Для

выполнения практических работ в учебнике предлагается использование системы программирования на языке Lazarus, в среде которой учащиеся последовательно выполняют этапы реализации простейшей программы, свойства объектов и обработчики событий.

Одна из важнейших идей технологии быстрого проектирования программ – повторное использование написанного ранее готового кода. Чтобы облегчить решение этой задачи, было предложено использовать еще одну декомпозицию: разделить модель, т.е. данные и методы их обработки, и представление – способ взаимодействия модели с пользователем (интерфейс).

В учебнике Угринович Н. Д. «Информатика. 10 класс. Базовый уровень» [40] раздел «Объектно-ориентированное программирование» изучается в 10 классе, на него выделено 8,5 учебных часов. В учебнике рассматриваются языки: Visual Basic .NET, Visual C#, Lazarus. Поурочное планирование данного раздела представлено на рисунке 3.

Содержание учебного курса		Урочная часть (ч)		Внеурочная часть (ч)	
Параграфы учебника (теория)	Практические работы к параграфам учебника	теория	практика	теория	практика
4.3. Введение в объектно-ориентированное программирование 4.3.1. Объекты: свойства и методы 4.3.2. События 4.3.3. Проекты и приложения		1	1		
4.4. Система объектно-ориентированного программирования Microsoft Visual Studio 4.4.1. Интегрированная среда разработки языков Visual Basic .NET и Visual C#	**Практическая работа 4.1** Создание проекта «Консольное приложение»	1	1	1	2
4.5. Система объектно-ориентированного программирования Lazarus			1	1	2
4.6. Переменные в языках объектно-ориентированного программирования		0,5			
4.7. Графический интерфейс	**Практическая работа 4.2** Создание проекта «Переменные»	0,5	0,5	1	1
	Практическая работа 4.3 Создание проекта «Отметка»	0,5	0,5	1	1
	Практическая работа 4.4 Создание проекта «Перевод целых чисел»	0,5	0,5	1	1

Рисунок 3 - Поурочное планирование раздела «ООП» учебника Н. Д. Угринович «Информатика. 10 класс. Базовый уровень»

Основной единицей в объектно-ориентированном программировании является программный объект, который объединяет в себе как описывающие его свойства, так и действия объекта (процедуры) – методы. Программные объекты обладают свойствами, имеют методы и для них можно описать реакцию на события. Классы объектов являются «шаблонами», определяющими наборы свойств, методов и событий, по которым создаются объекты. Основными классами объектов являются объекты, реализующие графический интерфейс проектов. Также автор упоминает о точечной нотации (dot-запись), при которой имена объектов, свойств и методов отделяются друг от друга знаком точки.

События представляют собой изменение некоторого состояния, распознаваемое объектом. Для реакции на это изменение могут быть описаны те или иные методы – обработчики, обрабатывающие события в программном коде. Обработчик события представляет собой процедуру, которая начинает выполняться после реализации определенного события.

С одной стороны, система объектно-ориентированного визуального программирования является системой программирования, так как позволяет кодировать алгоритмы на данном языке, а с другой стороны, система объектно-ориентированного визуального программирования является средой проектирования, так как позволяет осуществлять визуальное конструирование графического интерфейса.

Проект включает в себя программные модули форм и самостоятельные программные модули в виде отдельных файлов. Проект может быть запущен на выполнение только из системы объектно-ориентированного программирования. Проекты объединяются в решения, которые включают один или несколько проектов. Решение создается автоматически при создании нового проекта.

Не менее важными компонентами разработки приложений являются трансляторы языков программирования: интерпретаторы и компиляторы. Для преобразования проекта в приложение необходимо выполнить компиляцию проекта, в процессе которой приложение сохраняется в исполняемом файле с

расширением exe. Приложение интегрирует программный код и графический интерфейс в одном исполняемом файле, который может запускаться непосредственно в операционной системе.

При выполнении учащимися практических учебных проектов и приложений автор учебника использует методику поэтапной разработки:

1) создание графического интерфейса проекта;
2) установка значений свойств объектов графического интерфейса;
3) создание и редактирование программного кода;
4) сохранение проекта;
5) компиляция проекта в приложение.

Тема «Интегрированная среда разработки Visual Studio и языки программирования Visual Basic .NET и Visual C#» содержит полное описание функциональных возможностей системы, приемов их использования и производит ее настройки. Систему объектно-ориентированного программирования Lazarus автор учебника также рассматривает в качестве инструментария выполнения практических заданий по теме.

Тема «Переменные в языках объектно-ориентированного программирования» включает такие понятия, как переменная, тип переменной, правила написания имени переменной, ее объявление и область действия (локальная или глобальная), а также присваивание переменным значений.

Система объектно-ориентированного программирования позволяет визуализировать процесс создания графического интерфейса разрабатываемого проекта, рассмотрены понятия формы, элементов управления. Приведено сравнение названий элементов управления трех языков программирования: Visual Basic .NET, Visual C# и Lazarus, набор элементов управления данных языков программирования практически совпадает.

1.3 Электронный учебно-методический комплекс по разделу «Объектно-ориентированное программирование» как компонент школьной информационно-образовательной среды

В системе образования все больше используются информационные ресурсы и технологии, и все меньше — печатные информационные ресурсы, такие как учебные пособия, методические пособия, словари, энциклопедии и т.д. Повсеместное распространение информационных ресурсов и технологий в различных сферах деятельности общества требует новых подходов к проектированию образовательной среды. Используя автоматизированные системы управления и машинное обучение, на основе анализа информации преподаватели могут обеспечить эффективный подход к вопросам повышения качества образования.

Информационная образовательная среда — сложная система, включающая в себя интеллектуальные, культурные, программно-методические, организационные и технические ресурсы и обеспечивающая формирование гармонично развитой личности учащегося. Внедрение и использование цифровых образовательных ресурсов обеспечивает информационно-методическую поддержку учебного процесса, своевременный, системный мониторинг и анализ результатов образовательного процесса, дистанционное взаимодействие преподавателя и учащихся.

В образовании все интенсивнее развиваются электронные средства обучения (ЭСО), содержащее систематизированный материал по соответствующей научно-практической области знаний, обеспечивающее творческое и активное овладение учащимися знаниями, умениями и навыками в этой области. ЭСО должно отличаться высоким уровнем исполнения и художественного оформления, полнотой информации, качеством методического инструментария, качеством технического исполнения, наглядностью, логичностью и последовательностью изложения. Благодаря специфике своего определения, ЭСО существенно повышают качество

визуальной и аудиоинформации, она становится ярче, красочнее, динамичнее. Огромными возможностями обладают в этом плане современные технологии мультимедиа. Разные виды ЭСО имеют свою специфику создания, назначения и использования. Основными видами ЭСО являются: электронные учебно-методические комплексы (ЭУМК); электронные курсы; электронные учебники (ЭУ); автоматизированные обучающие системы (АОС); программные средства для контроля и измерения уровня знаний, умений и навыков обучающихся; электронные тренажеры; программные средства лабораторий удаленного доступа и виртуальных лабораторий; экспертные обучающие системы (ЭОС); интеллектуальные обучающие системы (ИОС).

В ряду ЭСО особое значение имеют электронные учебно-методические комплексы. Если традиционный учебно-методический комплекс – это система нормативной и учебно-методической документации, средств обучения и контроля, необходимых и достаточных для качественной организации основных и дополнительных образовательных программ, согласно учебного плана. Возможности ЭУМК значительно шире. С позиций системотехники электронный учебно-методический комплекс (ЭУМК) – это автоматизированная информационная система (АИС) учебного назначения, которая на новом качественном уровне обеспечивает непрерывность и полноту дидактического цикла процесса обучения и содержит организационные и систематизированные теоретические, практические, контролирующие материалы, построенные на принципах интерактивности, информационной открытости, дистанционности и формализованности процедур оценки знаний. Структура ЭУМК должна включать в себя, помимо обучающего блока, блока контроля и оценки знаний, блок обо всех компонентах учебной дисциплины, которые входят в состав рабочей программы, для планирования образовательной траектории и расписания обучающихся.

И. А. Позанова [29] выделяет в зависимости от масштаба охватываемой предметной области электронные учебно-методические комплексы по дисциплинам (ЭУМКД) и по специальности (или направлению) (ЭУМКС). В

плане функционирования электронный учебно-методический комплекс имеет обеспечивающую и функциональную части.

В состав обеспечивающей части входит:

1) информационное обеспечение — это совокупность проектных решений по объемам, размещению, формам организации учебной и методической информации (ФГОС для данной специальности; рабочие программы; фондовые лекции; учебные пособия для отработки практических и лабораторных заданий; перечни выносимых на зачет и экзамен учебных вопросов; тесты промежуточного контроля остаточных знаний; учебные и учебно-методические пособия; список рекомендованной литературы, адреса веб-сайтов в сети Интернет);
2) техническое обеспечение — комплекс технических средств, предназначенных для обеспечения его работы, а также соответствующая документация на эти средства и технологические процессы (мультимедийные проекторы; интерактивные доски; системы видеоконференций; компьютерные тренажеры; средства компьютерной техники; компьютерные сети и устройства для подключения компьютеров к ним; средства для оперативной печати (копирования) раздаточного материала);
3) математическое и программное обеспечение ЭУМК — это совокупность математических методов, моделей, алгоритмов, используемых в учебных целях для решения задач, а также системные и специальные программные продукты, прикладное программное обеспечение и техническая документация к ним;
4) методическое и организационное обеспечение ЭУМК — это совокупность средств и методов, средств и документов, регламентирующих взаимодействие преподавателя и ЭУМК, обучаемого и преподавателя, обучаемого и ЭУМК в этапах его разработки и использования в учебном процессе.

Функциональная часть ЭУМК должна определяться теми задачами, для которых он разрабатывается:

1) для оказания методической помощи преподавателям при подготовке и проведении занятий по данной дисциплине;

2) как средство комплексного воздействия на обучаемого путём сочетания концептуальной, иллюстративной, справочной, тренажерной и контролирующей частей.

Структура и пользовательский интерфейс обеспечивающей и функциональной частей ЭУМК должны обеспечить эффективную помощь преподавателю для организации учебного процесса и обучаемому при изучении дисциплины. Использование ЭУМК в образовательном процессе дает педагогам дополнительные дидактические возможности:

1) обратную связь между пользователем и ЭСО, что позволяет обеспечить интерактивный диалог;

2) компьютерную визуализацию учебной информации, предполагающую реализацию возможностей современных средств визуализации объектов, процессов, явлений (как реальных, так и виртуальных), а также их моделей, представление их в динамике;

3) компьютерное моделирование изучаемых объектов, явлений, процессов;

4) автоматизацию процессов вычислительной и информационно-поисковой деятельности;

5) автоматизацию процессов управления учебной деятельностью и контроля за результатами усвоения материала.

Таким образом, необходимо отметить, что использование разных видов ЭСО в образовательном процессе значительно влияет на формы и методы представления учебного материала, характер взаимодействия между обучаемым и педагогом и, соответственно, на методику проведения занятий. Вместе с тем ЭСО не заменяют традиционные подходы к обучению, а значительно повышают их эффективность. Любой из типов уроков может быть проведен с использованием ЭСО. Главное для педагога — найти соответствующее место ЭСО в образовательном процессе. Создание ЭУМК имеет особое значение, так как позволяет комплексно подходить к решению основных дидактических задач с использованием информационных ресурсов и

ЭСО. ЭУМК - это инновационный образовательный продукт, обладающий новыми дидактическими возможностями.

Выводы по главе 1

Подводя итог теоретической части исследования можно сделать несколько обобщающих выводов.

В результате всестороннего анализа научно-педагогических и методических основ обучения объектно-ориентированному программированию в школьном курсе информатики было выявлено, что раздел «Объектно-ориентированное программирование» представлен как обязательный в Федеральном государственном образовательном стандарте среднего образования [1], в примерной образовательной программе среднего общего образования [2], в учебниках И. Г. Семакина [37], К. Ю. Полякова [32], Н. В. Макаровой [22], Н. Д. Угринович [40], В. В. Кузнецова [19].

Анализ методических особенностей изучения раздела «Объектно-ориентированное программирование» в профильном курсе информатики средней школы позволил выделить и определить основные понятия объектно-ориентированного программирования, особенности его реализации и требования ФГОС СОО к предметным и метапредметным результатам обучения учащихся в среде современных объектно-ориентированных языков.

Глава 2 Проектирование и реализация ЭУМК по разделу «Объектно-ориентированное программирование»

2.1 Дидактические, программно-технологические и технические характеристики ЭУМК

Для проектирования и реализации ЭУМК по объектно-ориентированному программированию необходимо проанализировать целевую аудиторию, выполнить постановку целей и задач ЭУМК, определить его структуру и содержание, выбрать формы и средства представления учебных материалов.

На изучение объектно-ориентированного программирования в соответствии с ЭУМК отводится 10 часов в профильном курсе информатики старшей школы.

Данный комплекс направлен на развитие познавательных интересов, алгоритмического мышления, а также памяти и логики учащихся. Комплекс имеет практическую направленность – учащимся предоставляется возможность познакомиться на практике с основами разработки консольных и оконных приложений на объектно-ориентированном языке C#. Данные занятия помогут развить навыки работы на компьютере, закрепить знания основных понятий ООП и повысить интерес школьников к информатике и к профессиям, связанным с IT-сферой.

Содержание ЭУМК было разработано в соответствии с учебниками Семакина И. Г. «Информатика. Углубленный уровень» [37], Полякова К. Ю. «Информатика. Углубленный уровень» [32], Угриновича Н. Д. «Информатика и информационные технологии» [40]. Также использованы методические рекомендации, представленные в работах Полякова К. Ю. [31], Полежаевой О.А. [30]; Красильниковой В.А. [17].

Цель ЭУМК по объектно-ориентированному программированию – это систематизация и закрепление у обучаемых полученных знаний по объектно-ориентированному программированию, знакомство с синтаксисом языка C# и его семантикой, развитие умений анализировать программы на языке

программирования C# и разрабатывать многокомпонентные программные продукты в среде программирования Visual Studio.

Задачи ЭУМК:

1) познакомить учащихся с базовыми понятиями ООП и основными конструкциями программирования;
2) дать представление о последовательности разработки консольных и оконных приложений;
3) научить программировать консольные приложения и конструировать интерфейс оконного приложения с применением основных принципов ООП;
4) способствовать повышению интереса к информатике и программированию;
5) способствовать формированию информационной культуры учащихся.

Предметные результаты освоения ЭУМК по объектно-ориентированному программированию:

учащийся должен знать:

1) базовые понятия ООП;
2) основные конструкции программирования;
3) основные спецификаторы класса;
4) основные алгоритмы обработки числовой и текстовой информации;
5) последовательность разработки консольных и оконных приложений;
6) принципы и механизм наследования;
7) основные элементы интерфейса системы программирования Visual Studio;

учащийся должен уметь:

8) описывать элементы класса;
9) использовать основные управляющие конструкции;
10) применять спецификаторы доступности;
11) получать доступ к закрытым данным класса;
12) использовать приемы разработки оконного приложения;
13) конструировать интерфейс оконного приложения;
14) программировать процедуры методов обработки событий;
15) разрабатывать дочерние классы;

16) понимать программы, написанные на выбранном для изучения универсальном алгоритмическом языке высокого уровня;

учащийся должен владеть:

17) универсальным языком программирования высокого уровня;
18) навыками разработки программ в среде программирования Visual Studio;
19) навыками тестирования и отладки программ.

Метапредметные результаты:

1) умение самостоятельно определять цели и составлять планы деятельности, а также выбирать успешные стратегии в различных ситуациях для достижения целей;
2) владение навыками познавательной, учебно-исследовательской и проектной деятельности, навыками разрешения проблем; способность и готовность к самостоятельному поиску методов решения практических задач, применению различных методов познания.

Личностные результаты:

1) сформированность мировоззрения, соответствующего современному уровню развития науки и общественной практики;
2) готовность и способность к образованию, в том числе самообразованию, на протяжении всей жизни; сознательное отношение к непрерывному образованию как условию успешной профессиональной и общественной деятельности; осознанный выбор будущей профессии и возможностей реализации собственных жизненных планов.

2.2 Педагогический и технологический сценарий ЭУМК

Процесс разработки электронных образовательных ресурсов состоит из двух основных этапов:

На первом подготовительном этапе производится:

1) отбор содержания учебных материалов;
2) структуризация и систематизация учебного материала;
3) разработка педагогического и технологического сценариев;

4) оформление текстовых материалов в цифровом формате и формирование основных разделов;

5) разработка и реализация мультимедийных элементов (видеосюжеты, звуковое сопровождение, графические изображения).

На втором этапе компоновки производится сборка в единое целое всех отобранных и разработанных частей ЭОР (информационных, обучающих, контролирующих) для предъявления обучающимся в соответствии с задуманным автором сценарием.

Планирование педагогического сценария предполагает тщательное проектирование содержания учебной деятельности и использования педагогических технологий. Для решения этих задач на этапе проектирования необходимо подготовить развернутую программу, подобрать учебный материал, подготовить задания, проставить порог прохождения каждого из заданий, прописать формулу оценивания результатов по курсу, составить сценарии видеолекции и интерактивных упражнений, а также разработать методические рекомендации по изучению курса. Эти сведения представлены в виде таблицы 1 и рисунка 4.

Таблица 1 – Общие данные о ЭУМК

Название ЭУМК	Объектно-ориентированное программирование
Основная информация	Данный курс представлен пятью взаимосвязанными лабораторными работами, которые позволят Вам полностью разобраться с синтаксисом языка C# и его семантикой, а также освоить объектно-ориентированное программирование на языке C#. Изучение курса начинается с основных понятий и парадигм ООП. Будут рассмотрены понятия классов и особенности языка C#. По окончании курса Вы сможете понимать основные принципы построения и структурирования приложений, написанных на языке программирования C#, а также создавать полноценные последовательные алгоритмы в своих решениях. Онлайн курс даст Вам необходимый уровень знаний и навыков, Вы освоите базовые возможности языка программирования C#, и это станет хорошим фундаментом для изучения более сложных технологий.
Формат	Данный курс рассчитан на 5 недель: на выполнение каждой лабораторной работы выделяется 1 неделя.

Продолжение таблицы 1

Требования	Для прохождения курса необходимо: • уметь работать с платформой .NET • знать основные типы данных; • уметь вводить и выводить данные; • уметь использовать конструкции цикла; • уметь оперировать массивами.
Программа ЭУМК	1. Введение 2. Классы и объекты 3. Поля и методы класса 4. Конструкторы и свойства 5. Механизм наследования 6. Разработка графического интерфейса пользователя
Результаты обучения	Вы научитесь создавать и использовать классы, методы и объекты, овладеете навыками и опытом разработки программ в среде программирования Visual Studio, а также навыками тестирования и отладки программ на объектно-ориентированном языке C#. По итогу курса Вы разработаете конкретный продукт – удобный календарь.
Информация о преподавателе	Артамонова Екатерина Евгеньевна, студент Поволжского православного института Электронная почта: ekaterina08artamonova@gmail.com Телефон: 89674931998

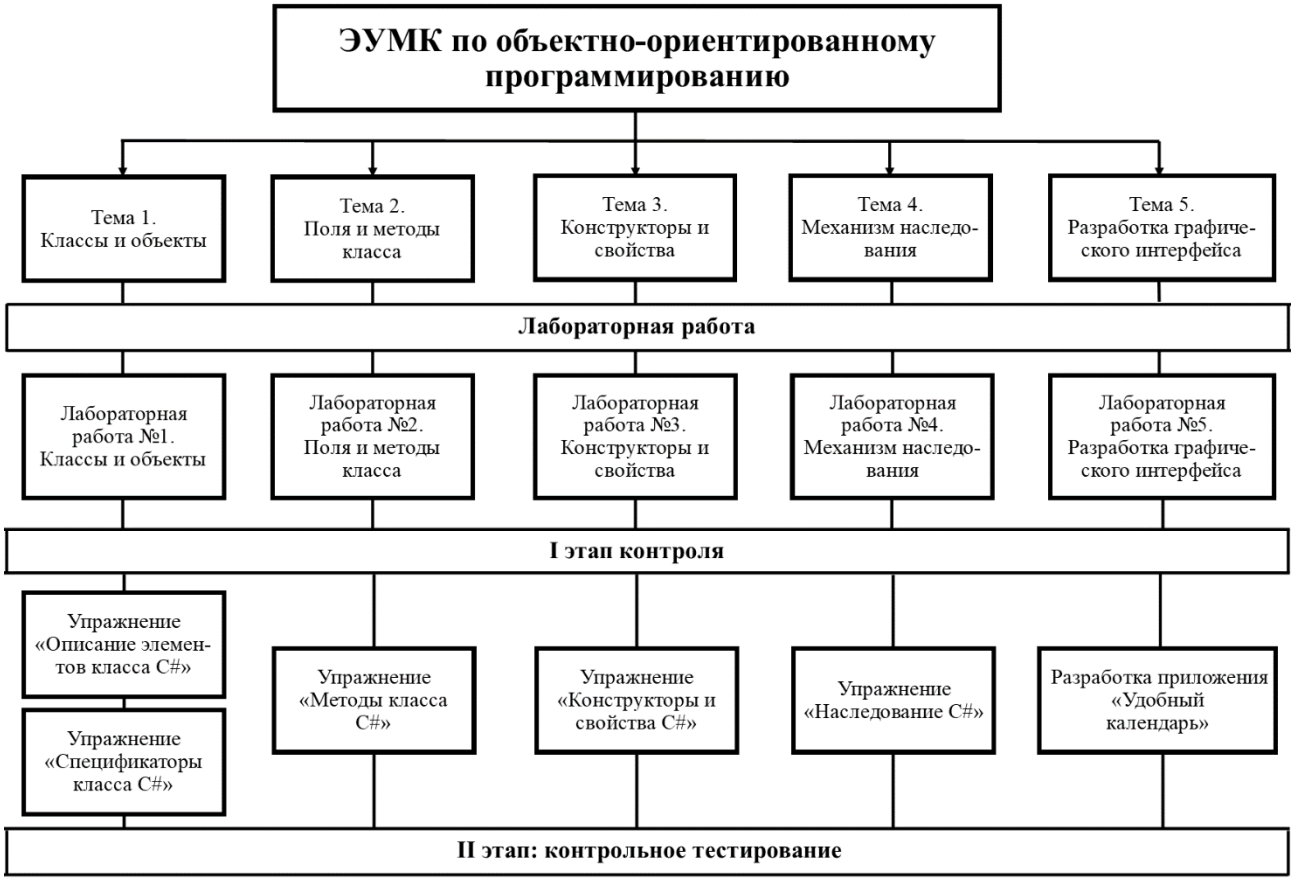

Рисунок 4 - Структура ЭУМК

Электронный учебно-методический комплекс представлен введением и пятью темами, включающими в себя теоретический материал, практические задания и лабораторные работы. В качестве дополнительных средств обучения в курсе используются авторские электронные образовательные ресурсы (ЭОР): видеолекция и интерактивные упражнения в облачном сервисе LearningApps [1]. ЭУМК реализован средствами системы управления обучением Moodle [2]. Разработка консольных и оконных приложений осуществляется в интегрированной среде разработки MS Visual Studio.

В таблице 2 представлено поурочное планирование обучения объектно-ориентированному программированию учащихся старших классов профильного уровня с использованием разработанных электронных образовательных ресурсов.

Таблица 2 – Поурочное планирование обучения объектно-ориентированному программированию учащихся старших классов профильного уровня

Тема урока	Контрольный этап	Лабораторная работа	Кол-во часов
Тема 1. Классы и объекты	Интерактивные упражнения «Описание элементов класса C#», «Спецификатор ы класса C#»	Лабораторная работа №1. Классы и объекты	2,5
Тема 2. Поля и методы класса	Интерактивное упражнение «Методы класса C#»	Лабораторная работа №2. Поля и методы класса	2,5
Тема 3. Конструкторы и свойства	Интерактивное упражнение «Конструкторы и свойства C#»	Лабораторная работа №3. Конструкторы и свойства	2,5
Тема 4. Механизм наследования	Интерактивное упражнение «Наследование C#»	Лабораторная работа №4. Механизм наследования	2,5
Тема 5. Разработка графического интерфейса пользователя	Разработка приложения «Удобный календарь» (видеоурок)	Лабораторная работа №5. Разработка графического интерфейса пользователя	3
Контрольное тестирование			1
		Всего часов	14

ЭУМК включает в себя три части: теоретическую, практическую и контрольную. Первая часть представлена в виде лекций. Каждая лекция содержит определенный набор определений и понятий.

Вторая часть организована в виде лабораторных работ, в ходе выполнения которых учащиеся не только формируют навыки и умения программирования на объектно-ориентированном языке C#, но и развивают алгоритмическое и логическое мышление. Предметом оценивания деятельности учащихся являются их готовые разработанные программы.

Третья часть представляет собой модуль контроля и диагностики знаний в форме двухэтапного контроля. На первом этапе учащиеся выполняют интерактивные упражнения на закрепление пройденного материала непосредственно после изучения соответствующей темы. Второй этап реализован в форме тестирования и проводится после изучения всех тем раздела «Объектно-ориентированное программирование».

Подробное описание каждого из модулей содержится в следующих параграфах.

2.3 Проектирование и реализация теоретико-познавательного модуля ЭУМК

Теоретико-познавательный модуль ЭУМК по объектно-ориентированному программированию организован в форме лекций и рассчитан на 5 учебных часов (таблица 3).

Таблица 3 – Структура теоретико-познавательного модуля ЭУМК

№	Тема	Описание	Кол-во часов
1	Классы и объекты	Лекция знакомит учащихся с понятиями класса и объекта, спецификаторами доступа классов и способами их применения, элементами класса и их спецификаторами. Также рассматривается механизм присваивания и сравнения.	1

Продолжение таблицы 3

2	Поля и методы класса	Лекция знакомит учащихся с полями и методами класса, элементами описания методов, их типами и параметрами. Также рассматриваются четыре типа параметров обмена данными между вызывающей и вызываемой функциями (параметры-значения, параметры-ссылки, выходные параметры и параметры массивы), их особенности и применение. В лекции освещена перегрузка методов и использование ключевого слова this.	1
3	Конструкторы и свойства	Лекция знакомит учащихся с конструкторами и свойствами, процессом создания собственных конструкторов, методами аксессоров get и set.	1
4	Механизм наследования	Лекция знакомит учащихся с механизмом наследования, иерархиями классов, понятиями родительский класс и дочерний, абстрактными классами.	1
5	Разработка графического интерфейса пользователя	Лекция знакомит учащихся с графическим интерфейсом пользователя, формами, типами элементов формы (ярлыки, кнопки, окна редактирования, поля рисунков) и их свойствами.	1

Цели и задачи теоретико-познавательного модуля ЭУМК.

Цель - сформировать представление об объектно-ориентированном программировании, о его ключевых принципах и понятиях, о технологии программирования основных конструкций ООП.

Задачи:

1) изучить базовые понятия ООП;
2) рассмотреть принципы и приемы разработки основных конструкций программирования ООП.

Изучив данный модуль, учащийся должен:

знать:

1) базовые понятия ООП;
2) основные конструкции программирования;
3) основные спецификаторы класса;
4) основные алгоритмы обработки числовой и текстовой информации;
5) последовательность разработки консольных и оконных приложений;
6) принципы и механизм наследования;

7) основные элементы интерфейса системы программирования Visual Studio;

уметь:

8) различать спецификаторы доступности;

9) различать типы параметров обмена данными между вызывающей и вызываемой функциями;

10) понимать программы, написанные на выбранном для изучения универсальном алгоритмическом языке высокого уровня.

Тема 1. Классы и объекты (приложение А).

Учебные вопросы:

1. Класс, объект.
2. Спецификаторы доступности.
3. Создание экземпляра класса.
4. Элементы класса.
5. Константы, поля, методы, свойства, конструкторы, деструкторы, индексаторы, операции, события, типы.
6. Спецификаторы элементов класса.
7. Механизм присваивания и сравнения.

Тема 2. Поля и методы класса.

Учебные вопросы:

1. Поля и методы класса.
2. Элементы описания методов класса.
3. Параметры-значения.
4. Параметры-ссылки.
5. Выходные параметры.
6. Параметры массивы.
7. Применение параметров обмена данными между вызывающей и вызываемой функциями.
8. Перегрузка методов.
9. Ключевое слово this.

Тема 3. Конструкторы и свойства.

Учебные вопросы:

1. Конструкторы.
2. Создание собственных конструкторов.
3. Свойства.
4. Метод аксессора get.
5. Метод аксессора set.

Тема 4. Механизм наследования.

Учебные вопросы:

1. Механизм наследования.
2. Иерархия классов.
3. Родительский класс.
4. Дочерний класс.
5. Абстрактный класс.

Тема 5. Разработка графического интерфейса пользователя.

Учебные вопросы:

1. Графический интерфейс пользователя.
2. Форма.
3. Элементы формы.
4. Свойства ярлыков.
5. Свойства кнопок.
6. Свойства окон редактирования.
7. Свойства полей рисунков.

При изучении теоретического материала основное внимание нужно уделить системе основных понятий темы.

Инкапсуляцией называется объединение в классе данных и подпрограмм для их обработки, а также скрытие внутреннего устройства объектов.

Наследование – это когда любой класс может быть порожден другим классом. Порожденный класс (наследник) автоматически наследует все поля, методы, свойства и события.

Полиморфизм позволяет использовать одинаковые имена для методов, входящих в различные классы.

Абстрагирование – это процесс выделения наиболее существенных характеристик некоторого объекта, отличающих его от всех других видов объектов, важных с точки зрения дальнейшего рассмотрения и анализа, и игнорирование менее важных или незначительных деталей.

Класс – это множество объектов, связанных общностью свойств, поведения, связей и семантики. Любой объект является экземпляром класса. Класс является типом данных, определяемым пользователем.

Объект – это структура данных, содержащая поля данных различных типов и заголовки методов и обобщающая структуру «Запись».

Метод – это функциональный элемент класса, который реализует вычисления или другие действия, выполняемые классом или экземпляром. Методы определяют поведение класса.

Свойство – это атрибут объекта, определяющий то, как объект выглядит или как он может себя вести.

Объект, созданный по «шаблону» класса объектов, являются экземпляром класса и наследует весь набор свойств, методов и событий данного класса.

Событие (Event) представляет собой действие, распознаваемое элементом управления.

Обработчик события – процедура, которая начинает выполняться после реализации определенного события.

Атрибут – признак или свойство, характеризующее объект.

Форма – объект, представляющий собой окно на экране, в котором размещаются объекты – элементы управления.

Элементы управления – объекты, являющиеся элементами графического интерфейса проекта и реагирующие на события, производимые пользователем или другими программными объектами.

Проект – результат процессов программирования и проектирования, который объединяет в себе программный код и графический интерфейс.

Приложение интегрирует программный код и графический интерфейс в одном исполнимом файле, который может запускаться непосредственно в операционной системе.

Консольное приложение – приложение не имеющее графического интерфейса.

Конструктор – это специальный метод, инициализирующий объект, содержащий виртуальные методы, он объявляется специально зарезервированным словом constructor. Конструктор инициализирует объект путем установления связи между объектом и специальной таблицей виртуальных методов, содержащей адреса кодов, реализующих виртуальные методы. Конструктор может также использоваться для инициализации полей данных объекта.

Деструктор – это специальный метод, освобождающий память «кучи» от динамических объектов. Он объявляется с использованием специально зарезервированного слова destructor.

Графический интерфейс пользователя (GUI) – программа, выводящая информацию с помощью форм и панелей, называемых окнами.

Все понятия вводятся последовательно, по мере их усложнения, причем каждое разъясняется на соответствующем демонстрационном примере.

Реализация теоретико-познавательного модуля ЭУМК была выполнена в виде гипертекстовых мультимедиа учебных материалов, расположенных в открытой системе управления обучением Moodle [2]. Каждая тема представлена в качестве модуля «Книга», который позволяет преподавателю создать многостраничный ресурс, подобный книге, с главами и подглавами. Книги могут содержать медиа-файлы, а также длинную текстовую информацию, которая может быть разбита на разделы.

2.4 Проектирование и реализация практического модуля

Практический модуль организован в форме лабораторных работ, в ходе которых учащиеся программируют консольные и оконные приложения.

Данный модуль рассчитан на 5 учебных часов. Структура модуля представлена в таблице 4.

Таблица 4 – Структура практического модуля ЭУМК

№	Название	Описание	Кол-во часов
1	Лабораторная работа № 1. Классы и объекты	Цель работы: познакомить учащихся с классами, объектами, их элементами и спецификаторами, научить создавать простые классы.	1
2	Лабораторная работа № 2. Поля и методы класса	Цель работы: познакомить учащихся с методами, их параметрами, перегрузкой методов.	1
3	Лабораторная работа № 3. Конструкторы и свойства	Цель работы: познакомить учащихся с конструкторами и инициализаторами, научить создавать собственные конструкторы и перегружать их, научить использовать в программах аксессоры get и set.	1
4	Лабораторная работа № 4. Механизм наследования	Цель работы: познакомить учащихся с принципом наследования, научить создавать базовый и производные классы.	1
5	Лабораторная работа № 5. Разработка графического интерфейса пользователя	Цель работы: познакомить учащихся с графическим интерфейсом пользователя и основными его элементами, научить разрабатывать оконное приложение.	1

Лабораторные работы предусматривают решение комплексных учебных задач, требующих от ученика применения как теоретических знаний, полученных при изучении раздела «Объектно-ориентированное программирование» курса информатики профильного уровня, так и практических навыков.

Цели и задачи практического модуля ЭУМК.

Цель – закрепление на практике знаний ключевых понятий и принципов объектно-ориентированного программирования, формирование умений и навыков программирования основных конструкций ООП, разработки консольных и оконных приложений.

Задачи:

1) закрепление знаний базовых понятий ООП;
2) закрепление знаний основных конструкций программирования;

3) формирование умений и навыков применения ключевых принципов ООП;

4) формирование умений и навыков разработки основных конструкций программирования ООП;

5) формирование умений и навыков понимания программ, написанных на выбранном для изучения универсальном алгоритмическом языке высокого уровня;

6) формирование умений и навыков разработки программ в среде программирования Visual Studio.

Изучив данный модуль, учащийся должен:

знать:

1) базовые понятия ООП;

2) основные конструкции программирования;

3) последовательность разработки консольных и оконных приложений;

4) основные элементы интерфейса системы программирования Visual Studio;

уметь:

5) описывать элементы класса;

6) использовать основные управляющие конструкции;

7) применять спецификаторы доступности;

8) осуществлять перегрузку методов;

9) получать доступ к закрытым данным класса;

10) создавать собственные конструкторы;

11) создавать базовые и дочерние классы;

12) конструировать интерфейс оконного приложения;

13) программировать процедуры методов обработки событий;

14) понимать программы, написанные на выбранном для изучения универсальном алгоритмическом языке высокого уровня;

владеть:

15) универсальным языком программирования высокого уровня;

16) навыками разработки программ в среде программирования Visual Studio;

17) навыками тестирования и отладки программ.

Методические рекомендации по выполнению лабораторной работы № 1. Классы и объекты.

Данная лабораторная работа подразумевает модификацию учеником программы «Школьник», предложенную в теме 1. Классы и объекты, таким образом, чтобы все исходные данные, а именно ФИО, возраст, класс, домашний адрес вводились с клавиатуры. Учащемуся необходимо будет использовать конструкцию «цикл» для того, чтобы была возможность обработать информацию для множества студентов.

Методические рекомендации по выполнению лабораторной работы № 2. Поля и методы класса.

Лабораторная работа подразумевает разработку учеником класса, содержащего различные поля, где все данные должны вводиться с клавиатуры. Учащемуся необходимо разработать несколько методов в соответствии с заданием и перегруженный метод вывода сведений на экран, а также необходимо предусмотреть возможность использования в программе конструкции «цикл» для обработки данных. В ходе лабораторной работы ученик напишет программу, демонстрирующую все разработанные им элементы класса.

Методические рекомендации по выполнению лабораторной работы № 3. Конструкторы и свойства.

Для выполнения лабораторной работы ученику необходимо доработать программу из лабораторной работы №2, а именно: разработать и включить в класс собственный конструктор, инициализирующий статические поля класса; модифицировать разработанный класс, используя спецификатор доступности private для защиты полей экземпляра и предоставить свойства для управления доступа к ним. Также учащемуся нужно включить в методы аксессора set процедуры проверки значений переменных экземпляра.

Методические рекомендации по выполнению лабораторной работы № 4. Механизм наследования.

В ходе выполнения лабораторной работы ученику необходимо модифицировать проект, созданный в лабораторной работе №3: разработать абстрактный базовый класс, создать не менее двух производных классов, включающих собственные конструкторы, переопределенные переменные, свойства и методы экземпляра.

Методические рекомендации по выполнению лабораторной работы № 5. Разработка графического интерфейса пользователя (приложение Б).

Ученику необходимо спроектировать графический интерфейс пользователя в соответствии с программой из предыдущей лабораторной работы. Программа должна воспринимать исходные данные из текстовых полей, а не с консоли. Нужно написать обработчики событий для всех управляющих элементов графического интерфейса.

Критерии оценки лабораторных работ:

«5» (отлично): задание лабораторной работы выполнено в полном объеме, учащийся продемонстрировал умение иллюстрировать теоретические положения ранее изученного материала.

«4» (хорошо): задание лабораторной работы выполнено в полном объеме, учащийся продемонстрировал умение иллюстрировать теоретические положения ранее изученного материала с замечаниями.

«3» (удовлетворительно): задание лабораторной работы выполнено не в полном объеме или с замечаниями, учащийся продемонстрировал умение иллюстрировать теоретические положения ранее изученного материала с замечаниями.

«2» (неудовлетворительно): задание лабораторной работы не выполнено или выполнено неправильно, учащийся не продемонстрировал умение иллюстрировать теоретические положения ранее изученного материала.

2.5 Проектирование и реализация контрольного модуля

Контрольный модуль реализован в форме двухэтапного контроля, рассчитанного на 4 учебных часа. Структура модуля представлена в таблице 5.

Первый этап представляет собой интерактивные упражнения в облачном сервисе LearningApps [1] и разработку графического интерфейса пользователя «Удобный календарь». Каждое задание данного этапа проводится непосредственно после изучения соответствующей темы и проверяет уровень усвоения пройденного материала.

Второй этап реализован в форме тестирования и проводится после изучения всех тем раздела «Объектно-ориентированное программирование». Тест состоит из 20 вопросов (приложение В), направленных на проверку усвоения знаний базовых понятий, принципов и основных конструкций объектно-ориентированного программирования.

Таблица 5 – Структура контрольного модуля ЭУМК

№	Форма контроля	Описание	Кол-во часов
1	Интерактивное упражнение «Описание элементов класса C#»	Цель задания: закрепление знаний об элементах класса в объектно-ориентированном языке C#.	0,5
2	Интерактивное упражнение «Спецификаторы класса C#»	Цель задания: закрепление знаний о видах спецификаторов класса и их функциях.	
3	Интерактивное упражнение «Методы класса C#»	Цель задания: закрепление знаний о методах класса и способах их объявления.	0,5
4	Интерактивное упражнение «Конструкторы и свойства C#»	Цель задания: закрепление знаний о конструкторах, позволяющих получать доступ к закрытым типам данных, и их свойствах.	0,5
5	Интерактивное упражнение «Наследование C#»	Цель задания: закрепление знаний о принципах наследования и их осуществлениях.	0,5
6	Разработка приложения «Удобный календарь» (видеоурок)	Цель задания: закрепление знаний о создании графического интерфейса пользователя программы для работы в операционной системе Windows.	1
7	Контрольное тестирование		1

Интерактивные упражнения - это различные задания, направленные на закрепление знаний в игровой форме. Данный тип ЭОР способствует формированию познавательного интереса учащихся, мотивирует их к обучению и обеспечивает чередование учебной деятельности, что позволяет снизить усталость и напряжение.

Для разработки интерактивных упражнений был выбран веб-сервис LearningApps. Сервис LearningApps.org создан с целью поддержки учебного процесса с помощью интерактивных приложений. Данный сервис позволяет в режиме онлайн создавать и использовать интерактивные задания самых разных видов: викторины, вставка пропусков в текст, кроссворды и игры с буквами на составление слов, подбор пары и многое другое. Такие упражнения направлены на проверку и закрепление знаний в игровой форме. На сервисе имеется коллекция шаблонов различных приложений, используя которые, можно создать свое уникальное интерактивное задание. Помимо этого, созданные интерактивные модули можно вставлять в свои сетевые блоги, сайты или скачивать в виде готового SCORM-модуля для проигрывания в системе дистанционного обучения.

Задание «Разработка приложения «Удобный календарь» представлено в форме видеоурока (приложение Г). Видеоурок — это дистанционная форма обучения, процесс передачи знаний при помощи мультимедийных технологий. В отличие от обычной формы урока, видеоурок является более увлекательным процессом обучения. В качестве инструментария разработки видеоурока была использована среда видеоредактора FastStone Capture. FastStone Capture - полнофункциональный инструмент записи видео как экрана целиком, так и отдельных его областей. Программа имеет полный набор функций, необходимый для записи видеоуроков. Интерфейс программы FastStone Capture представлен на рисунке 5.

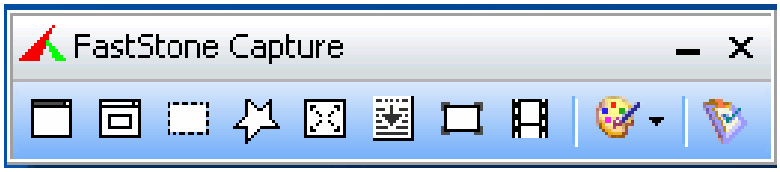

Рисунок 5 - Интерфейс видеоредактора FastStone Capture

Интерактивное упражнение «Описание элементов класса C#» (приложение Д).

Учебные вопросы:

1) класс;

2) элементы класса;

3) спецификаторы элементов класса;

4) открытое поле;

5) статическое поле.

Методические рекомендации по выполнению:

1. Перед выполнением задания повторите принцип объявления класса и спецификаторы доступа.

2. Впишите в пропуски пропущенные части кода.

Интерактивное упражнение «Спецификаторы класса C#» (приложение Е).

Учебные вопросы:

1) спецификаторы элементов класса;

2) новое описание класса;

3) неограниченный доступ;

4) ограниченный доступ;

5) доступ из данной программы;

6) закрытый доступ;

7) статический класс;

8) абстрактный класс.

Методические рекомендации по выполнению:

1. Перед выполнением задания повторите спецификаторы класса и их описание.

2. Соотнесите определение спецификатора класса с его названием. Если соотнесено верно – карточка пропадет с поля.

Интерактивное упражнение «Методы класса C#» (приложение Ж).

Учебные вопросы:

1) метод;

2) класс;

3) функция ввода информации;

4) функция вывода информации.

Методические рекомендации по выполнению:

1. Перед выполнением задания повторите принцип объявления класса и метода.

2. Расставьте строки кода в правильном порядке. Комментарий должен идти после кода, к которому относится.

Интерактивное упражнение «Конструкторы и свойства C#».

Учебные вопросы:

1) конструктор;
2) свойства;
3) метод аксессора get;
4) метод аксессора set.

Методические рекомендации по выполнению:

1. Перед выполнением задания повторите определения конструктора и свойств.
2. Прочитайте определение и назовите понятие по буквам, отгадывая каждую букву отдельно. Количество попыток ограничено.

Интерактивное упражнение «Наследование C#».

Учебные вопросы:

1) механизм наследования;
2) родительский класс;
3) дочерний класс;
4) абстрактный класс.

Методические рекомендации по выполнению:

1. Перед выполнением задания повторите понятия базового, дочернего и абстрактного классов и способы их объявления.
2. Отсортируйте описание классов по двум категориям: базовый класс, дочерний класс.

Разработка приложения «Удобный календарь» (видеоурок).

Учебные вопросы:

1) графический интерфейс пользователя;
2) форма;
3) ярлык;
4) кнопка;
5) окно редактирования;

6) поле рисунка.

Методические рекомендации по выполнению:

1. Внимательно посмотрите обучающий видеоурок.
2. Заранее задайте размеры формы.
3. Задайте все необходимые свойства формы и ее элементов.

Оценивание знаний в форме тестирования выполняется по 5-бальной шкале:

Оценка «5» (отлично): 18 - 20 верных ответов.

Оценка «4» (хорошо): 14 - 17 верных ответов.

Оценка «3» (удовлетворительно): 9 - 13 верных ответов.

Оценка «2» (неудовлетворительно): 0 - 8 верных ответов.

Итоговая оценка результатов прохождения курса формируется по накопительной балльно-рейтинговой системе оценивания, учитывается выполнение лабораторных работ и всех форм контроля.

Выводы по главе 2

При проектировании и реализации программно-методического обеспечения обучения разделу «Объектно-ориентированное программирование» был проведен анализ дидактических, программно-технологических и технических характеристик каждого модуля, входящего в ЭУМК, выполнена постановка целей и задач обучения, определены предметные и межпредметные результаты учащихся при освоении каждого образовательного ресурса.

ЭУМК состоит из трех частей: теоретической, практической и контрольной.

В процессе изучения теоретической части ЭУМК у учащихся должно сформироваться представление об объектно-ориентированном программировании, его ключевых принципах и понятиях, технологии программирования основных конструкций ООП.

В ходе освоения практической части школьники должны закрепить на практике знания ключевых понятий и принципов объектно-ориентированного программирования, приобрести навыки и умения разработки, тестирования и отладки программ в среде программирования Visual Studio.

Третья часть ЭУМК представляет собой модуль контроля и диагностики знаний в форме двухэтапного контроля. На первом этапе учащиеся выполняют интерактивные упражнения непосредственно после изучения соответствующей темы, что позволяет проверить уровень усвоения пройденного материала. Второй этап реализован в форме тестирования и проводится после изучения всех тем раздела «Объектно-ориентированное программирование».

Глава 3 Оценка эффективности разработанного ЭУМК

3.1 Общая характеристика исследования

Проведенный педагогический эксперимент был направлен на изучение реально складывающегося опыта организации учебного процесса и реализацию следующих целей:

1) исследование проблем теории и методики обучения объектно-ориентированному программированию в общеобразовательной школе;
2) построение методики обучения объектно-ориентированному программированию и внедрение её в практику обучения учащихся 10-11 классов.

Педагогический эксперимент проводился в два этапа:

1. На первом (констатирующем) исследовались: состояние проблемы обучения объектно-ориентированному программированию в средней школе; оптимальная последовательность представления учебных материалов; теоретические и методологические предпосылки разработки методики обучения объектно-ориентированному программированию в профильном курсе информатики и ИКТ средней школы.

Это потребовало анализа соответствующей педагогической, дидактической, методической, психологической литературы, учебных планов и программ школьного предмета «Информатика и ИКТ», а также существующих учебных и методических пособий (глава 1).

Результаты констатирующего этапа эксперимента позволили нам обосновать актуальность темы исследования в связи с выявившимся противоречием между необходимостью создания учебных пособий и методических рекомендаций, позволяющих организовать процесс обучения объектно-ориентированному программированию, и недостаточной разработанностью научно-практических рекомендаций в этой области.

На основе анализа актуальности и выявленных противоречий сформулирована проблема исследования, заключающаяся в необходимости

обоснования и дополнения методических рекомендаций по обучению объектно-ориентированному программированию учащихся профильных классов средних школ.

2. Целью второго (формирующего) этапа педагогического эксперимента являлась детальная разработка каждого компонента методики обучения объектно-ориентированному программированию учащихся средних школ: определение целей и задач обучения; обоснование принципов отбора содержания обучения с последующей его детализацией и преобразованием в учебный материал; выбор оптимальных методов, средств и форм организации учебного процесса. Одной из главных задач второго этапа исследования была разработка методических рекомендаций по проведению лабораторного практикума на ЭВМ и соответствующих электронных образовательных ресурсов. Список лабораторных работ приведен в таблице 4.

Результаты этого этапа педагогического эксперимента позволили сформулировать гипотезу нашего исследования, согласно которой использование предложенного программно-методического обеспечения обучения школьников объектно-ориентированному программированию, регламентируемое личностно-ориентированным и компетентностным подходами, а также дидактическими принципами наглядности, научности и индивидуальности, позволит обеспечить достижение предметных и метапредметных результатов обучения в соответствии с образовательным стандартом общего среднего образования.

3.2 Методика проведения и результаты педагогического эксперимента

Педагогический эксперимент проводимого исследования был организован на базе муниципального бюджетного общеобразовательного учреждения городского округа Тольятти «Классическая гимназия №39» в рамках прохождения педагогической и преддипломной практик. Изучение предмета «Информатика и ИКТ» в 11-х классах, реализуемого в рамках ФГОС СОО [1] и учебного плана школы, проходит на профильном уровне. Обучение

проводилось в компьютерном классе. Аудитория оборудована 15-ю персональными компьютерами, интерактивной электронной доской, видеопроектором, аудиоколонками, маркерной доской, организован доступ к сети Интернет.

К эксперименту были привлечены обучающиеся 11 класса.

Всего в эксперименте участвовал 21 учащийся. Во время проведения педагогического эксперимента были использованы такие эмпирические методы исследования как наблюдение, анкетирование, тестирование.

Организация и методика проведения констатирующего этапа педагогического эксперимента.

Констатирующий этап педагогического эксперимента проводился с целью анализа состояния сформированности предметных и межпредметных результатов обучения при освоении раздела «Объектно-ориентированное программирование» в профильном курсе информатики. На этом этапе решались задачи: формирование выборки обучающихся для участия в эксперименте, а также определение диагностического инструментария. Выборка составила 19 человек. Для проверки уровня сформированности предметных и межпредметных результатов обучения объектно-ориентированному программированию на констатирующем этапе был проведен входной тест из состава учебно-методического комплекса (УМК) «Информатика. Углубленный уровень: учебник для 11 класса» авторского коллектива И. Г. Семакина, Е. К. Хеннер, Л. В. Шестаковой [3].

Тест состоит из 15 вопросов (приложение И). Он направлен на проверку знаний основ объектно-ориентированного программирования: базовые принципы и понятия ООП (класс, объект, метод, инкапсуляция, наследование, полиморфизм и т. д.), основные конструкции программирования, спецификаторы доступа.

Количество правильных ответов учащихся свидетельствует об уровне сформированности предметных и межпредметных результатов обучения объектно-ориентированному программированию. Высокий уровень

предусматривает, что учащийся дает правильные ответы на не менее 90% тестовых заданий, базовый уровень – от 76 до 89% заданий, начальный – 61% до 75% и низкий – менее 60%.

Результаты тестирования учащихся экспериментальной группы на констатирующем этапе педагогического эксперимента представлены в таблице 6.

Таблица 6 - Уровень предметных и межпредметных результатов обучения по разделу «Объектно-ориентированное программирование» на констатирующем этапе.

Уровень		Экспериментальная группа		
	Оценка	Количество оценок	Общее количество баллов	Среднее значение
Низкий	2	5	10	
Начальный	3	9	27	
Базовый	4	4	16	
Высокий	5	1	5	
Всего		19	58	3,05

Данные проведенного начального среза показали, что среди обучаемых экспериментальной группы на низком уровне – 26% учащихся, на начальном уровне – 48%; на базовом уровне – 21%; на высоком уровне – 5% (рисунок 6).

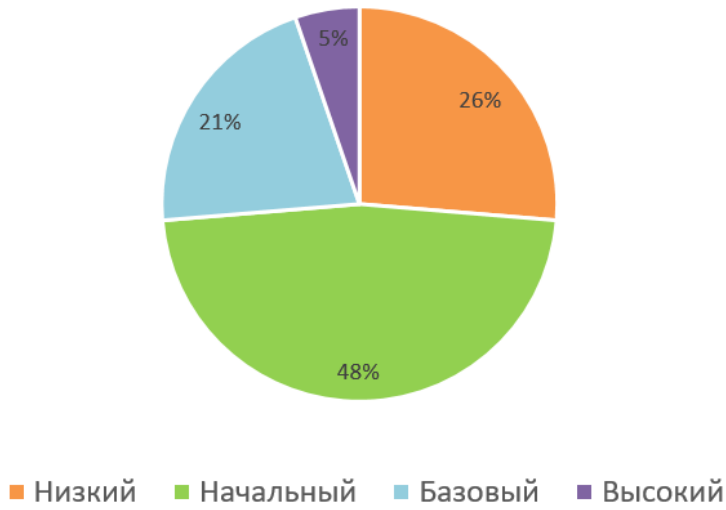

Рисунок 6 - Уровень предметных результатов обучения в экспериментальной группе на констатирующем этапе эксперимента

Анализ результатов, полученных в ходе констатирующего этапа эксперимента, позволил сделать вывод том, что большинство обучающихся, принимающих участие в эксперименте, имеют низкий и начальный уровни сформированности предметных результатов обучения объектно-ориентированному программированию. Данные результатов эксперимента позволяют предположить, что использование разработанного программно-методического обеспечения раздела «Объектно-ориентированное программирование» в профильном курсе информатики повысит качество усвоения учебного материала и уровень предметных и межпредметных результатов обучения по данному разделу, что может служить подтверждением гипотезы исследования.

ЗАКЛЮЧЕНИЕ

В ходе теоретического и экспериментального исследования, направленного на проектирование и разработку программно-методического обеспечения раздела «Объектно-ориентированное программирование» для учащихся профильных классов средней школы, были получены следующие основные результаты:

1. В теоретической части исследования обоснована актуальность; описаны теоретические и методологические предпосылки разработки методики обучения объектно-ориентированному программированию в профильном курсе информатики; проанализированы педагогическая, дидактическая, методическая и психологическая литература, нормативно-правовая документация, учебные планы и программы школьного предмета «Информатика и ИКТ»; выделены и определены основные понятия объектно-ориентированного программирования, особенности его реализации и требования ФГОС СОО к предметным и метапредметным результатам обучения раздела «Объектно-ориентированное программирование».

2. Практическая часть исследования представлена программно-методическим обеспечением раздела «Объектно-ориентированное программирование». Были описаны структура и содержание электронного учебно-методического комплекса, разработаны теоретико-познавательный, практический и контрольный модули, направленные на формирование представления об объектно-ориентированном программировании, его ключевых принципах и понятиях, технологии программирования основных конструкций ООП, а также на приобретение навыков и умений разработки, тестирования и отладки программ в среде программирования Visual Studio.

3. В ходе экспериментальной части исследования был проведен педагогический эксперимент в рамках прохождения педагогической и преддипломной практик на базе муниципального бюджетного общеобразовательного учреждения «Классическая гимназия №39». Результаты

педагогического эксперимента позволяют предположить, что организация образовательного процесса с использованием разработанного программно-методического обеспечения будет способствовать повышению уровня предметных и межпредметных результатов обучения объектно-ориентированному программированию учащихся средних школ в соответствии с образовательным стандартом общего среднего образования, а также будет положительно влиять на их уровень информационной и логико-алгоритмической культуры мышления.

Разработанное программно-методическое обеспечение раздела «Объектно-ориентированное программирование» может быть использовано учителями информатики, а отдельные электронные образовательные ресурсы могут быть использованы учащимися для самостоятельной работы и углубленного изучения программирования.

БИБЛИОГРАФИЧЕСКИЙ СПИСОК

1. ГОСТ N 24480. Федеральный Государственный образовательный стандарт среднего общего образования: утвержден приказом Министерства образования и науки Российской Федерации от 17 мая 2012г., №413 / Министерство образования и науки Российской Федерации. – Москва: 2012 г. – URL: http://www.consultant.ru/document/cons_doc_LAW_131131/ – (дата обращения: 28.04.2020). – Текст: электронный.

2. ГОСТ №2/16-з. Примерная образовательная программа среднего общего образования: одобрена решением федерального учебно-методического объединения по общему образованию, протокол от 28.06.2016г., №2 /16-3 / Министерство образования и науки Российской Федерации. – Москва: 2016г. – URL: http://www.consultant.ru/document/cons_doc_LAW_161101/ – (дата обращения: 28.04.2020). – Текст: электронный.

3. ГОСТ N 258. Федеральный базисный учебный план и примерные учебные планы для образовательных учреждений Российской Федерации, реализующих программы общего образования: утвержден Постановлением Правительства Российской Федерации от 09.03.2004 N 258. – Москва: 2012 г. – URL: http://www.consultant.ru/document/cons_doc_LAW_47213/ – (дата обращения: 28.04.2020). – Текст: электронный.

4. ГОСТ Р 52653-2006. Информационно-коммуникационные технологии в образовании. Термины и определения: утвержден приказом Федерального агентства по техническому регулированию и метрологии от 27 декабря 2006 г., N 419-ст. – Москва: 2006г. – URL: http://www.consultant.ru/cons/cgi/online.cgi?req=doc&base=OTN&n=9057#07652045227597943 – (дата обращения: 01.05.2020). – Текст: электронный.

5. Концепция долгосрочного социально-экономического развития российской федерации на период до 2020 года: утверждена распоряжением Правительства Российской Федерации от 17 ноября 2008 г., N 1662-р / Правительство Российской Федерации. – Москва: 2008г. – URL:

http://www.consultant.ru/cons/cgi/online.cgi?req=doc&base=LAW&n=308069&fld=134&dst=100001,0&rnd=0.6420990475794179#07393454031838735 – (дата обращения: 28.04.2020). – Текст: электронный.

6. Биллиг, В.А. Объектное программирование в классах на C# 3.0 / В.А. Биллиг. – 2-е изд. – Москва: Национальный Открытый Университет «ИНТУИТ», 2016. – 391 с. – URL: http://biblioclub.ru/index.php?page=book&id=428945 – (дата обращения: 04.05.2020). – Текст: электронный.

7. Биллиг, В.А. Основы программирования на C# 3.0: ядро языка / В.А. Биллиг. – 2-е изд. – Москва: Национальный Открытый Университет «ИНТУИТ», 2016. – 411 с. – URL: http://biblioclub.ru/index.php?page=book&id=428947 – (дата обращения: 04.05.2020). – Текст: электронный.

8. Гальперин, П. Я. Психология мышления и поэтапного формирования умственных действий / П. Я. Гальперин. Исследования мышления в советской психологии. – Москва: Просвещение, 1966. – 179 с.

9. Гафурова, Н.В. Методика обучения информационным технологиям. Теоретические основы: учебное пособие / Н.В. Гафурова, Е.Ю. Чурилова. – Красноярск: Сибирский федеральный университет, 2012. – 111 с. URL: http://biblioclub.ru/index.php?page=book&id=229302 – (дата обращения: 03.05.2020). – Текст: электронный.

10. Днепровская, Н.В. Открытые образовательные ресурсы / Н.В. Днепровская, Н.В. Комлева. – 2-е изд. – Москва: Национальный Открытый Университет «ИНТУИТ», 2016. – 140 с. – URL: http://biblioclub.ru/index.php?page=book&id=428994 – (дата обращения: 10.05.2020). – Текст: электронный.

11. Дудина, И. П. Объектно-ориентированное программирование: учебно-методическое пособие / И. П. Дудина. – Тольятти: Поволжский православный институт, 2016. – 221 с. – URL: http://elearn.pravinst.ru:180/pluginfile.php/7461/mod_resource/content/2/%D0%9B%D0%B0%D0%B1_%D0%BF%D1%80%D0%B0%D0%BA%D1%82%D0%B8%D0%BA%D1%83%D0%BC_%D0%A1%23.pdf – (дата обращения: 02.05.2020). – Текст: электронный.

12. Ершов, А.П. Школьная информатика: концепции, состояния, перспективы / А.П. Ершов, Г.А. Звенигородский, Ю.А. Первин. – Новосибирск: Препринт №152, 1979. – 51 с. – URL: http://ershov.iis.nsk.su/ru/node/805749 – (дата обращения: 13.05.2020). – Текст: электронный.
13. Жданко, Т. А. Системно-деятельностный подход: сущностная характеристика и принципы реализации / Т. А. Жданко. – Magister Dixit, 2012. – С. 183-189. – URL: https://cyberleninka.ru/article/n/sistemno-deyatelnostnyy-podhod-suschnostnaya-harakteristika-i-printsipy-realizatsii/viewer – (дата обращения: 15.05.2020). – Текст: электронный.
14. Зыков, С.В. Введение в теорию программирования. Объектно-ориентированный подход / С.В. Зыков. – 2-е изд. – Москва: Национальный Открытый Университет «ИНТУИТ», 2016. – 189 с. – URL: http://biblioclub.ru/index.php?page=book&id=429073 – (дата обращения: 22.05.2020). – Текст: электронный.
15. Копаев, А.В. О практическом значении алгоритмического стиля мышления / А.В. Копаев. – Информационные технологии в общеобразовательной школе, 2003. – № 4. – С. 6-11
16. Котов, О.М. Язык C#: краткое описание и введение в технологии программирования: учебное пособие / О.М. Котов; Министерство образования и науки Российской Федерации, Уральский федеральный университет имени первого Президента России Б. Н. Ельцина. – Екатеринбург: Издательство Уральского университета, 2014. – 209 с. - URL: http://biblioclub.ru/index.php?page=book&id=275809 – (дата обращения: 17.05.2020). – Текст: электронный.
17. Красильникова, В.А. Использование информационных и коммуникационных технологий в образовании: учебное пособие / В.А. Красильникова. – Москва: Директ-Медиа, 2013. – 292 с. – URL: http://biblioclub.ru/index.php?page=book&id=209293 – (дата обращения: 11.05.2020). – Текст: электронный.

18. Кузнецов, А. А. Общая методика обучения информатике: учебное пособие / А.С. Кузнецов, Т.Б. Захарова, А.С. Захаров. – Москва: Прометей, 2016. – 300 с. – URL: http://biblioclub.ru/index.php?page=book&id=438600 – (дата обращения: 09.05.2020). – Текст: электронный.
19. Кузнецов, В.В. Основы объектно-ориентированного программирования в Delphi: учебное пособие / В.В. Кузнецов, И.В. Абдрашитова; под ред. Т.Б. Корнеевой. – Томск, 2011. – 122 с. – URL: https://omu.ru/files/docs/132663_451521.pdf – (дата обращения: 06.05.2020). – Текст: электронный.
20. Лапчик, М. П. Методика обучения информатике: учебное пособие / М. П. Лапчик, М. И. Рагулина, И. Г. Семакин, Е. К. Хеннер; под ред. М. П. Лапчика. — Санкт-Петербург: Издательство «Лань», 2016. — 392 с. – URL: https://b-ok.cc/book/2910680/fc072e – (дата обращения: 11.05.2020). – Текст: электронный.
21. Лобачев, С. Л. Основы разработки электронных образовательных ресурсов: учебный курс / С. Лобачев. – 2-е изд. – Москва: Национальный Открытый Университет «ИНТУИТ», 2016. – 189 с. – URL: http://biblioclub.ru/index.php?page=book&id=429160 – (дата обращения: 19.05.2020). – Текст: электронный.
22. Макарова, Н. В. Практикум по программированию. 10-11 класс. Базовый уровень: учебник / Н. В. Макарова. – Санкт-Петербург: Питер, 2008. – 176 с.
23. Макарова, Н. В. Методическая поддержка деятельности учителя информатики в условиях многоцелевой образовательной среды / Н. В. Макарова, Ю. Ф. Титова. – Известия РГПУ им. А.И. Герцена, 2019. – № 194. – С. 143 – 155. – URL: https://drive.google.com/file/d/0B6696ckkWj_zZEtKMjJwYkN6aFk/view – (дата обращения: 12.05.2020). – Текст: электронный.
24. Марченко, А. Л. Основы программирования на C# 2.0: методические рекомендации / А. Л. Марченко; Национальный Открытый Университет «ИНТУИТ». – Москва: Интернет-Университет Информационных Технологий,

2007. – 553 с. – URL: http://biblioclub.ru/index.php?page=book&id=233313 – (дата обращения: 06.05.2020). – Текст: электронный.

25. Машбиц, Е. И. Психолого-педагогические проблемы компьютеризации обучения / Е. И. Машбиц. — Москва: Педагогика, 1988. — 192 с. – URL: http://pedlib.ru/Books/6/0442/index.shtml – (дата обращения: 08.05.2020). – Текст: электронный.

26. Мещерякова, И.Н. Возможности электронного обучения в развитии познавательной активности студентов: учебно-методическое пособие / И.Н. Мещерякова. – Москва: Флинта, 2014. – 63 с. – URL: http://biblioclub.ru/index.php?page=book&id=279813 – (дата обращения: 23.05.2020). – Текст: электронный.

27. Минькович, Т.В. Модель методических систем обучения информатике / Т.В. Минькович. – Москва: Логос, 2011. – 307 с. – URL: http://biblioclub.ru/index.php?page=book&id=119451 – (дата обращения: 20.05.2020). – Текст: электронный.

28. Околелов, О.П. Справочник по инновационным теориям и методам обучения, воспитания и развития личности: настольная книга педагога: справочник / О.П. Околелов. – Москва; Берлин: Директ-Медиа, 2015. – 272 с. – URL: http://biblioclub.ru/index.php?page=book&id=278853 – (дата обращения: 14.05.2020). – Текст: электронный.

29. Позанова, И. А. Основы построения электронных учебно-методических комплексов / И. А. Позанова. – Вестник Восточно-Сибирского института МВД России. — 2011. – 6 с. — URL: https://cyberleninka.ru/article/n/osnovy-postroeniya-elektronnyh-uchebno-metodicheskih-kompleksov/viewer – (дата обращения: 24.05.2020). – Текст: электронный.

30. Полежаева, О. А. Информатика. УМК для старшей школы: 10-11 классы. Углубленный уровень. Методическое пособие для учителя / О. А. Полежаева, М. С. Цветкова. – Москва: БИНОМ, 2013. – 114 с. – URL: http://files.lbz.ru/pdf/mpSemakin10-11uufgos.pdf – (дата обращения: 19.05.2020). – Текст: электронный.

31. Поляков, К. Ю. Информатика. 10-11 классы. Базовый и углубленный уровни: методическое пособие / К. Ю. Поляков, Е. А. Еремин. – Москва: БИНОМ, 2016. – 128 с. – URL: http://www.lbz.ru/metodist/iumk/informatics/files/polyakov-10-11-bu-uu-met.pdf – (дата обращения: 11.05.2020). – Текст: электронный.
32. Поляков, К. Ю. Информатика. Углубленный: учебник для 11 класса: в 2 ч. Ч. 2 / К. Ю. Поляков, Е. А. Еремин. – Москва: БИНОМ, 2013. – 304 с. – URL: https://vk.com/doc67715714_489824519?hash=df8c12d29c94622867&dl=c45e6cf01ffa842e03 – (дата обращения: 11.05.2020). – Текст: электронный.
33. Потапенко, С. М. Задачи регионального содержания как фактор активизации познавательной деятельности на уроках информатики: монография / С. М. Потапенко; Министерство образования и науки Российской Федерации, Северный (Арктический) федеральный университет имени М. В. Ломоносова. – Архангельск: САФУ, 2013. – 103 с. – URL: http://biblioclub.ru/index.php?page=book&id=436191 – (дата обращения: 22.05.2020). – Текст: электронный.
34. Рогожин, М. Ю. Подготовка и защита письменных работ: учебно-практическое пособие / М. Ю. Рогожин. – Москва; Берлин: Директ-Медиа, 2014. – 238 с. – URL: http://biblioclub.ru/index.php?page=book&id=253712 – (дата обращения: 28.05.2020). – Текст: электронный.
35. Рубинштейн, С. Л. О мышлении и путях его исследования: монография / С. Л. Рубинштейн. – Москва: Издательство Академии Наук СССР, 1958 – 151 с. – URL: http://biblioclub.ru/index.php?page=book&id=476734 – (дата обращения: 05.05.2020). – Текст: электронный.
36. Рыжов, В. Н. Методика преподавания информатики: учебное пособие / В. Н. Рыжов. – Саратов: Саратовский государственный университет имени Н. Г. Чернышевского, 2011. – 512 с. – URL: http://elearn.pravinst.ru:180/pluginfile.php/7039/mod_resource/content/4/Рыжов.pdf – (дата обращения: 08.05.2020). – Текст: электронный.

37. Семакин, И. Г. Информатика. Углубленный уровень: учебник для 11 класса: в 2 ч. Ч. 1 / И. Г. Семакин, Е. К. Хеннер, Л. В. Шестакова. – Москва: БИНОМ, 2014. – 176 с. – URL: https://vk.com/doc67715714_489017864?hash=8a70655fb95c2cc46a&dl=fa5ee96d06b4fe539a – (дата обращения: 11.05.2020). – Текст: электронный.
38. Суханов, М.В. Основы Microsoft .NET Framework и языка программирования C#: учебное пособие / М.В. Суханов, И.В. Бачурин, И.С. Майоров; Министерство образования и науки Российской Федерации, Федеральное государственное автономное образовательное учреждение высшего профессионального образования Северный (Арктический) федеральный университет им. М.В. Ломоносова. – Архангельск: ИД САФУ, 2014. – 97 с. – URL: http://biblioclub.ru/index.php?page=book&id=312313 – (дата обращения: 17.05.2020). – Текст: электронный.
39. Трайнев, В.А. Электронно-образовательные ресурсы в развитии информационного общества: обобщение и практика: монография / В.А. Трайнев. – Москва: Дашков и Ко, 2015. – 256 с. – URL: http://biblioclub.ru/index.php?page=book&id=253962 – (дата обращения: 22.05.2020). – Текст: электронный.
40. Угринович, Н. Д. Информатика. 10 класс. Базовый уровень: учебник / Н. Д. Угринович. — Москва: БИНОМ, 2017. — 288 с. – URL: https://znayka.pw/uchebniki/10-klass/informatika-10-klass-bazovyj-uroven-ugrinovich-n-d-uchebnik/ – (дата обращения: 13.05.2020). – Текст: электронный.
41. Шубович, М. М. Компетентностный подход как идеология современного личностно-ориентированного образования / М. М. Шубович. – Вестник Казанского технологического университета, 2009. − С. 397-403. – URL: https://cyberleninka.ru/article/n/kompetentnostnyy-podhod-kak-ideologiya-sovremennogo-lichnostno-orientirovannogo-obrazovaniya-1 – (дата обращения: 07.05.2020). – Текст: электронный.
42. Якиманская, И. С. Личностно-ориентированное обучение в современной школе / И. С. Якиманская. – Москва: Сентябрь, 1996. – 96 с. – URL:

https://hum.uch-lit.ru/szbrannoe/yakimanskaya-i-s-lichnostno-orientirovannoe-obuchenie-v-sovremennoy-shkole-onlayn – (дата обращения: 09.05.2020). – Текст: электронный.

ЭЛЕКТРОННЫЕ РЕСУРСЫ

1. LearningApps.org. Создание мультимедийных интерактивных упражнений: сайт. – URL: https://learningapps.org/ – (дата обращения: 04.05.2020). – Текст: электронный

2. Moodle. Объектно-ориентированное программирование: сайт. – URL: https://visual-c-sharp.moodlecloud.com/login/index.php – (дата обращения: 13.05.2020). – Текст: электронный

3. БИНОМ. УМК Семакин И. Г.: сайт. – URL: https://lbz.ru/metodist/authors/informatika/2/ – (дата обращения: 21.05.2020). – Текст: электронный

ПРИЛОЖЕНИЯ

ПРИЛОЖЕНИЕ А

Технологическая карта урока

Тема: «Понятия класса и объекта»

Предмет: информатика в 11 классе согласно ФГОС

Автор карты: Артамонова Екатерина Евгеньевна

Тольятти

2020 год

Технологическая карта урока

Предмет:	Информатика	11 класс	Профильный уровень
Тема урока:	Понятия класса и объекта		
Автор карты:	Артамонова Екатерина Евгеньевна		
Тип урока:	Урок открытия нового знания		
Цель урока:	Создать условия для формирования у учащихся знаний понятий класса и объекта и их элементов.		
Задачи урока:	Обучающие	Развивающие	Воспитательные
	• знакомство учащихся с понятиями класса и объекта.	• развитие логического и алгоритмического мышления школьников, приемов умственной деятельности, формирование и развитие функционального мышления учащихся, развитие познавательных потребностей учащихся.	• побуждение интереса к изучению информатики; • формирование творческого воображения и умения решать нестандартные задачи.
Основные понятия темы:	• Класс; • Объект; • Спецификатор; • Метод; • Данные; • Поле класса.		
Материалы и оборудование:	Компьютер, мультимедийный проектор.		
Планируемые образовательные результаты:	Личностные	Метапредметные	Предметные
	• сформированность мировоззрения, соответствующего современному уровню развития науки и общественной практики.	• владение основами самоконтроля, самооценки, принятия решений и осуществления осознанного выбора в учебной и познавательной деятельности; • владение основными универсальными умениями информационного характера: постановка и формулирование цели; структурирование и визуализация информации.	• знание базовых понятий ООП; • знание основных спецификаторов класса; • умение описывать элементы класса; • умение применять спецификаторы доступности.
Методическое назначение средств ИКТ на уроке:	Демонстрационные средства ИКТ визуализируют изучаемые объекты, явления, процессы с целью их исследования и изучения. Учебно-игровые средства ИКТ предназначены для создания учебных ситуаций, в которых деятельность обучаемых реализуется в игровой форме.		

Структура и ход урока

№	Этап урока	Задачи этапа	Деятельность учителя	Деятельность ученика	УУД
1	Организационный момент	Проверка готовности к уроку.	Приветствует учащихся, проверяет их готовность к уроку, отмечает отсутствующих.	Приветствуют учителя, проверяют свою готовность к уроку.	**Личностные**: самоопределение, самоконтроль; **Коммуникативные:** планирование учебного сотрудничества с учителем и сверстниками.
2	Этап подготовки учащихся к активному и сознательному усвоению новых знаний	Формирование положительной мотивации. Подготовка мышления учащихся и организация осознания ими внутренней потребности к построению нового способа действий.	Предлагает учащимся ответить на некоторые вопросы: 1. Что такое алгоритм? 2. Назовите свойства алгоритма. 3. Назовите виды алгоритмов.	Ученики отвечают на вопросы: 1. Алгоритм – это предназначенное для конкретного исполнителя точное описание последовательности действий, направленных на решение поставленной задачи. 2. Дискретность, результативность, массовость, детерминированность, выполнимость и понятность. 3. Линейный, ветвящийся, циклический.	**Регулятивные:** целеполагание, оценка; **Познавательные:** осознанное построение речевого высказывания в устной форме, установление причинно-следственных связей. **Коммуникативные:** умение слушать и вступать в диалог, формулирование и аргументация своего мнения, планирование учебного сотрудничества с учителем и сверстниками.
3	Актуализация опорных знаний	Формулирование задач деятельности и выбор модели и средств их реализации.	Мы начинаем изучать объектно-ориентированное программирование. Давайте запишем тему и сформулируем цель нашего урока. Тема: «Понятия класса и объекта»	Цель урока: узнать понятия класса и объекта и их элементов.	**Регулятивные:** планирование, целеполагание, прогнозирование; **Познавательные:** осознанное построение

					речевого высказывания в устной форме. **Коммуникативные:** умение с достаточно полнотой и точностью выражать свои мысли в соответствии с задачами и условиями коммуникации.
4	Изучение нового материала	Формирование учащимися новой модели действий, умения ее применять при решении задачи, вызвавшей затруднение и аналогичных ей вопросах.	Изучение нового материала проходит с помощью гипертекстовых мультимедиа учебных материалов, расположенных в открытой системе управления обучением Moodle (https://visual-c-sharp.moodlecloud.com/login/index.php). C# является полноценным объектно-ориентированным языком. Это значит, что программу на C# можно представить в виде взаимосвязанных взаимодействующих между собой объектов. Описанием объекта является класс, а объект представляет экземпляр этого класса. Класс – это множество объектов, связанных общностью свойств, поведения, связей и семантики. Любой объект является экземпляром класса. Класс является типом данных, определяемым пользователем. Можно еще провести следующую аналогию. У нас у всех есть некоторое представление о человеке, у которого есть имя, возраст, какие-то другие характеристики. То есть некоторый шаблон - этот шаблон можно назвать классом. Конкретное воплощение этого шаблона может отличаться, например, одни люди имеют одно имя, другие - другое имя. И реально существующий человек (фактически экземпляр данного класса) будет представлять объект этого класса. Класс является типом данных, определяемым пользователем. Он должен представлять собой одну логическую сущность, например, являться моделью реального объекта или процесса. Элементами класса являются данные и функции (методы), предназначенные для их обработки. Описание класса содержит ключевое слово class, за которым следует его имя, а далее в фигурных скобках - тело класса, то есть список его элементов. class Person { }	Слушают преподавателя, читают с экрана, записывают главные мысли в тетрадь. Класс – это множество объектов, связанных общностью свойств, поведения, связей и семантики. Любой объект является экземпляром класса. Класс является типом данных, определяемым пользователем. Описание класса: class Person { } Спецификаторы классов: 1) new: Используется для вложенных классов. Задает новое описание класса взамен унаследованного от предка. Применяется в иерархиях объектов; 2) public: Доступ не ограничен;	**Логические:** классификация объектов по выделенным признакам; **Познавательные:** структурирование знаний, выделение необходимой информации, осознанное построение речевого высказывания в письменной форме; **Коммуникативные:** планирование учебного сотрудничества с учителем и сверстниками, умение с достаточно полнотой и точностью выражать свои мысли в соответствии с задачами и условиями коммуникации, умение слушать и вступать в диалог, формулирование и аргументация своего мнения.

			Спецификаторы Спецификаторы определяют свойства класса, а также доступность класса для других элементов программы. Класс можно описывать непосредственно внутри пространства имен или внутри другого класса. В последнем случае класс называется вложенным. В зависимости от места описания класса некоторые из этих спецификаторов могут быть запрещены. В C# применяются следующие спецификаторы классов: 1) new: Используется для вложенных классов. Задает новое описание класса взамен унаследованного от предка. Применяется в иерархиях объектов; 2) public: Доступ не ограничен; 3) protected: Используется для вложенных классов. Доступ только из элементов данного и производных классов; 4) internal: Доступ только из данной программы (сборки); 5) protected internal: Доступ только из данного и производных классов или из данной программы; 6) private: Используется для вложенных классов, доступ только из элементов класса, внутри которого описан данный класс; 7) static: Статический класс или член класса; 8) abstract: Абстрактный класс. Применяется в иерархии объектов. Спецификаторы 2-6 называются спецификаторами доступа. Они определяют, откуда можно непосредственно обращаться к данному классу. Для классов, которые описываются в пространстве имен непосредственно (то есть не вложенные классы) допускаются только два спецификатора: public и internal. По умолчанию, то есть если ни один спецификатор доступа не указан, подразумевается спецификатор internal. Класс является обобщенным понятием, определяющим характеристики и поведение некоторого множества конкретных объектов этого класса, называемых экземплярами или объектами класса. Объект – это структура данных, содержащая поля данных различных типов и заголовки методов и обобщающая структуру «Запись». Объекты создаются явным или неявным образом, то есть либо программистом, либо системой.	3) protected: Используется для вложенных классов. Доступ только из элементов данного и производных классов; 4) internal: Доступ только из данной программы (сборки); 5) protected internal: Доступ только из данного и производных классов или из данной программы; 6) private: Используется для вложенных классов, доступ только из элементов класса, внутри которого описан данный класс; 7) static: Статический класс или член класса; 8) abstract: Абстрактный класс. Применяется в иерархии объектов. Объект – это структура данных, содержащая поля данных различных типов и заголовки методов и обобщающая структуру «Запись». Создание экземпляра класса: Person tom = new Person();	

			Программист создает экземпляр класса с помощью операции new, например: Person tom = new Person(); Для каждого объекта при его создании в памяти выделяется отдельная область, в которой хранятся его данные. Кроме того, в классе могут присутствовать статические элементы, которые существуют в единственном экземпляре для всех объектов класса. Часто статические данные называют данными класса, а остальные – данными экземпляра. Функциональные элементы класса не тиражируются, то есть всегда хранятся в единственном экземпляре. Для работы с данными класса используются методы класса (статические методы), для работы с данными экземпляра — методы экземпляра, или просто методы. До сих пор мы использовали в программах только один вид функциональных элементов класса — методы. Поля и методы являются основными элементами класса. Метод – это функциональный элемент класса, который реализует вычисления или другие действия, выполняемые классом или экземпляром. Методы определяют поведение класса. Кроме того, в классе можно задавать целую гамму других элементов: свойства, события, индексаторы, операции, конструкторы, деструкторы, а также типы). Ниже приведено краткое описание всех элементов класса: • константы класса хранят неизменяемые значения, связанные с классом; • поля содержат данные класса; • методы реализуют вычисления или другие действия, выполняемые классом или экземпляром; • свойства определяют характеристики класса в совокупности со способами их задания и получения, то есть методами записи и чтения; • конструкторы реализуют действия по инициализации экземпляров или класса в целом; • деструкторы определяют действия, которые необходимо выполнить до того, как объект будет уничтожен; • индексаторы обеспечивают возможность доступа к элементам класса по их порядковому номеру;	Метод – это функциональный элемент класса, который реализует вычисления или другие действия, выполняемые классом или экземпляром. Методы определяют поведение класса. Элементы класса: • константы класса хранят неизменяемые значения, связанные с классом; • поля содержат данные класса; • методы реализуют вычисления или другие действия, выполняемые классом или экземпляром; • свойства определяют характеристики класса в совокупности со способами их задания и получения, то есть методами записи и чтения; • конструкторы реализуют действия по инициализации экземпляров или класса в целом; • деструкторы определяют действия, которые необходимо выполнить до того, как	

			• операции задают действия с объектами с помощью знаков операций; • события определяют уведомления, которые может генерировать класс; •типы — это типы данных, внутренние по отношению к классу. Присваивание и сравнение объектов Механизм выполнения присваивания один и тот же для величин любого типа, как ссылочного, так и значимого, однако результаты различаются. При присваивании значения копируется значение, а при присваивании ссылки — ссылка, поэтому после присваивания одного объекта другому мы получим две ссылки, указывающие на одну и ту же область памяти. Из этого следует, что если изменить значение одной величины ссылочного типа, это может отразиться на другой. Данные. Поля и константы Данные, содержащиеся в классе, могут быть переменными или константами. Переменные, описанные в классе, называют полями класса. При описании элементов класса используются спецификаторы: 1) new: новое описание поля, скрывающее унаследованный элемент класса; 2) public: доступ к элементу не ограничен; 3) protected: доступ только из данного и производных классов; 4) internal: доступ только из данной программы (сборки); 5) protected internal: доступ только из данного и производных классов или из данной программы; 6) private: доступ только из данного класса; 7) static: одно поле для всех экземпляров класса; 8) readonly: поле доступно только для чтения; 9) volatile: поле может изменяться другим процессом. По умолчанию элементы класса считаются закрытыми (private). Для полей класса этот вид доступа является предпочтительным, поскольку ноля определяют внутреннее строение класса, которое должно быть скрыто от пользователя. Все методы класса имеют непосредственный доступ к его закрытым полям. Поля, описанные со спецификатором static, а также константы существуют в единственном экземпляре для всех объектов класса, поэтому к ним обращаются не через имя экземпляра, а через имя	объект будет уничтожен; • индексаторы обеспечивают возможность доступа к элементам класса по их порядковому номеру; • операции задают действия с объектами с помощью знаков операций; • события определяют уведомления, которые может генерировать класс; •типы — это типы данных, внутренние по отношению к классу.	

			класса. Если класс содержит только статические элементы, экземпляр класса создавать не требуется. Именно этим фактом мы пользовались во всех предыдущих листингах. Обращение к полю класса выполняется с помощью операции доступа (точка). Справа от точки задается имя поля, слева — имя экземпляра для обычных полей или имя класса для статических.		
5	Применение полученных знаний	Усвоение учащимися нового метода действий. Проверка степени усвоения полученного знания.	Выполнение интерактивных упражнений: «Описание элементов класса C#» (https://learningapps.org/watch?v=p9c6yh2rt20): 1. Перед выполнением задания повторите принцип объявления класса и спецификаторы доступа. 2. Впишите в пропуски пропущенные части кода. • «Спецификаторы класса C#» (https://learningapps.org/watch?v=pyeejeptc20): 1. Перед выполнением задания повторите спецификаторы класса и их описание. 2. Соотнесите определение спецификатора класса с его названием. Если соотнесено верно – карточка пропадет с поля.	Выполняют задания на компьютере, обсуждают, высказывают свое мнение.	**Личностные:** самоорганизация, самоопределение; **Познавательные:** структурирование знаний, классификация объектов по выделенным признакам; **Регулятивные:** планирование, коррекция; **Коммуникативные:** умение вступать в диалог, формулирование и аргументация своего мнения.
6	Подведение итогов урока, рефлексия	Осознание учащимися способа преодоления затруднения и самостоятельная оценка полученных результатов выполненной работы.	Подводит обучающихся к выводу о достижении цели урока. Итак, что нового узнали сегодня на уроке? Молодцы, ребята, сегодня вы продуктивно поработали, давайте оценим вашу работу. Я считаю, что на оценку «5» поработали: ФИО обучаемых; на оценку «4» поработали: ФИО обучаемых. Согласны ли вы с моим мнением?	Анализируют свою работу на уроке.	**Познавательные:** осознанное построение речевого высказывания в устной форме; **Регулятивные:** оценка, саморегуляция; **Коммуникативные:** умение с достаточно полнотой и точностью выражать свои мысли в соответствии с задачами и условиями коммуникации.

7	Информация о домашнем задании	Проверка степени усвоения полученного знания.	Озвучивает домашнее задание: Лабораторная работа 1. Понятия класса и объекта.	Записывают задания в дневники.	**Личностные:** самоопределение; **Коммуникативные:** планирование учебного сотрудничества с учителем.

ПРИЛОЖЕНИЕ Б

Лабораторная работа №5. Разработка графического интерфейса пользователя

Задание: спроектировать графический интерфейс пользователя для одного из следующих заданий:

Вариант 1

1. Разработать класс «Абитуриент», содержащий такие поля, как «название факультета», «фамилия, имя абитуриента», «домашний адрес», «результаты ЕГЭ абитуриента», «участие/победа в олимпиаде», «проходной балл».
2. Разработать метод расчета общего балла абитуриента, метод сравнения общего балла с проходным баллом и перегруженный метод вывода сведений об абитуриенте на экран с указанием о зачислении.
3. Написать программу, демонстрирующую все разработанные элементы класса.
4. Перегрузить расчетные методы таким образом, чтобы пользователь мог взаимодействовать с графическим интерфейсом (программа должна воспринимать исходные данные из текстовых полей, а не с консоли).
5. Написать необходимые обработчики событий для всех управляющих элементов графического интерфейса.

Вариант 2

1. Разработать класс «Рекорд по плаванию», содержащий такие поля, как «ФИО пловца», «фамилия и имя тренера», «техника плавания», «дистанция», «время прохождения дистанции», «лучшее время на данной дистанции».
2. Разработать метод сравнения времени пловца с лучшим временем и перегруженный метод вывода сведений о пловце на экран с указанием об установлении рекорда.
3. Написать программу, демонстрирующую все разработанные элементы класса.
4. Перегрузить расчетные методы таким образом, чтобы пользователь мог

взаимодействовать с графическим интерфейсом (программа должна воспринимать исходные данные из текстовых полей, а не с консоли).

5. Написать необходимые обработчики событий для всех управляющих элементов графического интерфейса.

ПРИЛОЖЕНИЕ В

Контрольное тестирование по разделу «Объектно-ориентированное программирование»

1. Множество объектов, связанных общностью свойств, поведения, связей и семантики; тип данных, определяемый пользователем - это:

1) объект;
2) класс;
3) атрибут;
4) метод.

2. Объект – это:

1) множество объектов, связанных общностью свойств, поведения, связей и семантики; тип данных, определяемый пользователем;
2) функциональный элемент класса, который реализует вычисления или другие действия, выполняемые классом или экземпляром;
3) структура данных, содержащая поля данных различных типов и заголовки методов и обобщающая структуру «Запись»;
4) процедура, которая начинает выполняться после реализации определенного события.

3. Объединение в классе данных и подпрограмм для их обработки, а также скрытие внутреннего устройства объектов – это _____

4. Свойство системы, позволяющее описать новый класс на основе уже существующего с частично или полностью заимствующейся функциональностью – это _____

5. Способность использования одинаковых имен для методов, входящих в различные классы – это _____

6. Процесс выделения наиболее существенных характеристик некоторого объекта, отличающих его от всех других видов объектов, важных с точки зрения дальнейшего рассмотрения и анализа, и игнорирование менее важных или незначительных деталей – это _____

7. Метод – это:

1) атрибут объекта, определяющий то, как объект выглядит или как он может себя вести;
2) функциональный элемент класса, который реализует вычисления или другие действия, выполняемые классом или экземпляром;
3) действие, распознаваемое элементом управления;
4) процедура, которая начинает выполняться после реализации определенного события.

8. Специальный метод класса, инициализирующий объект, который не возвращает никакого значения, имеет то же имя, что и имя класса, объявляется в открытой части класса, называется ...

1) конструктором;
2) деструктором;
3) спецификатором доступа;
4) свойством.

9. Из всех приведенных ниже утверждений, касающихся объектов и классов, отметьте правильные:

1) значения полей описывают состояние объекта;
2) методы класса описывают поведение объектов;
3) поля класса могут быть разного типа;
4) поля класса должны быть разного типа;
5) в любом классе должно быть как минимум одно поле данных.

10. Спецификатор доступа private используется для того, чтобы сделать поля и методы доступными ...

1) только внутри класса;
2) только из потомков класса;
3) внутри класса, из потомков и извне;
4) только извне;
5) из потомков и извне.

11. Статические (static) поля данных класса...

1) являются общими для всех экземпляров класса (всех объектов);
2) являются неизменяемыми (константными) полями;
3) располагаются в закрытой (private) части класса;
4) могут инициализироваться только в конструкторе;
5) содержат ссылки на другие объекты.

12. Проведите соответствия между параметрами и ключевыми словами их описания:

1) параметры-значения;	a) params;
2) параметры-ссылки;	b) out;
3) выходные параметры;	c) без ключевого слова;
4) параметры-массивы;	d) ref.

13. От какого количества базовых классов может наследоваться класс-потомок?

1) от одного;
2) от двух;
3) от трех;
4) все вышеперечисленное.

14. Проведите соответствие между элементами графического интерфейса пользователя и их описанием:

1) Form;

2) Label;

3) Button;

4) TextBox;

5) PictureBox.

a) позволяет пользователю вводить текст;

b) базовый графический объект, на основе которого создается программный проект;

c) при щелчке возникает событие;

d) отображает изображение;

e) предоставляет элементу управления информацию.

15. Что будет выведено на консоль при выполнении программы?

```
1.public class Person
2.{
3.    public readonly string name = "Artur";
4.    public Person(string name)
5.    {
6.        this.name = name;
7.    }
8.}
9.class Program
10. {
11.     static void Main(string[] args)
12.     {
13.         Person person = new Person("John");
14.         Console.WriteLine(person.name);
15.         Console.ReadKey();
16.     }
17. }
```

1) возникнет ошибка;

2) "John";

3) "Arthur";

4) name.

16. Что будет выведено на консоль при выполнении программы?

```
1.int firstNumber = 5;
2.double secondNumber = 5.0;
3.Console.WriteLine(firstNumber == secondNumber);
```

1) возникнет ошибка;

2) false;

3) true.

17. Что будет выведено на консоль при выполнении программы?

```
1.class Person
2.{
3.    public string Name { get; set; }
4.    public Person(string name)
5.    {
6.        this.Name = name;
7.    }
8.}
9.class Program
10. {
11.     static void Main(string[] args)
12.     {
13.         const string personName = "Anatoliy";
14.         Person firstPerson = new Person(personName);
15.         Person secondPerson = new Person(personName);
16.         Console.WriteLine(firstPerson == secondPerson);
17.     }
18. }
```

1) возникнет ошибка;

2) false;

3) true.

18. Что будет выведено на консоль при выполнении программы?

```
1.byte firstValue = 200;
2.byte secondValue = 100;
3.Console.WriteLine(firstValue + secondValue);
```

1) возникнет ошибка;

2) -200;

3) 100;

4) -300;

5) 300;

6) -100.

19. Что будет содержать массив models после выполнения данного фрагмента кода?

```
1.object[] models = { 1, "300", 100, "5" };
2.Array.Sort(models);
```

1) возникнет ошибка;

2) 1, 100, "300", "5";

3) 1, "300", 100, "5";

4) 1, "5", 100, "300".

20. Возникнет ли ошибка при выполнении данного фрагмента кода?

```
1.object model = 156;
2.model = "157";
3.Console.WriteLine(model);
```

1) да;

2) нет.

Ответы:

1. 2)
2. 3)
3. инкапсуляция
4. наследование
5. полиморфизм
6. абстрагирование
7. 2)
8. 1)
9. 1), 2), 3)

10.1)

11.1)

12.1) - c, 2) - d, 3) – b, 4) – a

13.1)

14.1) - b, 2) - e, 3) - c, 4) - a, 5) - d

15.2)

16.3)

17.2)

18.5)

19.1)

20.2)

ПРИЛОЖЕНИЕ Г

Сценарий видеоурока «Разработка приложения «Удобный календарь»

Под графическим интерфейсом пользователя (GUI) понимается программа, выводящая информацию с помощью форм и панелей, называемых окнами.

Приведенный на экране рисунок иллюстрирует вид предлагаемого GUI (рисунок Г.1). Все прямоугольное пространство представляет собой форму и создается от класса C# с именем Form, расположенного в пространстве имен System.Windows.Forms. В языке C# форма выполняет функции контейнера для других элементов GUI. На форме мы расположим четыре типа элементов: ярлыки(Labels), кнопки (Buttons), окна редактирования (TextBox) и поля рисунков (PictureBox).

Рисунок Г.1 – Вид графического интерфейса

Прямоугольные области для ввода текста представляют собой окна редактирования и соответствующие им объекты будут созданы из класса TextBox. Кнопки представляют объекты, созданные из класса Button. Все идентифицирующие надписи являются объектами класса Labels и их принято называть ярлыками или метками.

Необходимо создать новый проект – Приложение Windows Forms, именуем его Calendar, располагаем в нужной директории и нажимаем ОК, теперь перед вами появилось окно формы. Сначала создадим главное окно приложения (окно входа в календарь) (рисунок Г.2):

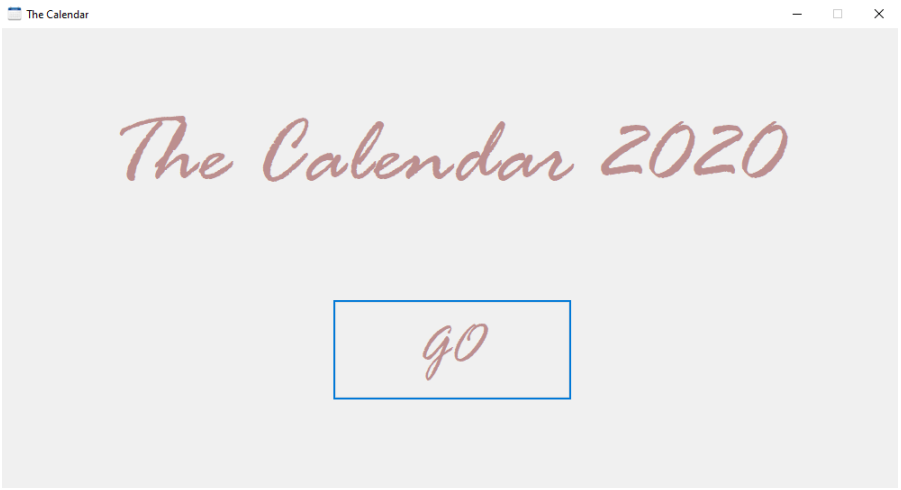

Рисунок Г.2 - Окно входа

Слева из Панели элементов нужно вытянуть на форму 1 Button, 1 Label и расположить, как показано на рисунке сверху.

Нажимаем на окно формы, справа в Свойствах находим свойство Size, вводим значение 1015; 550. Далее в свойстве Text записываем «The Calendar». Свойство MaximizeBox переводим в значение False, а FormBorderStyle переводим в значение FixedSingle. Это необходимо для того, что окно формы при работе не потеряло свои параметры. Далее задаем цвет фона в BackColor - Control. В свойстве Icon можно поменять значок приложения (в папке календарь.png). Выделяем Label и ищем свойство Text, именуем в соответствии с рисунком выше, далее находим Font, задаем шрифт Rage Italic и размер 80. Цвет шрифта меняем в свойстве Fore Color – RosyBrown. Повторяем процедуру для Button, но размер шрифта задаем 48. BackColor для обоих объектов задаем тот же, что и для формы.

Нажимаем дважды левой кнопкой мыши на «GO», вставляем в появившийся код, данный фрагмент программы:

```
January f1 = new January(); //создаем конструктор дочерней формы – месяц Январь
```

```
f1.Show(); //открываем форму
this.Hide();  //после открытия новой формы прячем исходную форму
```

Методу `public Form1()` добавляем строку:

```
this.CenterToScreen();//центрируем окно
```

Она необходима для того, чтобы окно Windows располагалось в центре экрана. Окно входа готово, переходим к самому календарю.

Для создания новой формы справа в окне обозревателя решений правой кнопкой мыши нажимаем на имя нашего проекта Calendar – Добавить – Форма (Windows Form)… Называем ее January. Редактируем в соответствии с нижеследующей фотографией (рисунок Г.3):

Рисунок Г.3 - Январь

Размер формы указываем 1117; 640. Остальные свойства меняем в соответствии с предыдущей формой. Теперь добавим 8 Label. Центральная надпись будет написана 68 шрифтом, дни недели – 27. Линии созданы с помощью изображения: из панели элементов выставляем на форму 2 PictureBox, в свойстве Image добавляем картинку (в папке line.png). Размещаем 31 Button и устанавливаем размер 149; 74. Шрифт тот же, размер 50, цвет – DimGray, нумерация как на образце.

Нажимаем на форму, справа, там же, где мы описывали свойства, нажимаем на значок молнии – События. Дважды щелкаем по событию FormClosing. В появившийся код вставляем:

```
Application.Exit();
```

Данный метод необходим для того, чтобы при закрытии окна календаря приложение прекращало свою работу. Без этого метода закроется лишь окно, а приложение так и будет нагружать ваш компьютер в диспетчере задач.

Переходим к программированию форм каждого дня месяца. Дважды щелкаем по кнопке с именем 1. В появившийся код вставляем

```
J1 f2 = new J1(); //создаем конструктор формы первого дня месяца
f2.Show(); //открываем форму
```

Методу `public Form1()` добавляем строку:

```
this.CenterToScreen();
```

Теперь создаем форму дня месяца: правой кнопкой мыши нажимаем на Calendar – Добавить – Форма (Windows Form)… Называем ее J1. Редактируем в соответствии с нижеследующей фотографией (рисунок Г.4):

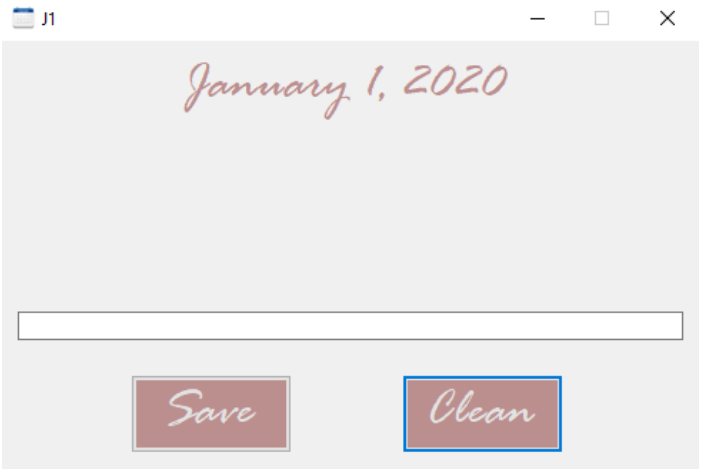

Рисунок Г.4 - Январь 1

Размер формы указываем 506; 339. Остальные свойства меняем в соответствии с предыдущей формой. Добавляем 2 Label, 2 Buttons, 1 TextBox. Центральная надпись будет написана 27 шрифтом. Второй ярлык размещаем под первым, немного левее. Шрифт для него – 14, Rage Italic, подчеркнутый, цвет шрифта черный. Размер кнопок 233; 91, фон – RosyBrown, шрифт – 27, цвет шрифта - ControlLight. Размер TextBox - 466; 20. Дважды кликаем по кнопке Save. Теперь нам необходимо подключить пространство имен: под строкой

```
using System.Windows.Forms;
```

прописываем следующий код:

```
using Calendar.Properties; //подключаем свойства нашей программы
```

Именно с помощью свойств мы будем сохранять наши заметки и планы на день в форме каждого дня. В образователе решений правой кнопкой мыши нажимаем на файл нашего проекта, далее Свойства – Параметры – Имя: J1, тип: string, – Enter. Возвращаемся к файлу J1.cs, для метода `button1_Click` прописываем код:

```
Settings.Default["J1"] = textBox1.Text; //в свойства программы записываем заметку из textBox1
Settings.Default.Save(); //сохраняем свойства
label2.Text = Settings.Default["J1"].ToString(); //присваиваем ярлыку сохраненный текст
```

Для метода `public J1` прописываем следующее:

```
this.CenterToScreen();
label2.Text = Settings.Default["J1"].ToString(); //для того, чтобы текст заметок открывался при новом запуске программы
```

Возвращаемся в конструктор формы J1. Дважды кликаем по кнопке Clean. Прописываем:

```
label2.Text = ""; //очистка поля
Settings.Default["J1"] = label2.Text; //запись пустого поля в свойства
Settings.Default.Save(); //сохранение свойства
label2.Text = Settings.Default["J1"].ToString(); //присваивание заметке текста свойств
```

Возвращаемся в конструктор формы January. Дважды щелкаем по кнопке с именем 2. Создаем новую форму J2 и проделываем те же операции, что и с формой J1. Не забудьте создать новое свойство! И так для каждого дня. После сохраняем весь разработанный проект.

Готово! У нас получился удобный календарь. Также можно добавить остальные месяцы. Я уверена, теперь у вас это получится!

ПРИЛОЖЕНИЕ Д

Сценарий интерактивного упражнения «Описание элементов класса C#»

Тип упражнения: заполнить пропуски. Постановка задачи для ученика: заполнить пропуски.

В пропуски необходимо вписать недостающую информацию, поэтому тип задания будет «впишите» с учетом регистра букв.

Необходимо подготовить исходный текст задания:

Заполните пропуски в следующем коде программы:

```
class Schoolboy
  {
     public  string fio;  // открытое поле ФИО
     public const string school = "Школа №20"; // константа - номер школы
     public static string nom_class;  // статическое поле класса - класс
     public int age;  // открытое поле - возраст
     //string adress; //закомментированный элемент - адрес
  }
```

В данном задании будут пропущены следующие символы и слова: {, public, const, static, //. Вместо них в тексте задания необходимо вписать: «-1-, -2-, -3-, -4-, -5-».

После обработки текст вставляем в окно «Заполнить пропуски», он выглядит следующим образом (рисунок Д.1):

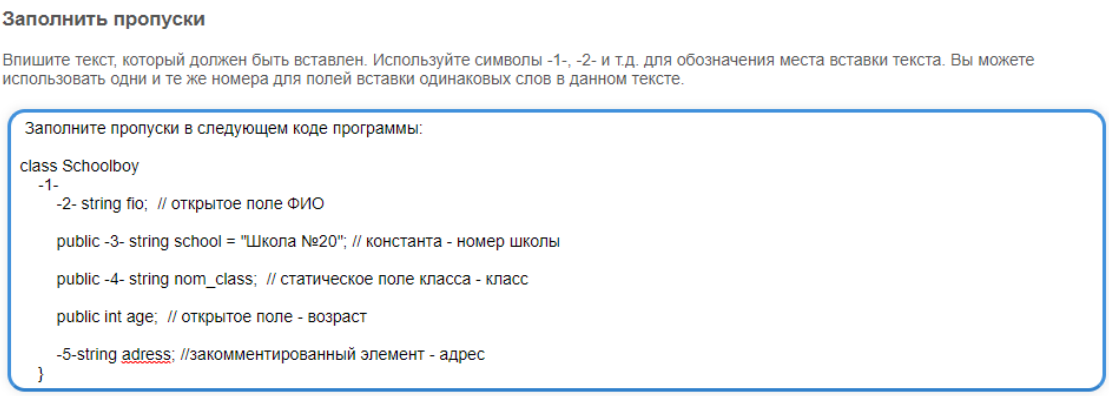

Рисунок Д.1 – Заполнить пропуски в интерактивном упражнении «Описание элементов класса C#»

Теперь для каждого пропуска прописываем верный ответ (рисунок Д.2).

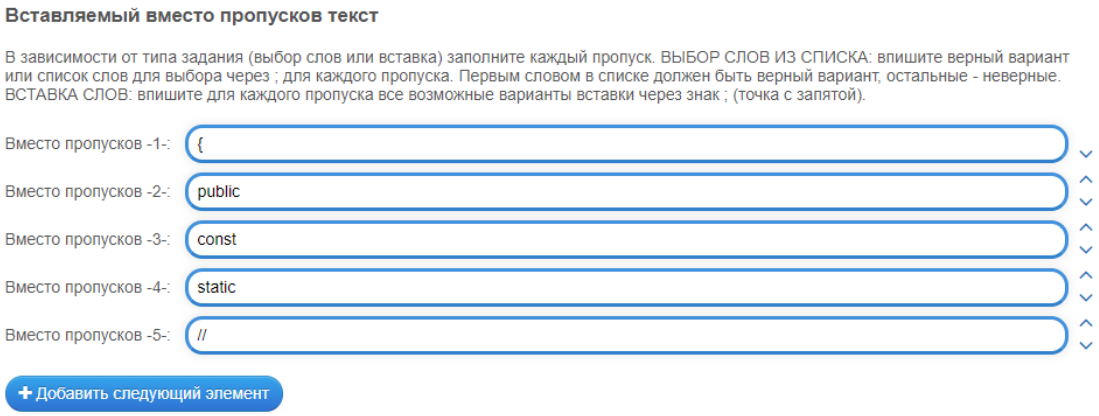

Рисунок Д.2 – Ответы на интерактивное упражнение «Описание элементов класса C#»

Прописываем обратную связь при правильно решенном задании и загружаем фоновую картинку (рисунок Д.3).

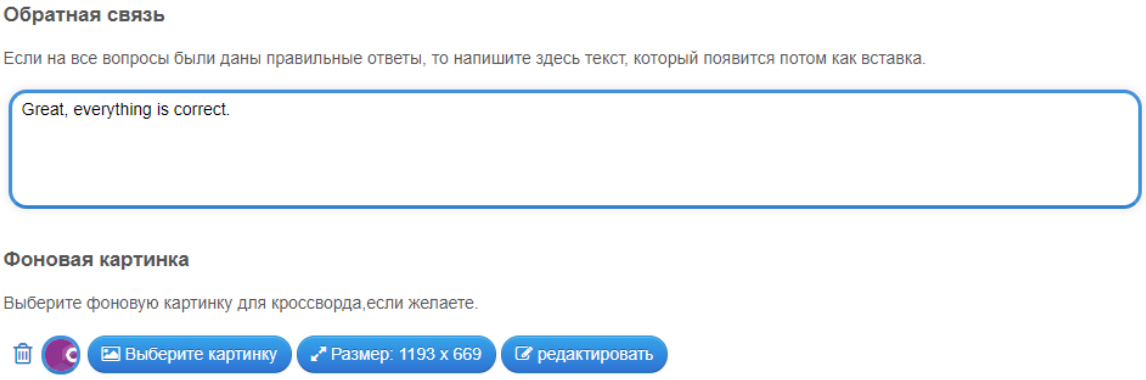

Рисунок Д.3 - Обратная связь интерактивного упражнения «Описание элементов класса C#»

Сохраняем приложение и смотрим на результат (рисунок Д.4).

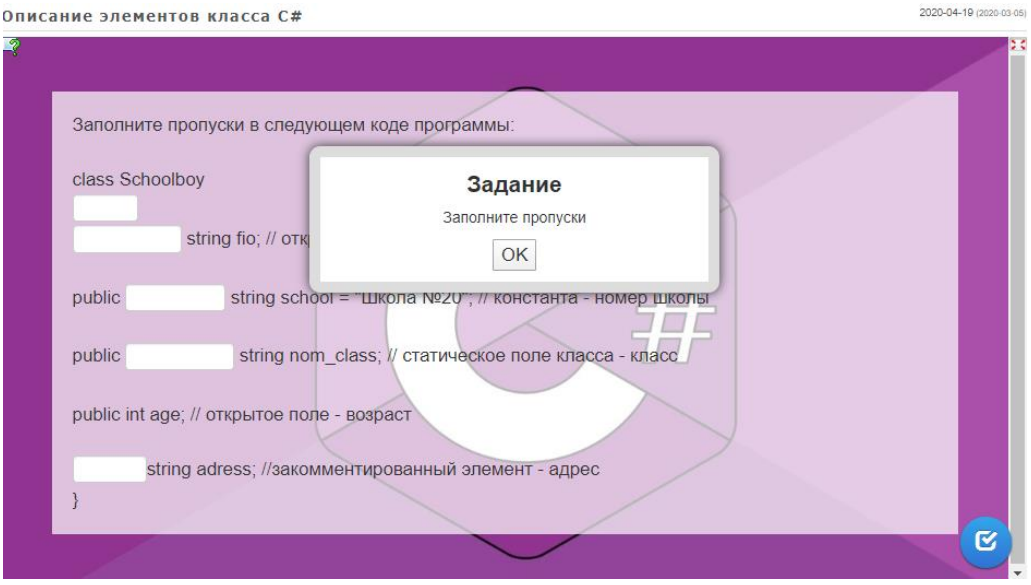

Рисунок Д.4 – Результат интерактивного упражнения «Описание элементов класса C#»

ПРИЛОЖЕНИЕ Е

Сценарий интерактивного упражнения «Спецификаторы класса C#»

Тип упражнения: найти пару. Постановка задачи для ученика: соотнеси определение спецификатора класса с его названием. Если соотнесено верно - карточка пропадет с поля.

Теперь составляем пары. Прописываем спецификатор, рядом с ним – его функцию (рисунки Е.1, Е.2).

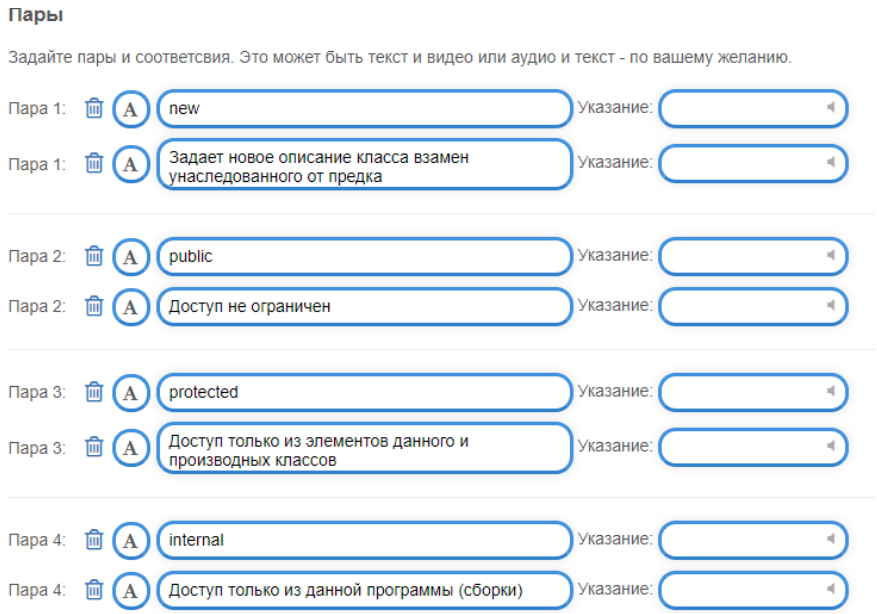

Рисунок Е.1 - Пары 1-4 интерактивного упражнения «Спецификаторы класса C#»

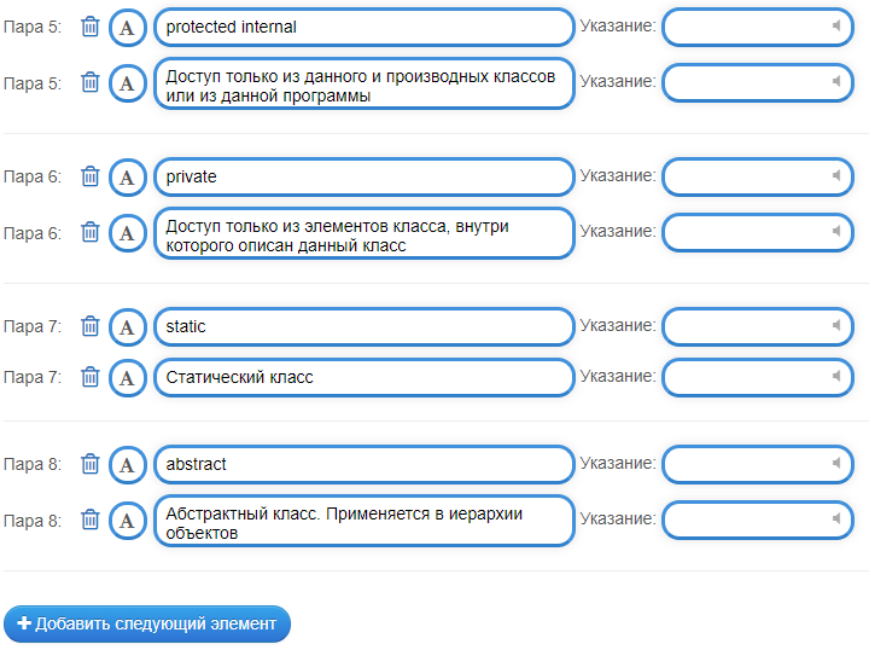

Рисунок Е.2 - Пары 5-8 интерактивного упражнения «Спецификаторы класса C#»

Ставим галочку напротив пункта «Удалять правильно составленные пары». Тип соединения пар выбираем: «Cards one above the other». Прописываем текст для обратной связи и сохраняем приложение (рисунок Е.3).

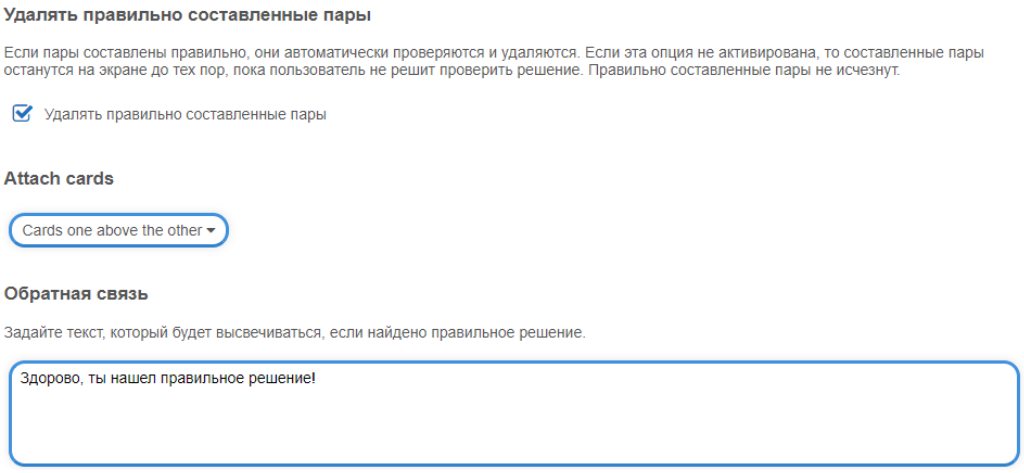

Рисунок Е.3 – Настройки интерактивного упражнения «Спецификаторы класса C#»

Результат (рисунок Е.4):

Рисунок Е.4 – Результат интерактивного упражнения «Спецификаторы класса C#»

ПРИЛОЖЕНИЕ Ж

Сценарий интерактивного упражнения «Методы класса C#»

Тип упражнения: простой порядок. Постановка задачи для ученика: расставить строки кода в правильном порядке. Комментарий идет сразу после кода, к которому относится.

Подготавливаем исходный код программы:

```
class Schoolboy
{
    // объявление класса
    public string fio;
    public void Print()
    {
        // объявление метода
        Console.WriteLine(fio);
        Console.ReadKey();
    }
}
```

Прописываем каждую строку кода в отдельную карточку (рисунок Ж.1).

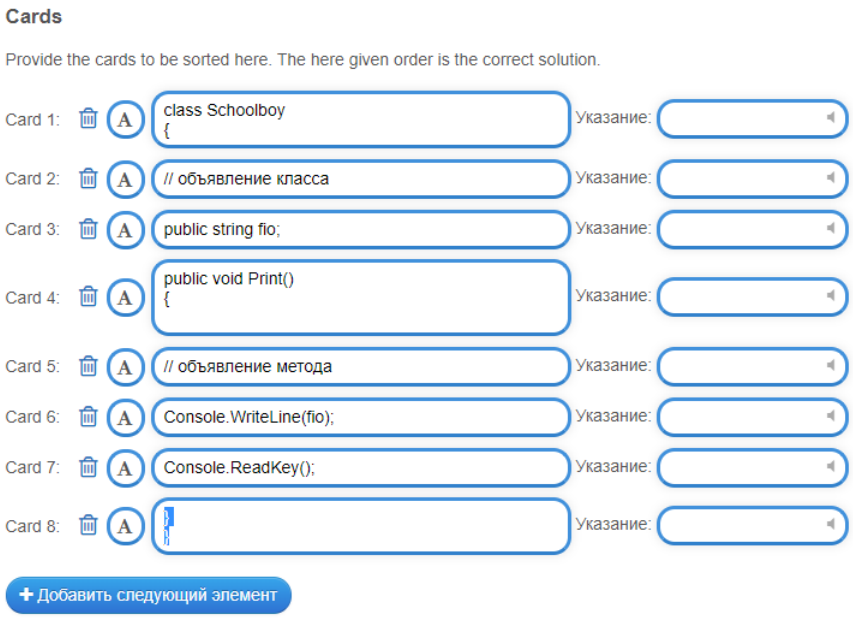

Рисунок Ж.1 - Карточки с кодом для интерактивного упражнения «Методы класса C#»

В расположении карточек выбираем: «Cards fixed placed vertical (order top down)». Прописываем текст для обратной связи и сохраняем приложение (рисунок Ж.2).

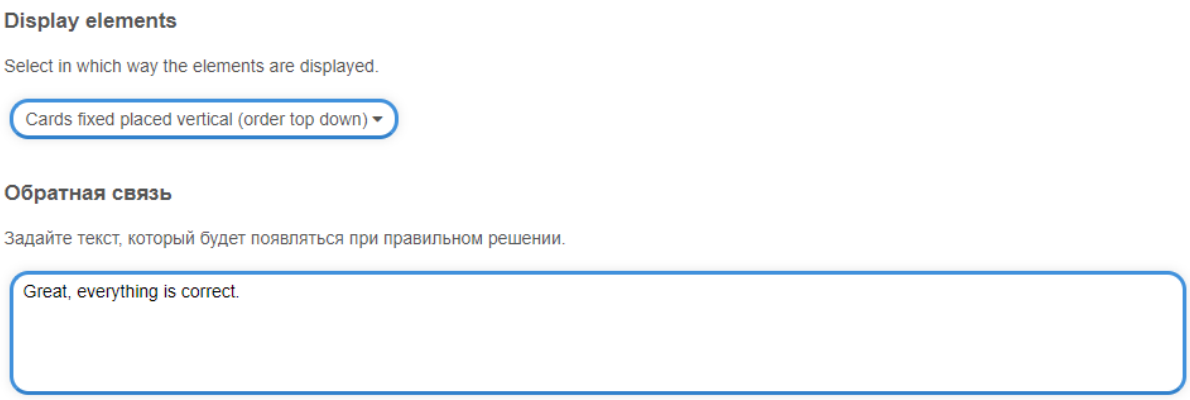

Рисунок Ж.2 - Обратная связь интерактивного упражнения «Методы класса C#»

Результат (рисунок Ж.3):

Рисунок Ж.3 – Результат интерактивного упражнения «Методы класса C#»

ПРИЛОЖЕНИЕ И

Входной тест из состава учебно-методического комплекса (УМК) «Информатика. Углубленный уровень: учебник для 11 класса» авторского коллектива И. Г. Семакина, Е. К. Хеннер, Л. В. Шестаковой

1. В основе концепции объектно-ориентированного программирования лежит понятие:

1) объекта;
2) класса;
3) инкапсуляции.

2. Инкапсуляция – это:

1) свойство системы, позволяющее описать новый класс на основе уже существующего с частично или полностью заимствующейся функциональностью;
2) сущность в адресном пространстве вычислительной системы, появляющаяся при создании экземпляра класса или копирования прототипа (например, после запуска результатов компиляции и связывания исходного кода на выполнение);
3) свойство системы, позволяющее объединить данные и методы, работающие с ними в классе, и скрыть детали реализации от пользователя.

3. В каких отношениях может находиться один класс с другим:

1) отношение наследования;
2) отношение включения;
3) отношение использования.

4. Термин «наследование» обозначает, что...

1) в производных классах присутствует часть состояния родительского класса;
2) производные классы содержат поля и методы родительского;

3) производные классы наследуют модификаторы доступа членов родительского класса.

5. В каком случае вызывается деструктор:

1) создание объекта;
2) удаление объекта;
3) редактирование объекта.

6. Спецификатор доступа public используется для того, чтобы сделать поля и методы доступными ...

1) только внутри класса;
2) только из потомков класса;
3) внутри класса, из потомков и извне;
4) только извне;
5) из потомков и извне.

7. Укажите корректные способы вызова метода класса:

1) имя_объекта.метод(аргументы);
2) имя_класса.имя_объекта(аргументы);
3) имя_класса.метод(аргументы);
4) указатель_на_объект.метод(аргументы);
5) указатель_на_объект->метод(аргументы);
6) указатель_на_объект.имя_объекта->метод(аргументы).

8. Специальный метод класса, который не возвращает никакого значения (даже void), имеет то же имя, что и имя класса, объявляется в открытой части класса, называется ...

1) конструктором;
2) деструктором;
3) компонентной функцией;
4) операторной функцией;

5) дружественной функцией;

6) public-функцией.

9. Укажите корректные способы доступа к открытым полям объекта:

1) имя_объекта.поле;

2) имя_класса.имя_объекта;

3) имя_класса.поле;

4) указатель_на_объект.поле;

5) указатель_на_объект->поле;

6) указатель_на_объект.имя_объекта.

10. Написание текста программы на языке высокого уровня производится на этапе...

1) кодирования;

2) отладки;

3) поддержки;

4) проектирования;

5) разработки спецификации.

11. Для работы с текстовыми данными используются ...

1) объекты класса text;

2) массивы символов;

3) стандартный тип char;

4) объекты класса string.

12. Из всех приведенных ниже утверждений, касающихся объектов и классов, отметьте правильные:

1) значения полей описывают состояние объекта;

2) методы класса описывают поведение объектов;

3) поля класса могут быть разного типа;

4) поля класса должны быть разного типа;

5) в любом классе должно быть как минимум одно поле данных.

13. Спецификатор доступа private используется для того, чтобы сделать поля и методы доступными ...

1) только внутри класса;
2) только из потомков класса;
3) внутри класса, из потомков и извне;
4) только извне;
5) из потомков и извне.

14. Статические (static) поля данных класса...

1) являются общими для всех экземпляров класса (всех объектов);
2) являются неизменяемыми (константными) полями;
3) располагаются в закрытой (private) части класса;
4) могут инициализироваться только в конструкторе;
5) содержат ссылки на другие объекты.

15. Полиморфизм – это:

1) объединение в единую структуру данных полей и методов;
2) способ создания новых классов в качестве наследников уже существующих;
3) возможность переопределения метода с одним интерфейсом в разных классах.

Ответы:

1. 1)	6. 3)	11.4)
2. 3)	7. 1), 5)	12.1), 2), 3)
3. 1)	8. 1)	13.1)
4. 2)	9. 1), 5)	14.1)
5. 2)	10.1)	15.3)