Министерство науки и высшего образования Российской Федерации
Федеральное государственное автономное образовательное учреждение
высшего образования
«Российский государственный профессионально-педагогический университет»

ЭЛЕКТРОННОЕ УЧЕБНОЕ ПОСОБИЕ «ОБЪЕКТНО-ОРИЕНТИРОВАННОЕ ПРОГРАММИРОВАНИЕ ГРАФИЧЕСКИХ ОБЪЕКТОВ НА ЯЗЫКЕ C#»

Выпускная квалификационная работа
по направлению подготовки 44.03.04 Профессиональное обучение
(по отраслям)
профилю подготовки «Информатика и вычислительная техника»
профилизации «Компьютерные технологии»

Идентификационный номер ВКР: 087

Екатеринбург 2019

Министерство науки и высшего образования Российской Федерации
Федеральное государственное автономное образовательное учреждение
высшего образования
«Российский государственный профессионально-педагогический университет»
Институт инженерно-педагогического образования
Кафедра информационных систем и технологий

К ЗАЩИТЕ ДОПУСКАЮ
Заведующий кафедрой ИС
_____ И. А. Суслова
«____» _____ 2019 г.

ВЫПУСКНАЯ КВАЛИФИКАЦИОННАЯ РАБОТА

ЭЛЕКТРОННОЕ УЧЕБНОЕ ПОСОБИЕ «ОБЪЕКТНО-ОРИЕНТИРОВАННОЕ ПРОГРАММИРОВАНИЕ ГРАФИЧЕСКИХ ОБЪЕКТОВ НА ЯЗЫКЕ C#»

Исполнитель: Д. А. Калинин
обучающийся группы № ЗКТ–402С
Руководитель:
старший преподаватель С. Н. Ширева
Нормоконтролер: С. Ю. Ярина

Екатеринбург 2019

АННОТАЦИЯ

Выпускная квалификационная работа состоит из электронного учебного пособия «Объектно-ориентированное программирование графических объектов на языке C#» и пояснительной записки на 71 страницах, содержащей 48 рисунков, 2 таблицы, 30 источников литературы.

Ключевые слова: ВЫСОКОУРОВНЕВОЕ ПРОГРАММИРОВАНИЕ, ЯЗЫК C#, ГРАФИКЧЕСКИЕ ОБЪЕКТЫ, ОБЪЕКТНАЯ ОРИЕНТАЦИЯ.

Калинин Д. А. Электронное учебное пособие «Объектно-ориентированное программирование графических объектов на языке C#»: выпускная квалификационная работа / Д. А. Калинин; Рос. гос. проф.-пед. ун-т, Ин-т инж.-пед. образования, Каф. информ. систем и технологий. — Екатеринбург, 2019. — 70 с.

В работе рассмотрены вопросы обучению дисциплины «Высокоуровневые методы программирования».

Цель работы: разработать электронное учебное пособие «Объектно-ориентированное программирование графических объектов на языке C#». Для этого был проведен анализ рабочей программы направления подготовки 09.03.02 Информационные системы и технологии профиль подготовки «Информационные технологии в медиаиндустрии» дисциплина «Высокоуровневые методы информатики и программирования».

В ходе создания электронного учебного пособия «Объектно-ориентированное программирование графических объектов на языке C#» были использованы:

1. Язык гипертекстовой разметки — HTML.
2. Формальный язык описания внешнего вида документа, написанного с использованием языка разметки — CSS.
3. Мультипарадигменный язык программирования JavaScript.

СОДЕРЖАНИЕ

ВВЕДЕНИЕ

Актуальность данной работы заключается в том, что объектно-ориентированное программирование помогает решить множество классических задач. Но для решения задач связанных с применением графики. Все знания помогут программистам для написания более качественного и гибкого кода.

В качестве примера объектно-ориентированного языка рассматривается C# в будущем. Расширения языка C# предоставляют все виды объектно-ориентированного программирования: больше структуры и модульности, больше абстракции и возможность повторного использования самого языка. Все эти характеристики соответствуют более структурированному, гибкому и простому в обслуживании коду.

Объектно-ориентированное программирование требует оставить в стороне характерные представления о программировании, которые долгие годы считались стандартом. Однако, как только это сделано, объектно-ориентированное программирование становится простым, интуитивно понятным и отличным средством решения многих проблем, которые беспокоят традиционное программное обеспечение. Отсюда вытекает актуальность создания электронного практикума целиком посвященного теме «Объектно-ориентированное программирование графических объектов на языке C#».

Объект исследования — процесс обучения студентов направления подготовки 09.03.02 Информационные системы и технологии профиль подготовки «Информационные технологии в медиаиндустрии» дисциплина «Высокоуровневые методы информатики и программирования».

Предмет исследования — учебные материалы по теме «Объектно-ориентированное программирование на языке C#».

Цель работы является разработка электронного учебного пособия «Объектно-ориентированное программирование графических объектов на языке C#».

В связи с поставленной целью были сформулированы задачи:

1. Проанализировать литературные и интернет-источники по теме «Высокоуровневые методы информатики и программирования».

2. Проанализировать рабочую программу по дисциплине «Высокоуровневые методы информатики и программирования».

3. Разработать структуру электронного учебного пособия «Объектно-ориентированное программирование графических объектов на языке C#».

4. Реализовать электронное учебное пособие «Объектно-ориентированное программирование графических объектов на языке C#».

Выполнение данных задач началось с выполнения курсовой работы по предмету «Педагогические программные средства», где был проведен начальный анализ учебной литературы и учебной программы по теме «Объектно-ориентированное программирование». На следующем этапе было прохождение педагогической практики в ходе, которой был разработан черновой вариант электронного учебного пособия «Объектно-ориентированное программирование графических объектов на языке C#».

1 АНАЛИЗ ИСТОЧНИКОВ ПО ИЗЛОЖЕНИЮ ОСОБЕННОСТЕЙ ПРЕПОДАВАНИЯ ОБЪЕКТНО-ОРИЕНТИРОВАННОГО ПРОГРАММИРОВАНИЯ

1.1 Обзор основной учебной литературы по теме объектно-ориентированное программирование

Анализ литературы при разработке электронного учебного пособия играет очень большую роль, поскольку позволяет отобрать и систематизировать материал.

В учебнике для вузов Давыдова Н. А. «Программирование» [5] присутствуют главы:

1. Основные понятия. В этой главе затронуты темы:
    - алгоритм и его свойства;
    - этапы решения задач на компьютере;
    - история языков программирования;
    - этапы развития технологии программирования; критерии качества программ;
    - структурный подход к программированию;
    - семантический подход к языкам программирования.
2. Введение в C#. В этой главе затронуты темы:
    - системы программирования;
    - интегрированная среда Visual Studio;
    - проект программы; алфавит языка C#;
    - идентификаторы; структура файла проекта программы;
    - типы данных в C#;
    - операторы языка C#;
    - подпрограммы; рекурсия; массивы; строки;

- алгоритмы поиска информации;
- алгоритмы сортировки информации;
- файлы, записи, множества;
- программные модули в Visual Studio.

3. Классы и объекты. В этой главе присутствуют темы:
   - класс-абстракция;
   - класс-структура данных;
   - конструктор;
   - деструктор;
   - операция присваивания;
   - конструктор копии.

4. Объектно-ориентированное проектирование. В этой главе затронуты темы: проектирование сложных систем, декомпозиция сложных систем, объектная модель, классы и объекты, основные этапы создания объектно-ориентированного программного продукта.

5. Реализация объектной модели в языке C#. В этой главе затронуты темы:
   - отличия в объектной модели языка C#;
   - инкапсуляция, наследование, полиморфизм;
   - структура описания классов;
   - составляющие класса поля;
   - составляющие класса, методы, свойства;
   - указатели на методы события, делегирование;
   - совместимость полиморфного присваивания;
   - классовые операции.

6. Визуальное программирование в среде Visual Studio. В этой главе затронуты темы:
   - основы визуального программирования;
   - компоненты в Visual Studio;

- общие свойства компонентов;
- типовой процесс построения компонента;
- графика в Visual Studio;
- реакции на события мыши и клавиатуры;
- обработка исключений;
- базы данных в Visual Studio.

7. Динамические структуры данных. В этой главе затронуты темы:
   - динамическая память и указатели;
   - связанные списки;
   - очереди и стеки;
   - деревья.

В учебном пособии для вузов «Основы алгоритмизации и программирования» Петрова С. Б., Ширевой С. Н. [14] рассмотрены следующие темы, в которых находятся лабораторные работы.

1. В теме «Линейный вычислительный процесс» присутствуют две лабораторные работы:
   - арифметические выражения;
   - программный счет.
2. В теме «Разветвляющийся вычислительный процесс» присутствует две лабораторные работы:
   - условный оператор;
   - оператор выбора.
3. В теме «Циклический вычислительный процесс» присутствует три лабораторные работы:
   - циклы со счетчиком;
   - цикл с условием;
   - вычисления с точностью.
4. В теме «Вспомогательные алгоритмы» присутствует пять лабораторных работ:

- функции;
- процедуры;
- рекурсия;
- стандартные процедуры и функции работы со строками;
- модули.

5. В теме «Алгоритмы работы со структурированными типами данных» присутствует пять лабораторных работ:

- стандартные алгоритмы работы с одномерными массивами;
- формирование массива;
- двумерный массив;
- записи;
- сортировка.

6. В теме «Алгоритмы работы с файлами» присутствуют две лабораторные работы:

- текстовые файлы;
- типизированный файл.

В учебнике для вузов «Программирование на языке высокого уровня» Павловская Т. А. [13] рассмотрены следующие темы, разделенные по частям:

1. Структурное программирование, в этой части рассмотрены темы:

- базовые средства языка C++;
- модульное программирование;
- технология создания программ.

2. Объектно-ориентированное программирование, в этой части рассмотрены темы:

- классы;
- наследование;
- обработка исключительных ситуаций;
- преобразования типов;
- рекомендации по программированию.

3. Стандартная библиотека, в этой части рассмотрены темы:
    - потоковые классы;
    - строки;
    - контейнерные классы;
    - итераторы и функциональные объекты;
    - алгоритмы;
    - средства для численных расчетов;
    - другие средства стандартной библиотеки.

Вывод: материал проанализированной учебной литературы подходит для теоретического материала и практических заданий электронного учебного пособия «Объектно-ориентированное программирование графических объектов на языке C#».

1.2 Обзор интернет-источников по теме объектно-ориентированное программирование

В Интернете существует множество ресурсов в свободном доступе по теме «Программирование в компьютерных системах». Рассмотрим некоторые свободные ресурсы:

1. На сайте «Язык программирования Visual Studio» есть раздел, посвященный теме «Системное программирование» в которой находятся статьи и примеры программных кодов, а также видео-уроки по применению языка Embedded C++ [19].

2. YouTube-канал «Техностим» — это образовательный канал для IT специалистов. Есть курсы по программированию, информация о передовых IT технологиях, использование опыт проектов. Все лекции проходят в вузах России. [22].

3. Блок по языкам программирования «vadimstroganov.com» Вадима Строганова. В блоке присутствуют темы про системное программирование:
    - компилятор;

- интерпретатор;
- транслятор;
- как удалить свободный веб-сервер Apache с сервера;
- как удалить неиспользуемые docker images [28].

4. Сайт «Алгоритмизация. Введение в язык программирования C++. На этом сайте содержатся лекции по программированию. Программы, стоимость и места проведения семинаров, тренингов, конференций [1].

5. Блок по C# объектно-ориентированное программирование «CODE BLOG Программирование». На этом сайте содержится видео уроки, книги и задачи по объектно-ориентированному программированию [14].

6. Сайт Professorweb.ru Net & Web Programming. На этом сайте содержится раздел [30] руководство по C#.

В данном разделе описываются базовые понятия и возможности языка C#, например, такие как условные и арифметические операторы, циклы, операторы перехода, массивы, строки. Также раскрывается объектно-ориентированная природа C#, подробно разбираются классы и их функции-члены (методы, конструкторы, деструкторы, свойства и индексаторы). Так же вы найдете подробное описание перегрузок функций-членов класса.

Подробно описаны интерфейсы, делегаты, события и лямбда-выражения. Также в данный раздел включено подробное описание использования обобщений и основанных на них коллекций, перечислителей и итераторов. В конце приведена очень интересная тема, рассказывающая об особенностях сборки мусора и использовании класса System.GC.

В разделе «Сборки .NET» сначала будет показано, как создавать пространства имен в .NET, и в чем состоит разница между однофайловыми и многофайловыми, а также приватными и разделяемыми сборками. Затем будет описано, как в исполняющей среде определяется местонахождение сборки, и что собой представляет глобальный кэш сборок, конфигурационные файлы приложений (файлы *.config), сборки политик издателя и пространство имен System.Configuration. Затем затрагивается подробное описание ре-

флексии, использование атрибутов, создание доменов приложений. В конце, мы разберем синтаксис промежуточного языка CIL.

В разделе «Потоки и файлы» рассматривается создание многопоточных приложений начиная с описания потока управления (thread) и заканчивая настройкой синхронизации потоков и использованием пула потоков. Также рассказывается про библиотеку TPL, открывающую мир в параллельное программирование; здесь рассказывается про создание и использование задач (Task), а так же описывается основополагающий класс Parallel. Второй частью данного раздела, является описание работы с файлами, в частности чтение, запись, копирование и удаление файлов, так же рассматриваются потоки (stream) и описывается работа с реестром.

Раздел «Основы .NET» описывает общую работу с приложениями .NET — мониторинг, безопасность и локализация. Также рассмотрены два API-интерфейса для работы с расширениями (MAF и MEF).

В разделе «Сетевое программирование (network programming)» приведен материал по описанию основных сетевых протоколов и работы с ними через .NET, работа с сетью P2P, каналами RSS и Atom, очереди сообщений Message Queuing.

Раздел «Оптимизация приложений на .NET Framework» описывает возможности увеличения производительности алгоритмов и приложений является чрезвычайно важным аспектом разработки и может дать вам преимущество перед конкурентами, а вашим пользователям обеспечить удовольствие от использования быстрых и отзывчивых приложений. В данном разделе описываются внутренние особенности операционной системы Windows, среды выполнения CLR и аппаратного обеспечения, влияющие на производительность приложений.

Вывод: материал проанализированных интернет-источников подходит для дополнения и корректировки теоретического материала и практических заданий, которые были уже включенные в электронное учебное пособие

«Объектно-ориентированное программирование графических объектов на языке C#».

1.3 Обзор рабочей программы дисциплины «Высокоуровневые методы информатики и программирования»

В данном разделе будет проводиться анализ рабочей программы дисциплины «Высокоуровневые методы информатики и программирования» направления подготовки 09.03.02 Информационные системы и технологии профиль подготовки «Информационные технологии в медиаиндустрии».

Цель освоения дисциплины «Высокоуровневые методы информатики и программирования»: дальнейшее изучение современного подхода к программированию на основе объектно-ориентированной технологии, приобретение навыков написания программ на языке C#.

Задача: освоить основные понятия, свойства и принципы объектно-ориентированного программирования.

Дисциплина «Высокоуровневые методы информатики и программирования» относится к вариативной части учебного плана.

Для изучения учебной дисциплины необходимы знания, умения и владения, формируемые следующими дисциплинами:

1. Информационные технологии.
2. Технологии обработки информации.

Перечень учебных дисциплин, для которых необходимы знания, умения и владения, формируемые данной учебной дисциплиной:

1. Web-дизайн.
2. Web-программирование.
3. Программирование компьютерной графики.
4. Интеллектуальные системы и технологии.
5. Технологии программирования.

Дисциплина направлена на формирование следующих компетенций:

1. ПК–5 (способность проводить моделирование процессов и систем).
2. ПК–11 (способность к проектированию базовых и прикладных информационных технологий).
3. ПК–17 (способность использовать технологии разработки объектов профессиональной деятельности в областях: машиностроение, приборостроение, техника, образование, медицина, административное управление, юриспруденция, бизнес, предпринимательство, коммерция, менеджмент, банковские системы, безопасность информации).
4. ПК–25 (способность использовать математические методы обработки, анализа и синтеза результатов профессиональных исследований).

В результате освоения дисциплины (модуля) обучающийся должен знать:

1. Что такое объект и класс.
2. Что такое конструктор класса.
3. Основные элементы объектной модели: инкапсуляция, наследование и полиморфизм.
4. Как создаются и используются классы.
5. Основы проектирования информационной системы с точки зрения классов.
6. Элементы моделирования системы с точки зрения классов.
7. Программирование графического интерфейса.

Уметь:

1. Создавать диаграммы классов и взаимодействия с ними.
2. Разрабатывать объектно-ориентированные приложения с помощью какого-нибудь объектно-ориентированного языка программирования.

Владеть:

1. Объектно-ориентированные анализом и проектированием.
2. Созданием простейших приложений с использованием классов объектов и функций.

3. Созданием конструкторов классов.

Общая трудоёмкость дисциплины составляет 4 зачетные единицы (144 час.), семестр изучения — 5, распределение по видам работ представлено в таблице 1.

Таблица 1 — Распределение трудоемкости дисциплины по видам работ

Вид работы	Форма обучения
	очная
	Семестр изучения
	5 сем.
	Кол-во часов
Общая трудоемкость дисциплины по учебному плану	144
Контактная работа, в том числе:	52
Лекции	18
Лабораторные работы	34
Самостоятельная работа студента	92
Промежуточная аттестация, в том числе:	
Экзамен	5 сем.

Дисциплина изучается в течение пятого семестра обучения и состоит из лекций, практических и лабораторных работ. Также в дисциплину входит самостоятельная работа учащихся. Количество часов, предъявленных для изучения тем дисциплины, отображены в таблице 2.

Таблица 2 — Тематический план дисциплины

Наименование разделов и тем дисциплины (модуля)	Сем.	Всего, час.	Вид контактной работы, час.			СРС
			Лекции	Практ. занятия	Лаб. Работы	
1. Объектная модель	5	70	8	18	14	30
2. Объектно-ориентированный анализ и проектирование. Язык UML	5	74	10	20	12	32

Содержание разделов дисциплин «Высокоуровневые методы информатики и программирования»

Раздел 1 «Объектная модель» содержит три темы:

1. Тема «Главные элементы объектной модели». В этой теме рассматриваются:

- главные и дополнительные элементы объектной модели;
- природа объекта;

- состояние и поведение объекта;
- применение объектной модели.

2. Тема «Статические члены класса». В этой теме рассматриваются:
   - класс как контейнер и класс как тип;
   - параметры методов.
3. Тема «Инкапсуляция». В этой теме рассматриваются:
   - главный элемент объектной модели — инкапсуляция;
   - доступ к членам класса;
   - свойства.
4. Тема «Иерархия классов и объектов». В этой теме рассматриваются:
   - виды иерархии;
   - иерархия классов и иерархия объектов;
   - одиночное и множественное наследование;
   - поведение методов при наследовании;
   - полиморфизм;
   - индексаторы.
5. Тема «Абстрактные классы и интерфейсы». В этой теме рассматриваются:
   - использование абстрактных классов и интерфейсов;
   - реализация интерфейсов;
   - создание собственных интерфейсов и использование стандартных интерфейсов.
6. Тема «Делегаты». В этой теме рассматриваются:
   - синтаксис делегатов;
   - массивы делегатов;
   - групповые делегаты;
   - обратные вызовы.
7. Тема «События». В этой теме рассматриваются:

- события;
- механизм работы с событиями.

Раздел 2. Объектно-ориентированный анализ и проектирование. Язык UML. В этом разделе затронута одна тема:

8. Тема «Диаграммы классов». В этой теме рассматриваются:
   - объектно-ориентированный анализ и проектирование;
   - язык UML;
   - генерация кода.

Образовательные технологии

Для изучения дисциплины используются различные образовательные технологии:

1. Технологии проведения занятий в форме диалогового общения, которые переводят образовательный процесс в плоскость активного взаимодействия обучающегося и педагога. Обучающийся занимает активную позицию и престает быть просто слушателем семинаров или лекций. Технологии представлены: групповыми дискуссиями, конструктивный совместный поиск решения проблемы, тренинг (микрообучение и др.), ролевые игры (деловые, организационно-деятельностные, инновационные, коммуникативные).

2. Для поддержки самостоятельной работы обучающихся использованы информационно-коммуникационные образовательные технологии, в частности, облачные технологии, электронная информационно-образовательная среда (ЭИОС), электронные средства обучения и электронно-библиотечные системы. При этом результативность организации самостоятельной работы обучающихся существенно повышается за счет доступности материалов, упорядоченности работ и возможности получения консультации преподавателя.

3. Технология обучения в сотрудничестве применяются при проведении семинарских, практических и лабораторных занятий, нацелены на совместную работу в командах или группах и достижение качественного образовательного результата.

1.4 Методические указания по освоению дисциплины «Высокоуровневые методы информатики и программирования»

1.4.1 Методические указания по проведению лабораторных работ

Проведение лабораторных или практических работ направлено на формирование практических навыков и умений в области решения задач прикладного характера, повышает мотивацию к приобретению профессионально значимых навыков за счет погружения в профессиональную проектную деятельность, позволяет акцентировать внимание студента на совокупности ранее полученных теоретических знаний и отслеживать их практико-ориентированный характер.

В процессе лабораторной или практической работы студенты получают первоначальное знакомство с элементами будущей профессиональной деятельности, формируют представление о принципах практической реализации теоретической информации.

Лабораторные работы:

1. Создание графического классов..
2. Инкапсуляция графического класса.
3. Наследование графического класса.
4. Полиморфизм графики.
5. Создание интерфейса для графических операций.

1.4.2 Методические указания по выполнению лабораторных работ

Основными задачами контрольной работы являются:

1. Практическое применение теоретических знаний, полученных в процессе изучения дисциплины.

2. Определение степени изучения и усвоения студентом программного материала.

3. Привитие ему первичных навыков самостоятельной работы, связанных с поиском научной и учебной литературы.

Формирование способности к анализу и объективной оценке научного и практического материала.

Написание теста предполагает углубление и систематизацию полученных знаний об изучении курса в целом и по выбранной теме в частности; развитие навыков сбора и обобщения практического материала, работы с первоисточниками.

Развитие навыков применения полученных знаний для решения конкретных научно — практических задач, формулирования и обоснования собственной позиции в их решении.

Материалы, необходимые для выполнения практических работ, находятся на кафедре и в электронной информационно-образовательной среде (ЭИОС).

1.4.3 Методические указания по самостоятельной работе

Концепция построения образовательного процесса в системе высшего образования предполагает большой объем самостоятельной работы студента, что требует его системной организации. С этой целью в рамках дисциплины планируется создание концепции организации самостоятельной работы, которая включает в себя: информационно-методическое обеспечение дисциплины, организацию деятельности по самоконтролю, формирование дистанционной поддержки с помощью информационно-коммуникационных технологий.

Самостоятельная работа студента обеспечивает подготовку студента к текущему занятию и включает в себя:

1. Прочтение лекций, изучение основной и дополнительной литературы, по теме «Объектно-ориентированное программирование».

2. Выполнение заданий, работу по отдельным темам учебных дисциплин в соответствии с учебным планом 09.03.02 «Информационные системы и технологии» по дисциплине «Высокоуровневые методы информатики и программирования».

3. Подготовку к итоговому зачету по дисциплине «Высокоуровневые методы информатики и программирования».

1.5 Информация по электронным учебным пособиям

1.5.1 Общие понятия и определения

Электронное издание — издание, зарегистрированное на носителе данных, предназначенное для использования электронных устройств электронного документа, прошедшего редакционно-издательскую обработку, предназначенное для распространения в неизмененном виде с выходными данными. В одном электронном издании могут быть выделены источники информации, средства создания и обработки информации, управленческие структуры. Электронное издание может быть исполнено на любом электронном носителе — магнитном, оптическом, а также опубликовано в электронной компьютерной сети [16].

При этом образовательное электронное издание или эквивалентное электронное учебное пособие представляет собой электронное издание, содержащее систематизированный материал по соответствующей научно – практической области знаний, обеспечивающее творческое и активное овладение студентами знаниями, навыками и умениями в этой области.

Одним из видов компьютерного инструментария для образовательных целей, который можно рассматривать как компоненты, является электронный учебник [21].

Электронное учебное пособие — электронный учебный курс, частично или полностью заменяющий или дополняющий учебник и официально утвержденное в качестве данного вида издания [30].

Электронное учебное пособие — это программно-методический комплекс, обеспечивающий возможность самостоятельного освоения учебного курса или его большого раздела.

Электронный учебник или курс обычно содержит три компонента: презентационный компонент, в котором излагается основная информационная часть курса; упражнения, способствующие закреплению полученных знаний; тесты, позволяющие объективно оценить знания студента. Стоит отметить, что все компоненты тесно взаимосвязаны, и их доля в PES варьируется в зависимости от курса.

1.5.2 Описание компонентов электронного средства обучения

Компоненты электронного средства обучения могут быть разными, их наличие и количество определяется спецификой курса:

1. Учебник — учебное издание, дополняющее или частично заменяющее учебник, официально утвержденное в качестве данного вида издания.
2. Конспекты лекций — учебное пособие, содержащее теоретическую информацию в сжатой форме в соответствии с задачами дисциплины и отражающее авторское видение выбора и изложения учебного материала.
3. Сборник задач и упражнений — учебное пособие, содержащее комплекс заданий для обучения, контроля и самоконтроля в соответствии с целями дисциплины.
4. Задачник — это форма электронного учебника, наиболее естественно выполняющая функцию обучения. Главное в электронном задачнике дозированная помощь. Учащийся получает именно ту и только что информацию, которая необходима для решения конкретной задачи. Основная цель

данного пособия — подбор заданий, которые перекрывают весь теоретический материал.

5. Практические задания — учебное пособие по отработке практических навыков. Он должен:

- предоставить слушателю информацию о теме, цели и порядке проведения занятия;
- контроль знаний каждого студента;
- предоставить слушателю информацию о правильности ответа;
- предъявлять необходимый теоретический материал или методику решения задач;
- оценка знаний учащихся;
- обеспечить обратную связь в режиме «учитель-ЕС-ученик».

6. Лабораторная работа — учебное пособие, которое должно содержать средства для самостоятельной подготовки студента к работе, допуска к работе, эксперимента, обработки экспериментальных данных, оформления результатов лабораторной работы, защиты труда.

7. Методические рекомендации по практической подготовке учебного пособия, содержащее задания в соответствии с задачами дисциплины, продуктивными способами выполнения заданий и различными способами поддержки обучающегося.

8. Методические указания к лабораторным работам — пособие, содержащее задания и рекомендации по планированию и выполнению лабораторного эксперимента, а также материалы для самоконтроля и контроля (порядок защиты), описание специального оборудования и программного обеспечения, основные ожидаемые результаты и форму отчета.

9. Методические указания по курсовому проектированию — учебное пособие, содержащее комплексное задание преимущественно в контексте будущей профессиональной деятельности в соответствии с задачами дисциплины, а также рекомендации по планированию и выполнению задания, форму отчета, порядок защиты и правила оценки проекта.

10. Методические указания к практике — учебное пособие, содержащее сведения о месте, порядке и продолжительности практики, обязанностях научного руководителя и стажера, сроках, содержании и порядке отчетности по результатам практики (учебной, производственной, преддипломной, научно-исследовательской, научно-педагогической).

11. Методические указания по оформлению диплома — учебное пособие, содержащее сведения об организации дипломного проектирования, образцы тем и порядок их утверждения, основные этапы проектирования, структуру и содержание пояснительной записки к дипломному проекту, порядок защиты [15, с. 34].

12. Тренажеры — сборник контрольных материалов, учебное пособие, содержащее комплекс контрольных заданий в соответствии с задачами дисциплины.

13. Хрестоматия является учебно-практическое издание, содержащее систематически подобранные литературно-художественные, официальные, научные и иные произведения или фрагменты из них, составляющие объект исследования.

14. Сборник учебных материалов — учебное пособие, содержащее описание конкретных ситуаций, возникающих в профессиональной деятельности.

15. Справочник — издание практического назначения, с кратким изложением информации в систематизированной форме, основанное на выборочном прочтении, для того, чтобы его можно было быстро и легко запросить о нем.

16. Словарь — справочник, данные в котором упорядочены с помощью разбивки на небольшие статьи, отсортированные по названию или тематике. Имеются энциклопедические и лингвистические словари.

17. Глоссарий — словарь узкоспециализированных терминов в какой-либо отрасли знаний с толкованием, иногда переводом на другой язык, комментариями и примерами. В различных областях знания глоссарии могут

представлять собой специальное исследование, представляющее собой список терминов, требующих комментариев и пояснений.

18. Энциклопедия — это основная форма электронного учебника. Энциклопедии называют также научное справочное пособие, содержащее обозрение наук или дисциплин (преимущественно в форме словаря) для электронных энциклопедий характерен соответствующий сервис: ссылки, закладки, возможность повтора анимации и звуковых записей, поиск по ключевым словам [20, с. 327].

1.5.3 Требования к электронным учебным пособиям

При разработке электронного учебного пособия следует руководствоваться общими требованиями, которые выделяют его среди других средств обучения и способствуют максимальной направленности пособия на качественное содержание информации. Общие требования заключаются в следующем:

1. Логичность выделения структурной единицы, соотнесенность ее с содержанием раздела, наличие для учащегося возможности прямой навигации из любой структурной единицы в любую другую, логически с ней связанную, возможность перейти от данного раздела к другому разделу курса.

2. Материалы должны быть построены таким образом, чтобы обучаемый мог перейти от деятельности, выполняемой под руководством преподавателя, к деятельности, организуемой самостоятельно, к максимальной замене преподавательского контроля самоконтролем. Поэтому они должны содержать подробное описание рациональных приемов описанных видов деятельности, критериев правильности решений, рекомендации по эффективному использованию консультаций.

3. Информационные технологии предоставляют в распоряжение преподавателя мощный набор инструментов, которые должны эффективно ис-

пользоваться для достижения целей учебного процесса при дистанционном обучении [26].

Выделяют дидактические требования [22]:

1. Научности обучения — это обеспечение достаточной глубины и корректности изложения учебного материала с учетом последних достижений науки.

2. Доступности обучения — обеспечение соответствия степени теоретической сложности и глубины изучения возрастным и индивидуальным особенностям учащихся, не допущение чрезмерной усложненности и перегруженности учебного материала.

3. Систематичности и последовательности обучения — обеспечение формирования знаний, умений и навыков, учащихся в определенной логически связанной последовательности с обеспечением преемственности.

4. Наглядности обучения — обеспечение чувственного восприятия учащимися объектов, процессов, явлений.

5. Сознательности и активности обучения — обеспечение самостоятельных и активных действий, учащихся по извлечению учебной информации.

6. Прочности усвоения знаний — обеспечение закрепления знаний.

Электронное учебное пособие должно включать руководство пользователя, содержащее:

1. Техническое руководство, в котором дана краткая характеристика внутренней навигации электронного учебного пособия.

2. Методическое руководство, в котором приведены рекомендации по использованию электронного учебного пособия в учебном процессе.

К основным новым дидактическим требованиям относятся:

3. Структуризация учебного материала и структурно-функциональная связанность — обеспечение представления учебного материала с разбивкой на структурные единицы с обозначением структурно-функциональных связей между ними, отражающих внутреннюю логику изучаемого материала.

4. Интерактивность обучения — обеспечение взаимодействие учащегося с электронным учебником (интерактивного диалога, учащегося с электронным средством обучения).

5. Адаптивность обучения — обеспечение приспособления процесса обучения к уровню знаний, умений, психологических особенностей учащегося, работающего с учебным электронным изданием.

Технологические требования к электронному учебному пособию [21]:

1. Полное соответствие содержания электронного учебного ресурса государственному образовательному стандарту и типовой учебной программе соответствующей учебной дисциплины.

2. Текст учебного материала должен подвергаться редакторской обработке и корректорской правке.

3. Содержательная часть электронной учебного ресурса должна быть отрецензирована и получить соответствующий гриф.

4. В электронном учебном пособии должна быть предусмотрена комплексность, достаточная для самостоятельного изучения и практического усвоения учебного материала соответствующей дисциплины учащимися при консультационной поддержке и контроле со стороны преподавателей.

5. Должна быть четкая структуризация учебного материала с выделением совокупности взаимосвязанных понятий и закономерностей, разделов и объектов изучения.

6. Представление текстового учебного материала должно быть предельно лаконично с применение гипертекстовой разметки при его изложении.

7. В электронном учебном пособии должно быть применение наряду со статическими математическими зависимостями имитационных компьютерных моделей изучаемых объектов.

8. Обязательным условием при составлении электронного учебного пособия является наличие и использование таблиц, схем, статических и ани-

мированных графических изображений, видеофрагментов, повышающих наглядность изложения текстового учебного материала [26, с. 105].

9. При составлении структуры электронного учебного пособия должно быть обязательно предусмотрено включение в состав учебных ресурсов:

- практических заданий, предназначенных для формирования умений и навыков применения теоретических знаний в отношении каждого объекта изучения;
- средств выполнения виртуальных и/или реальных лабораторных исследований каждого объекта изучения;
- средств контроля и самоконтроля полученных знаний, умений и навыков;
- методических рекомендаций по целесообразному порядку изучения как всей учебной дисциплины;
- средств регистрации учащихся, их действий и результатов, получаемых при изучении учебной дисциплины;
- средств оперативного взаимодействия с преподавателями и другими учащимися.

10. В тексте учебного пособия должны выделяться фоном базовые термины курса, входящие в предметный указатель с предоставлением возможности обучаемому раскрыть смысл термина гиперпереходом в предметный указатель с возвратом к основному тексту.

11. В электронных учебных пособиях должно быть предусмотрено удобство и наглядность навигации, простота и оперативность переходов к требуемым разделам, объектам и средствам обучения.

12. Электронное учебное пособие должно обязательно заканчиваться списком основной литературы и дополнительным списком отечественных и зарубежных интернет-источников, связанных с тематикой пособия.

13. Электронное учебное пособие должно заканчиваться предметным указателем, необходимо представить дополнительные списки использован-

ных мультимедийных иллюстраций: рисунков, анимации, графиков, таблиц, фотографий, видео.

14. Все ссылки в электронном учебном пособии должны иметь возможность возврата к основному тексту.

15. Должна быть предусмотрена возможность применения на персональных компьютерах средней производительности с типовым набором аппаратно-программных средств.

16. Должна быть предусмотрена возможность обеспечения переносимости электронных учебных ресурсов на различные платформы.

17. Должна быть предусмотрена возможность доставки электронных ресурсов обучающимся с применением компьютерных сетей и/или на компакт-дисках [27, с. 10].

1.5.4 Основные этапы разработки

Этапы разработки электронного учебного пособия [11]:

1. Выбор источников.
2. Заключение договоров с авторами о праве на переработку.
3. Разработка оглавления и перечня понятий (индекса).
4. Переработка текстов в модули по разделам и создание помощника.
5. Реализация гипертекста в электронной форме.
6. Разработка компьютерной поддержки.
7. Отбор материала для мультимедийного воплощения.
8. Разработка звукового сопровождения.
9. Реализация звукового сопровождения.
10. Подготовка материала для визуализации.
11. Визуализация материала.

Рассмотреть методические рекомендации по разработке электронного учебника.

На первом этапе разработки электронного учебника целесообразно выбрать таких печатных и электронных изданий в качестве источников, которые наиболее полно соответствуют стандартной программе, лаконичны и удобны для создания гипертекстов, содержат большое количество примеров и задач, имеются в удобных форматах.

На втором этапе заключения договоров из полученного набора источников отбираются те, которые имеют оптимальное соотношение цены и качества.

На третьем этапе разрабатывается оглавление, то есть, материал разделен на разделы, состоящие из модулей, минимальных по объему, но замкнутых по содержанию, а также список понятий, необходимых и достаточных для освоения предмета.

На четвертом этапе исходные тексты обрабатываются в соответствии с содержанием, индексом и структурой модулей, также исключаются тексты, не включенные в списки, и те, которые не находятся в источниках, разрабатывается система контекстных ссылок, определяются связи между модулями и другими гипертекстовыми ссылками.

На пятом этапе гипертекст реализуется в электронной форме.

В результате получается примитивное электронное издание, которое уже можно использовать в образовательных целях. Многие такие электронные издания metinee называют электронными книгами. У него практически нет шансов на коммерческий успех, потому что студенты его не купят.

На шестом этапе разрабатывается компьютерная поддержка: определяется, какие математические действия в каждом конкретном случае назначаются компьютеру и в какой форме должен быть представлен ответ компьютера.

В результате, рабочий электронный учебник, который обладает свойствами, которые делают его необходимым для студентов, полезным для

аудиторных занятий и удобным для преподавателей. Он может распространяться на коммерческой основе.

На седьмом этапе изменяются способы объяснения отдельных понятий и утверждений и подбираются тексты для замены материалами.

На восьмом этапе, тексты звукового отдельные модули предназначены для разгрузки экрана от текстовой информации и использования слуховой памяти учащегося для облегчения понимания и запоминания изучаемого материала.

На девятом этапе разработанные тексты звукового записываются на диктофон и реализуются на компьютере.

На десятом этапе, сценарии визуализации модулей для достижения наибольшей наглядности, максимальной разгрузки экрана от текстовой информации и использования эмоциональной памяти учащегося для облегчения понимания и запоминания изучаемого материала.

На одиннадцатом этапе визуализируются тексты, т. е. компьютерное воплощение разработанных сценариев с использованием рисунков, графиков и, возможно, анимации.

Разработка электронного оборудования подходит к концу и начинается подготовка к его эксплуатации [25, с. 50].

2 ОПИСАНИЕ ЭЛЕКТРОННОГО УЧЕБНОГО ПОСОБИЯ «ОБЪЕКТНО-ОРИЕНТИРОВАННОЕ ПРОГРАММИРОВАНИЕ ГРАФИЧЕСКИХ ОБЪЕКТОВ НА ЯЗЫКЕ C#»

2.1 Структура электронного учебного пособия «Объектно-ориентированное программирование графических объектов на языке C#»

Структура электронного учебного пособия «Объектно-ориентированное программирование графических объектов на языке C#» отображена на рисунке 1.

Рисунок 1 — Структура электронного учебного пособия

Структура электронного учебного пособия «Объектно-ориентированное программирование графических объектов на языке C#» представлена двумя блоками.

Теоретический блок, который содержит теоретические сведения по каждому разделу дисциплины:

1. Создание графического класса
2. Инкапсуляция графического класса.
3. Наследование графического класса.
4. Полиморфизм графики.
5. Создание интерфейса для графических операций.

Лабораторный блок, который содержит индивидуальные задания по темам из теоретического блока, а также 10 вариантов индивидуальных заданий по каждой практической работы:

1. Создание графического класса
2. Инкапсуляция графического класса.
3. Наследование графического класса.
4. Полиморфизм графики.
5. Создание интерфейса для графических операций.

Программный код электронного учебного пособия находится в файлах формата языка гипертекстовой разметки (html) и каскадных таблиц стилей (css).

1. Файл с расположением элементов «Index.html». Файл «Index.html» содержит программный код, изображенный на рисунке 2.

2. Программный код начальных страниц находится на одном уровне каталога, где находится файл «Index.html». Файлы с начальными страницами:

- 11.html — левая и правая сторона страницы;
- 44.html — левая и правая сторона содержания формы;
- 66.html — верхнее меню;
- 77.html — название электронного учебного пособия;
- 77_1.html — содержание лекционного материала по темам;

- 77_2.html — содержание практического материала по темам;
- 88.html — начальная страница;
- 99.html — нижняя строка содержимом формы.

```
<html>
<head>
<title> ЭУП "Изучение высокоуровнего программирования" </title>
</head>


<frameset cols="10,400,10" rows="100" frameborder="0" framespacing="0" height="2000">
<frame src="11.html" name="f11" >

<frameset cols="100" rows="100,40,600,50" frameborder="0" framespacing="0">
<frame src="66.html" name="f66">
<frame src="77.html" name="f77" >

<frameset cols="50,100,400,50" rows="100" frameborder="0" framespacing="0">
<frame src="33.html" name="f21">
<frame src="22.html" name="f22">
<frame src="88.html" name="f88">
<frame src="33.html" name="f33">
</frameset>

<frame src="99.html" name="f99">
</frameset>

<frame src="44.html" name="f44">

</frameset>


</html>
```

Рисунок 2 — Программный код файла «Index.html»

3. Учебные страницы размещаются в файлах папки «theory», где находятся страницы лекционного материала, тестовых заданий, лабораторных работ и индивидуальных заданий.

- lec1.html — лекционный материал по теме «Создание графического класса»;
- lec2.html — лекционный материал по теме «Инкапсуляция графического класса»;
- lec3.html — лекционный материал по теме «Наследование графического класса»;
- lec4.html — лекционный материал по теме «Полиморфизм графики»;

- lec5.html — лекционный материал по теме «Создание интерфейса для графических операций»;
- test1.html — тестовые задания по проверке усвоения лекционного материала по теме «Создание графического класса»;
- test2.html — тестовые задания по проверке усвоения лекционного материала «Инкапсуляция графического класса»;
- test3.html — тестовые задания по проверке усвоения лекционного материала «Наследование графического класса»;
- test4.html — тестовые задания по проверке усвоения лекционного материала «Полиморфизм графики»;
- test5.html — тестовые задания по проверке усвоения лекционного материала «Интерфейс»;
- lab1.html — лабораторные задания по теме «Создание графического класса»;
- lab2.html — лабораторные задания по теме «Инкапсуляция графического класса»;
- lab3.html — лабораторные задания по теме «Наследование графического класса»;
- lab4.html — лабораторные задания по теме «Полиморфизм графики»;
- lab5.html — лабораторные задания по теме «Создание интерфейса для графических операций»;
- iz1.html — перечень индивидуальных заданий по вариантам для проверки усвоения темы «Создание графического класса»;
- iz2.html — перечень индивидуальных заданий по вариантам для проверки усвоения темы «Инкапсуляция графического класса»;
- iz3.html — перечень индивидуальных заданий по вариантам для проверки усвоения темы «Наследование графического класса»;

- iz4.html — перечень индивидуальных заданий по вариантам для проверки усвоения темы «Полиморфизм графики»;
- iz5.html — перечень индивидуальных заданий по вариантам для проверки усвоения темы «Создание интерфейса для графических операций»;

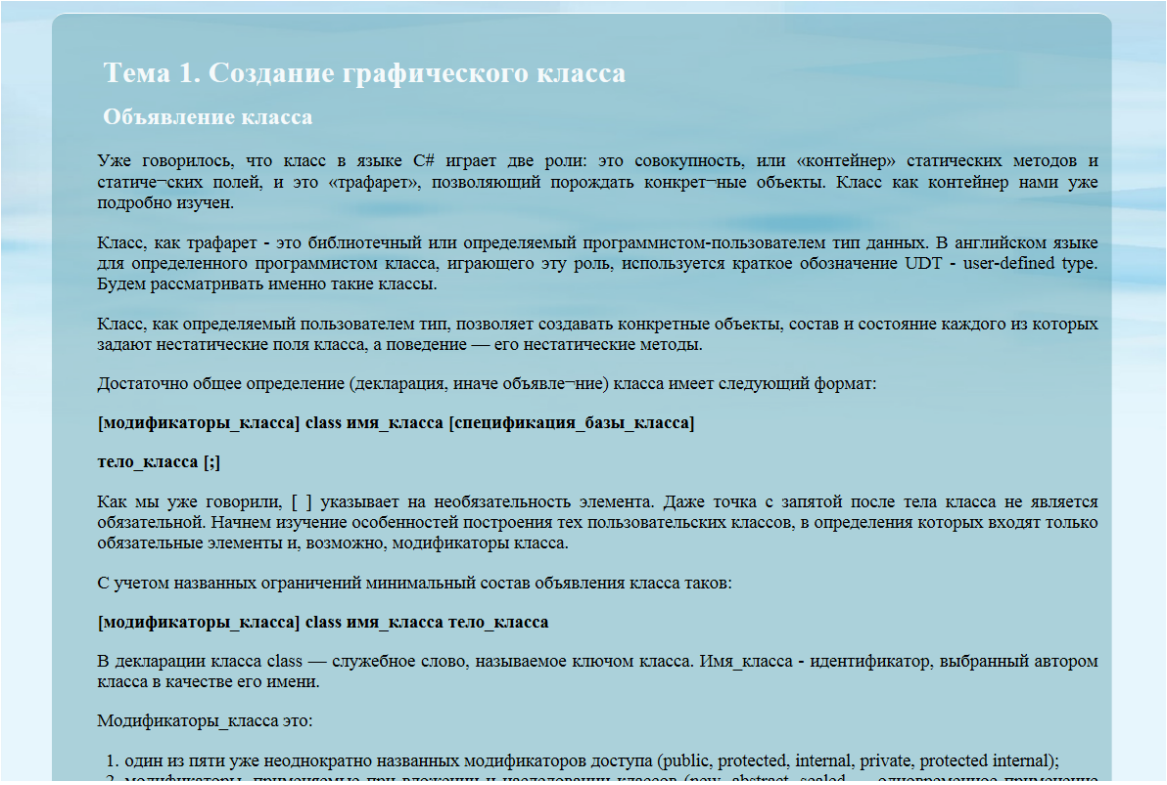
4. Файлы с наборами форматирования текста и изображений находятся в папке «css». На эту папку ссылаются учебные страницы:
   - main.css — основной контейнер форматов и стилей;
   - main1.css — дополнительный контейнер форматов и стилей.
5. Изображения находятся в папке «images».

2.2 Описание теоретического блока

Электронное учебное пособие «Объектно-ориентированное программирование графических объектов на языке C#» содержит лекционный материал, который распределен по темам.

Тема «Создание графического класса»

Начало лекция «Тема 1. Создание классов» изображена на рисунке 3.



Тема 1. Создание графического класса

Объявление класса

Уже говорилось, что класс в языке C# играет две роли: это совокупность, или «контейнер» статических методов и статиче¬ских полей, и это «трафарет», позволяющий порождать конкрет¬ные объекты. Класс как контейнер нами уже подробно изучен.

Класс, как трафарет - это библиотечный или определяемый программистом-пользователем тип данных. В английском языке для определенного программистом класса, играющего эту роль, используется краткое обозначение UDT - user-defined type. Будем рассматривать именно такие классы.

Класс, как определяемый пользователем тип, позволяет создавать конкретные объекты, состав и состояние каждого из которых задают нестатические поля класса, а поведение — его нестатические методы.

Достаточно общее определение (декларация, иначе объявле¬ние) класса имеет следующий формат:

[модификаторы_класса] class имя_класса [спецификация_базы_класса]

тело_класса [;]

Как мы уже говорили, [ ] указывает на необязательность элемента. Даже точка с запятой после тела класса не является обязательной. Начнем изучение особенностей построения тех пользовательских классов, в определения которых входят только обязательные элементы и, возможно, модификаторы класса.

С учетом названных ограничений минимальный состав объявления класса таков:

[модификаторы_класса] class имя_класса тело_класса

В декларации класса class — служебное слово, называемое ключом класса. Имя_класса - идентификатор, выбранный автором класса в качестве его имени.

Модификаторы_класса это:

1. один из пяти уже неоднократно названных модификаторов доступа (public, protected, internal, private, protected internal);

Рисунок 3 — Лекция «Тема 1. Создание классов»

В этой теме лекционного материала «Создание графического класса» состоит из трех разделов.

В разделе «Объявление класса» рассказывается:

1. О среде CLR, которая использует Windows, как дополнительно реализационный интерфейс, и графических устройств (GDI) вызванного GDI+. А также о его свойствах и взаимодействиях с форми Windows Forms и элементами управления.

2. О пространстве имен System.Drawing, которое обеспечивает доступ к функциональным возможностям графического интерфейса GDI+, пространства имен System.Drawing.Drawing2D, System.Drawing.Imaging, и System.Drawing.Text, которые обеспечивают дополнительные функциональные возможности.

3. О классе Graphics предоставляет методы рисования на устройстве отображения. Такие классы, как Rectangle и Point, инкапсулируют элементы GDI+. Класс Pen используется для рисования линий и кривых, а классы, производные от абстрактного класса Brush, используются для заливки фигур.

В разделе «Поля объектов» рассказывается:

1. О двух этапах работы с графикой. Это создание объекта Graphics, использование объекта Graphics для рисования линий и фигур, отображения текста или изображения и управления ими.

2. О создании объекта Graphics получение объекта Graphics через объект painteventargs при обработке события Paint формы или элемента управления. Это обычный способ получения ссылки на графический объект при создании кода рисования элементов управления. Подобным образом объект как свойство объекта printpageeventargs при обработке события printpage для printdocument.

3. О создании обработчика события painteventhandler для элементов управления или объекта printpage для printdocument.

В разделе «Объявления методов объектов» рассказывается о идентификации объекта Graphics из объекта PaintEventArgs события Paint. Пример отображения показан на рисунке 4.

```
private void Form1_Paint(object sender,
 System.Windows.Forms.PaintEventArgs pe)
{ // Declares the Graphics object and sets it to the Graphics object
  // supplied in the PaintEventArgs.
    Graphics g = pe.Graphics;
    // Insert code to paint the form here.}
```

Рисунок 4 — Программный код ссылки на объект

В разделе «Объявления методов объектов» рассказывается об идентификации объекта Graphics с помощью метода CreateGraphics формы или элемента управления. Пример отображения показан на рисунке 4 [8, с . 291].

```
Graphics g;
// Sets g to a graphics object representing the drawing surface of the
// control or form g is a member of.
g = this.CreateGraphics().
```

Рисунок 5 — Программный код создания объектов

Также в лекции присутствуют элементы программного кода, изображен на рисунке 6.

```
using System;
class Link
{       static Link beg;
Link next;
int numb;
public void add(int numb)
{this.numb = numb;
if (beg == null) // Список пуст
{ beg = this; return; }
Link temp = beg;
while (temp,next != null)
temp = temp,next;
temp,next = this;
}
static public void display()
{ Link temp = beg;
 while (temp != null;
{ Console.Write(temp.numb + " ");
          temp = temp.next;
}
Console. WriteLine();
}
}
class Program
{
static void Main()
{
Link a = new Link(), b = new Link(), c = new Link();
a.add(7);
b.add(-4);
c.add(0);
Link.display ();
}
}
```

Результат выполнения программы:

```
7 -4 0
```

Рисунок 6 — Отображение программного кода

В конце лекции «Тема 1. Создание классов» присутствует ссылка на тестовую форму для самопроверки. Начало тестовой формы отображено на рисунке 7.

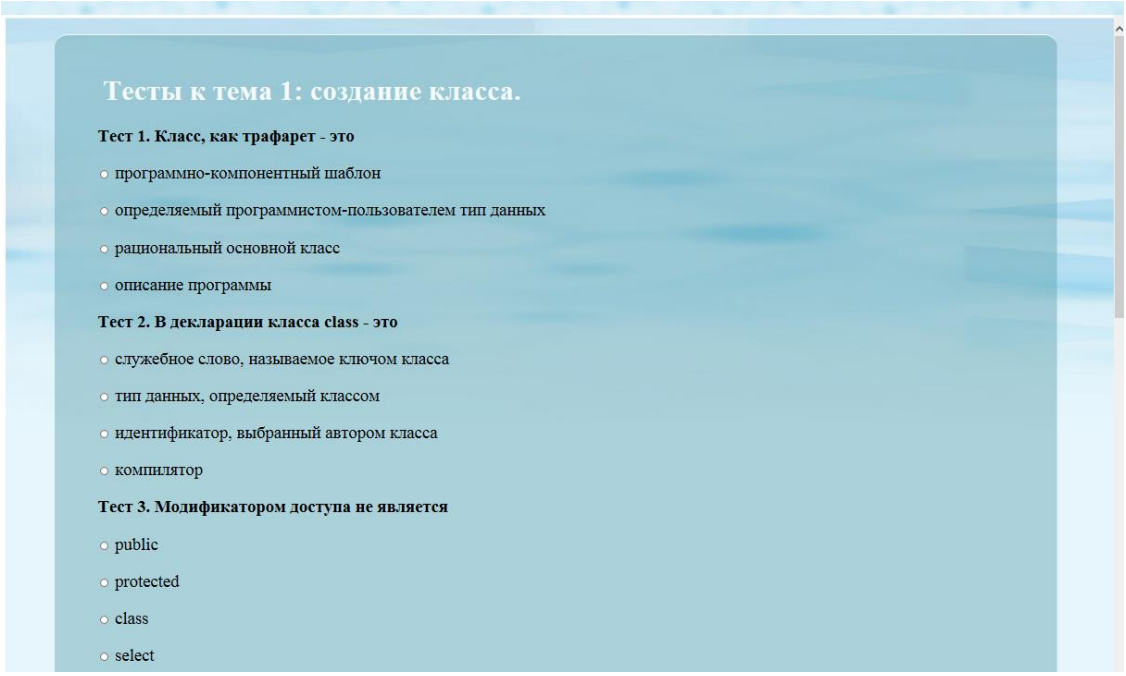


Рисунок 7 — Тестовая форма для лекции «Тема 1. Создание графического классов»

Тема «Инкапсуляция графического класса»

Начало лекция «Тема 2. Инкапсуляция графического класса» изображена на рисунке 8.

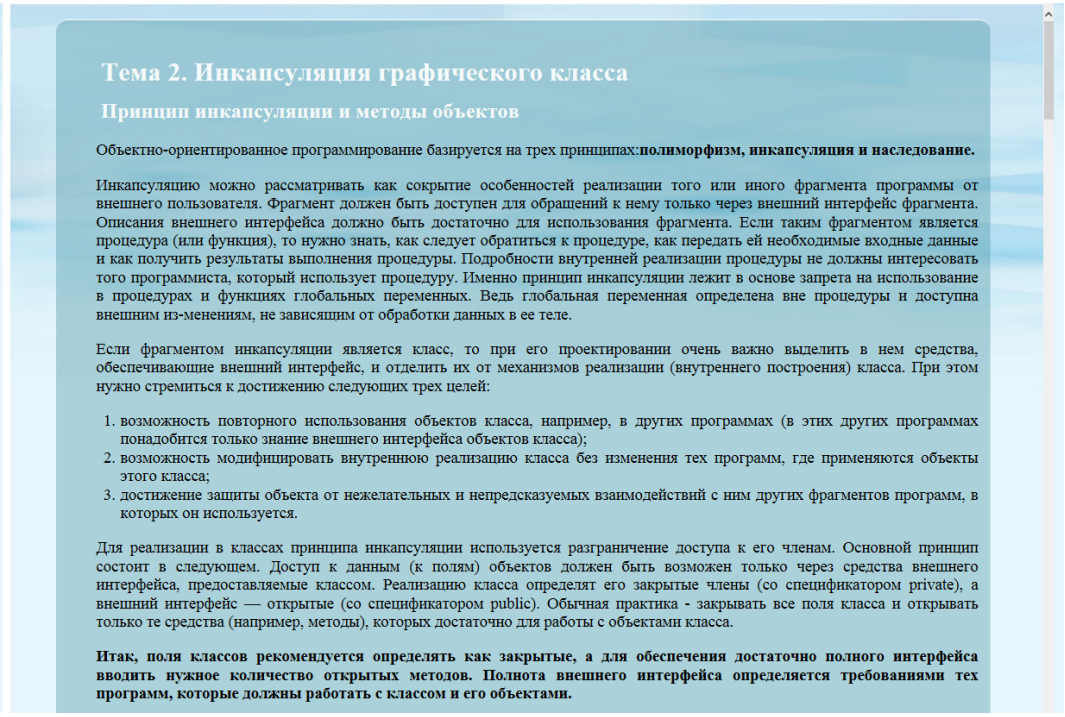


Рисунок 8 — Теория. Раздел 1 Структура классов

В этой теме лекционного материала «Создание графического класса» состоит из двух разделов.

В разделе «Принцип инкапсуляции и методы объектов» рассказывается о следующих параметров:

- фрагмент внешнего интерфейса;
- описания внешнего интерфейса;
- обращения к процедуре, как передать ей необходимые входные данные и как получить результаты выполнения процедуры.

В разделе «Поля классов» рассказывается о следующих параметрах:

- требования программ, которые должны работать с классом и его объектами;
- определение пары методов доступа и изменения;
- определение свойства .net [10, с. 177].
- расширения внешнего интерфейса класса и его объектов;

В разделе примеров программных кодов отображены программы с классом, объекты которого содержат сведения о людях. Изображено на рисунке 9.

```
class Person // Класс человек
{ // Закрытые поля:
    readonly int год_рождения;
    string фамилия;
    public Person(string name, int year) // Конструктор
    { фамилия = name; год_рождения = year; }
    public string getName()  //Аксессор
    { return фамилия; }
    public void setName(string name) // Мутатор
    { фамилия = name; }
    public int getAge() //Аксессор
    { return год_рождения; }
}
```

Рисунок 9 — Программный код класса «Person»

Также здесь есть конструктор общего вида, позволяющий при создании объекта указать фамилию и год рождения человека. Программный код фрагмента класса изображен на рисунке 10.

```
class Person // Класс человек
{ // Закрытые поля:
    readonly int год_рождения;
    string фамилия;
    public Person(string name, int year) // Конструктор
    { фамилия = name; год_рождения = year; }
    public string getName()  //Аксессор
    { return фамилия; }
    public void setName(string name) // Мутатор
    { фамилия = name; }
    public int getAge() //Аксессор
    { return год_рождения; }
}
```

Рисунок 10 — Программный код файла класса «Человек»

В этом разделе есть информация о свойстве инкапсуляции. Пример описания класса «Person» с закрытыми полями изображен на рисунке 11. А также результатом работы программы, который изображен на рисунке 12.

```
class Program
{
    static void Main(string[] args)
    {
            Person one = new Person("Кулик", 1976);
            Console.WriteLine("Фамилия: {0}, ГОД рождения: {1}", one.getName(), one.getAge());
            Console.Write("Введите новую фамилию: ");
            string name = Console.ReadLine(); one.setName(name);
            Console.WriteLine("Фамилия: {0}, год рождения: {1}", one.getName(), one.getAge());
    }
}
```

Рисунок 11 — Программный код класса «Person»

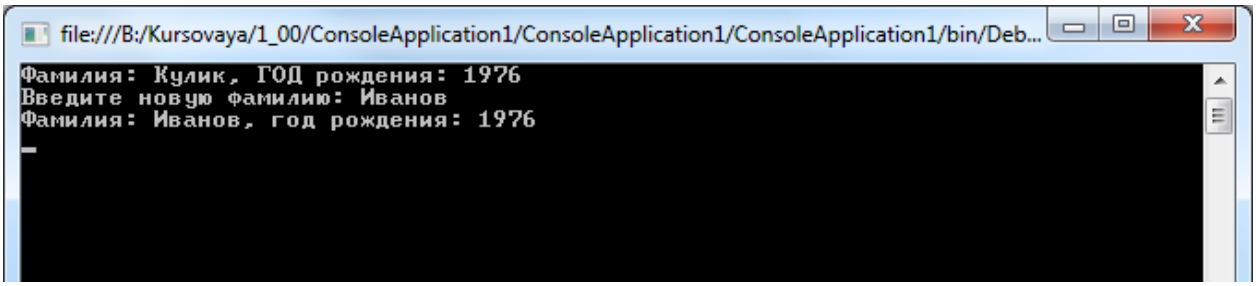


Рисунок 12 — Результат работы программы

В конце лекции «Тема 2. Инкапсуляция графического класса» присутствует ссылка на тестовую форму для самопроверки. Начало тестовой формы отображено на рисунке 13.

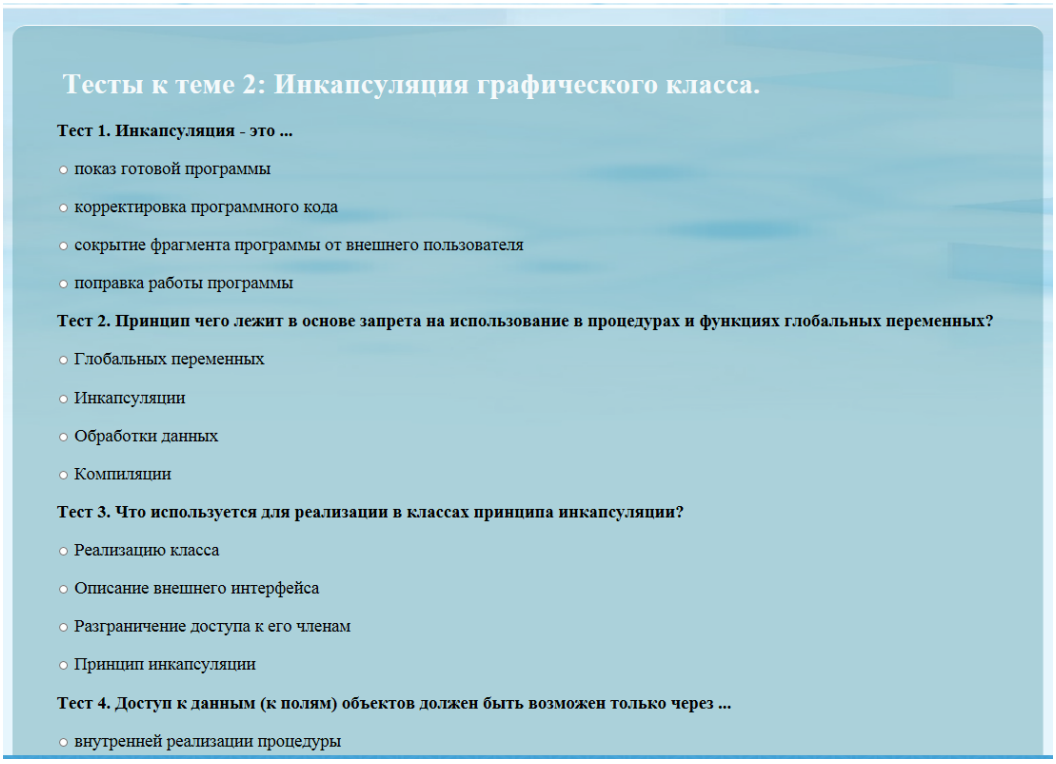


Рисунок 13 — Тестовая форма для лекции «Тема 2. Инкапсуляция графического класса»

Тема «Наследование графического класса»

Начало лекция «Тема 3. Наследование графического класса» изображена на рисунке 14.

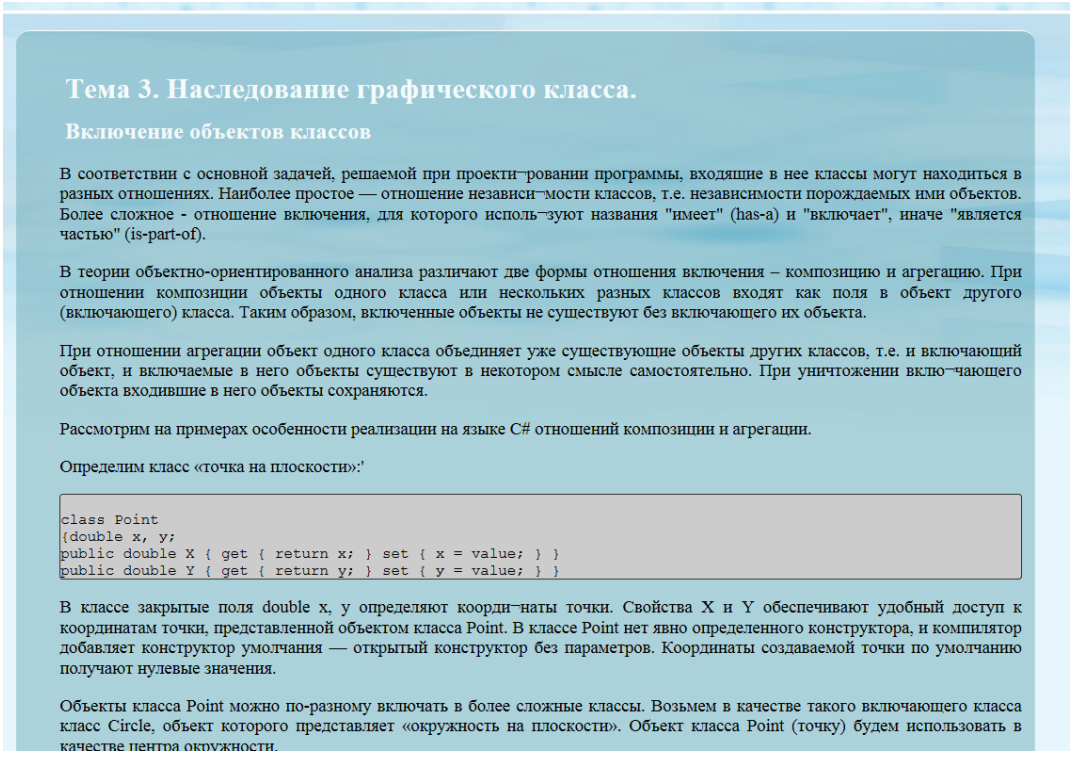


Рисунок 14 — «Тема 3. Наследование графического класса»

В этой теме лекционного материала «Создание графического класса» состоит из раздела «Включение объектов классов», в которую включены данные.

- наследование;
- базовый класс и производный класс;
- поддержка наследования в C#;
- настройка отношения между классами;
- указание базового класса C#;
- переменные экземпляра, методы, свойства и индексаторы;
- основное преимущество наследования.

Программный код метода «Main» и результат работы продемонстрированы на рисунке 15.

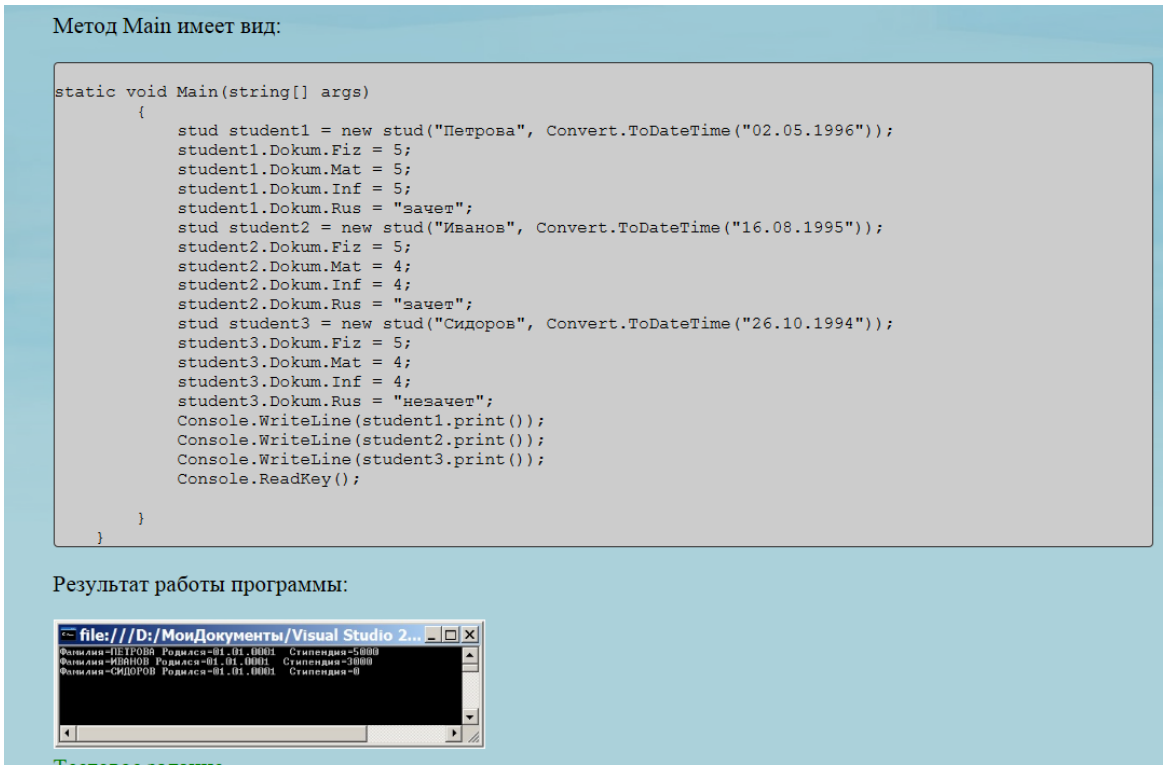


Рисунок 15 — Метод «Main»

В конце лекции «Тема 3. Наследование графического класса» присутствует ссылка на тестовую форму для самопроверки. Начало тестовой формы отображено на рисунке 16.

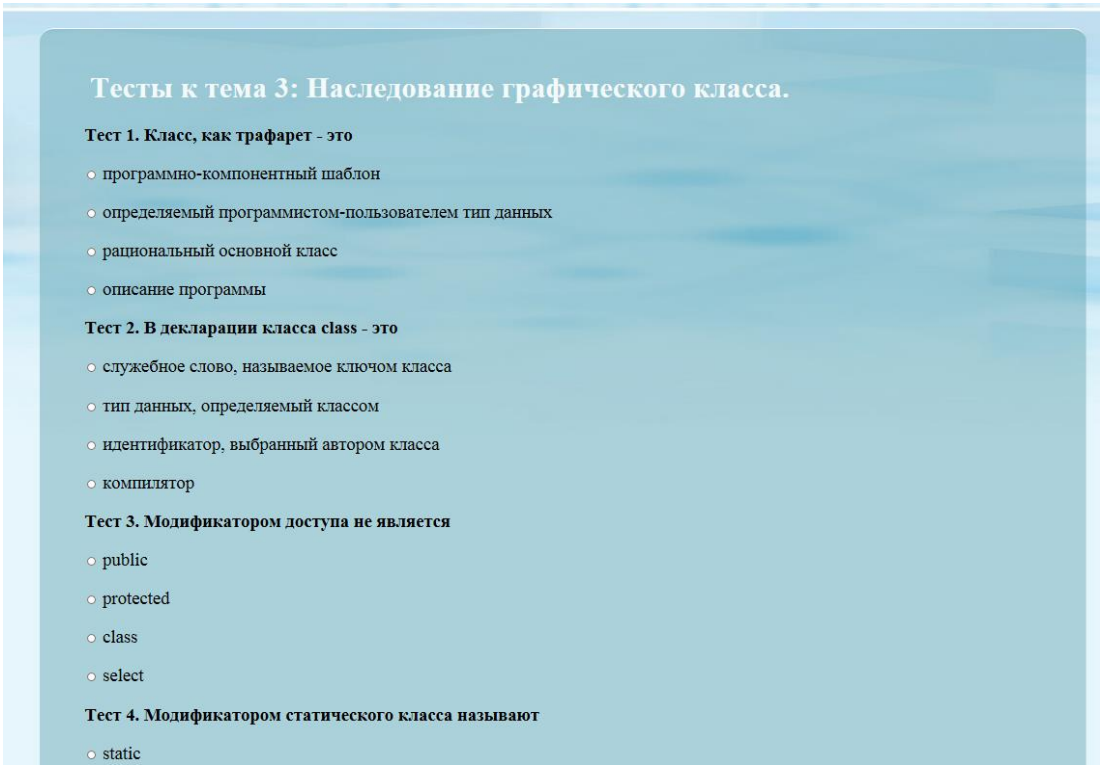


Рисунок 16 — Тестовая форма для лекции «Тема 3. Наследование графического класса»

Тема «Полиморфизм графики»

Начало лекция «Тема 4. Полиморфизм графики» изображена на рисунке 17.

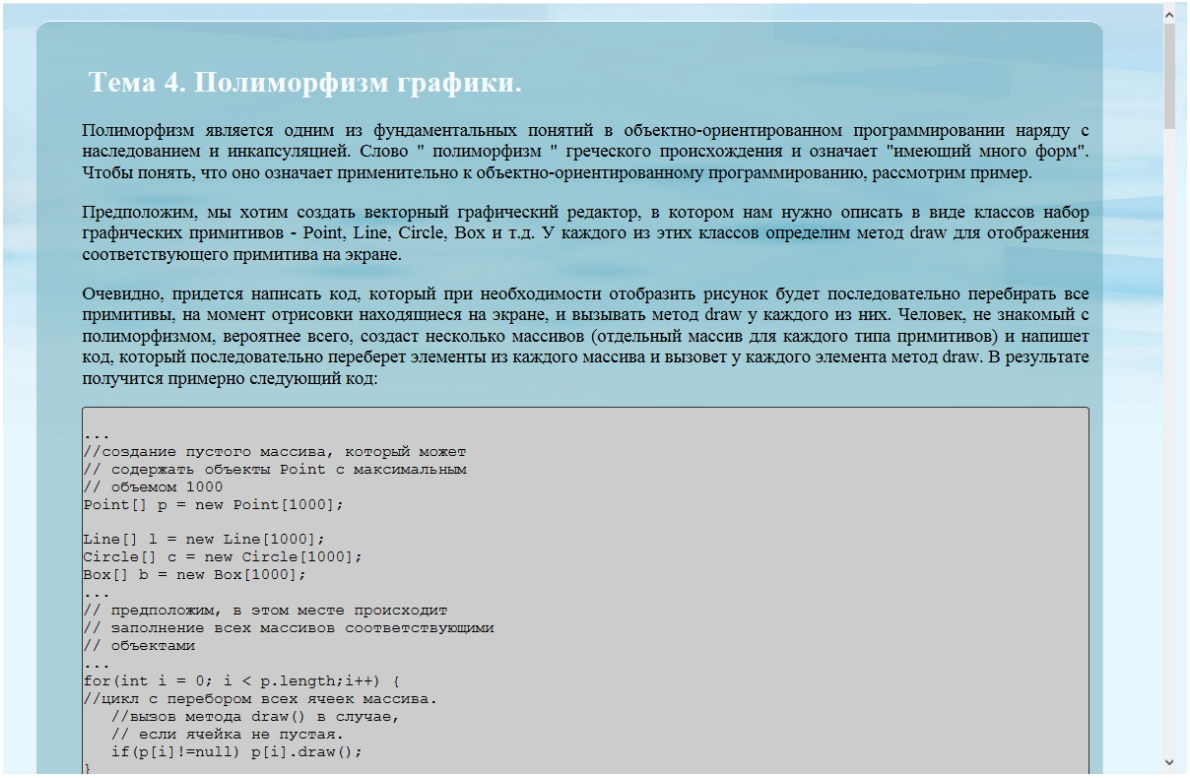


Рисунок 14 — «Тема 4. Полиморфизм графики»

В этой теме лекционного материала «Полиморфизм графики» состоит из определений:

- полиморфизм, как фундаментальное понятие в объектно-ориентированном программировании;
- полиморфизм, как графический объект в объектно-ориентированном программировании;
- параметрический полиморфизм, который подразумевает выполнение одного и того же кода;
- модификатор override [16, с. 143];
- метод, как индексатор, событием и свойством.

Ссылочные массивы с типом базового класса демонстрируют возможности виртуальных методов и свойств.

Также приведены примеры с добавлением методов virtual info() к описанию классов «Person», «Study» и «Prep», обсуждавшихся в предыдущей лаборатории. Программный код класса «Person» изображен на рисунке 15.

```
public class person
    {
        protected string fam;
        protected DateTime rogd;
        protected char pol;
        protected int vozr;
        void let()
        {
            double m;
            DateTime d = DateTime.Now;
            TimeSpan p = d - this.rogd;
            m = p.Days / 365.25;
            vozr= Convert.ToInt32(m);
        }
        public person()
        { fam="Имя";
          rogd=DateTime.Now;
          pol='0';
          vozr=0;
         }
        public person(string fam, DateTime rogd, char pol)
        { this.fam=fam.ToUpper();
            this.rogd=rogd;
            if (pol == 'м' | pol == 'М' | pol == 'ж' | pol == 'Ж')
                this.pol = pol;
            else this.pol = '0';
            let();
        }
        public string print()
        {
            return "Фамилия="+fam+" Родился="+rogd.ToString("dd.MM.yyyy")+" Пол="+pol.ToString();
        }
        public virtual string info() // виртуальный метод
        {
            return "Фамилия=" + fam + " Родился=" + rogd.ToString("dd.MM.yyyy");
        }
       . . .
  }
```

Рисунок 15 — Программный код класса «Person»

Информация о создании производного класса на примере программного кода класса «Stud», который является производным классом «Person», изображен на рисунке 16.

```
class stud : person
{
    string grup;
    string vid;

    public stud():base()
    {   grup = "Группа";
        vid = "";
     }

    public stud(string fam, DateTime rogd, char pol,string grup, string vid)
        :base( fam, rogd, pol)
    {
        this.grup = grup;
        this.vid = vid;
    }
    public new string print() // экранирующий метод
    { return base.print() + " Группа: " + grup + " Вид обучения - " + vid; }
    public override string info() //переопределяемый метод
    { return "Фамилия="+fam+" студент "+ " Группа: " + grup; }
    . . .
}
```

Рисунок 16 — Программный код производного класса «Stud»

Также в тексте есть пример на класс «Человек», в котором изображено сначала присвоение новых классов, а также создания производных открытых классов человек и персонала. Программный код класса «Prep», который является производным классом «Person», изображен на рисунке 17.

```
class prep : person
{
    string kaf;
    string dolgn;
    public prep()
        : base()
    {
        kaf = "";
        dolgn = "должность";
    }

    public prep(string fam, DateTime rogd, char pol, string kaf, string dolgn)
        : base(fam, rogd, pol)
    {
        this.kaf = kaf;
        this.dolgn = dolgn;

    }
    public new string print()// экранирующий метод
    { return base.print() + " Кафедра: " + kaf + " Должность - " + dolgn; }
    public override string info()//переопределяемый метод
    { return "Фамилия=" + fam + " преподаватель " + " Кафедра: " + kaf; }
   . . .
}
```

Рисунок 17 — Программный код производного класса «Prep»

Программный код класса «Programa», который производит алгоритм действий программы, изображен на рисунке 18 [28].

```
class Program
{
    static void Main(string[] args)
    {
        List<person> spis = new List<person>();
        stud spis1 = new stud("Петрова", Convert.ToDateTime("02.05.2000"), 'ж', "ПС-201", "очное");
        spis.Add(spis1);
        prep spis2 = new prep("Иванов", Convert.ToDateTime("09.05.1994"), 'ж', "Психологии", "доцент");
        spis.Add(spis2);
        stud spis3 = new stud("Сидоров", Convert.ToDateTime("26.09.1970"), 'м', "ЗКТ-101", "заочное");
        spis.Add(spis3); foreach (var t in spis) Console.WriteLine(t.print());
        Console.ReadKey(); foreach (var t in spis) Console.WriteLine(t.info());
        Console.ReadKey();
    }
}
```

Рисунок 18 — Программный код производного класса Program

Результат работы программы представлен на рисунке 19.

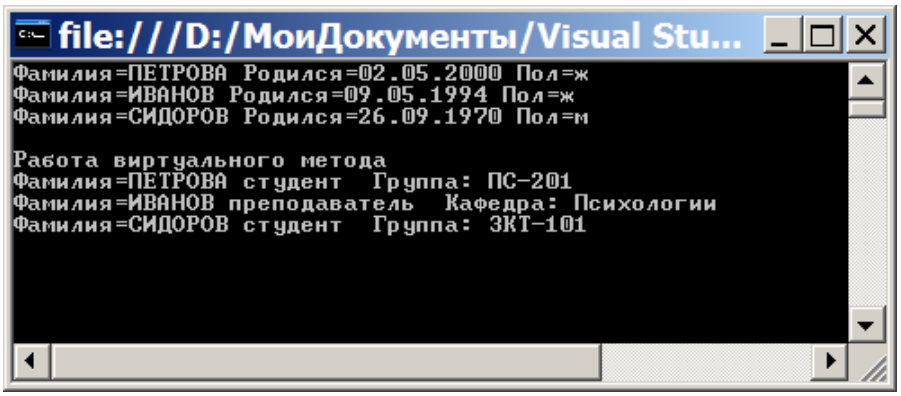


Рисунок 19 — Результат работы программы

В конце лекции «Тема 4. Полиморфизм графики» присутствует ссылка на тестовую форму для самопроверки. Начало тестовой формы отображено на рисунке 20.

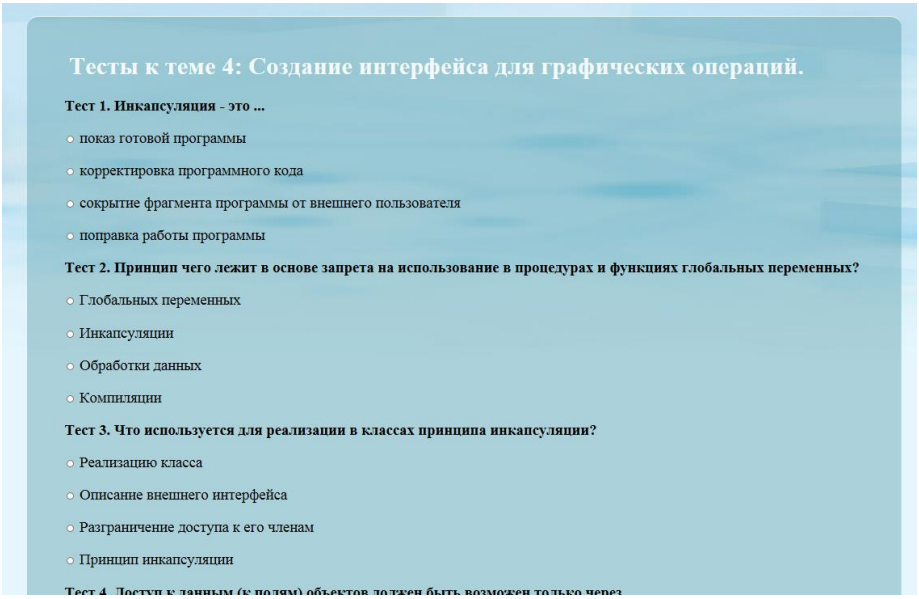
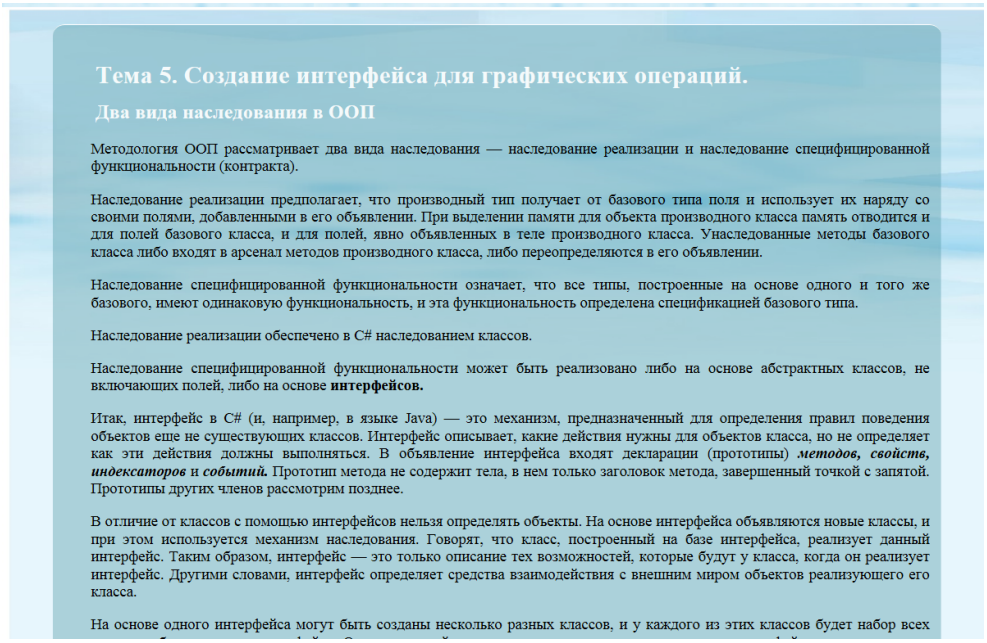


Рисунок 20 — Тестовая форма для лекции «Тема 4. Полиморфизм графики»

Тема «Создание интерфейса для графических операций»

Начало лекции «Тема 5. Создание интерфейса для графических операций» изображено на рисунке 21.



Тема 5. Создание интерфейса для графических операций.

Два вида наследования в ООП

Методология ООП рассматривает два вида наследования — наследование реализации и наследование специфицированной функциональности (контракта).

Наследование реализации предполагает, что производный тип получает от базового типа поля и использует их наряду со своими полями, добавленными в его объявлении. При выделении памяти для объекта производного класса память отводится и для полей базового класса, и для полей, явно объявленных в теле производного класса. Унаследованные методы базового класса либо входят в арсенал методов производного класса, либо переопределяются в его объявлении.

Наследование специфицированной функциональности означает, что все типы, построенные на основе одного и того же базового, имеют одинаковую функциональность, и эта функциональность определена спецификацией базового типа.

Наследование реализации обеспечено в C# наследованием классов.

Наследование специфицированной функциональности может быть реализовано либо на основе абстрактных классов, не включающих полей, либо на основе **интерфейсов.**

Итак, интерфейс в C# (и, например, в языке Java) — это механизм, предназначенный для определения правил поведения объектов еще не существующих классов. Интерфейс описывает, какие действия нужны для объектов класса, но не определяет как эти действия должны выполняться. В объявление интерфейса входят декларации (прототипы) ***методов, свойств, индексаторов*** и ***событий.*** Прототип метода не содержит тела, в нем только заголовок метода, завершенный точкой с запятой. Прототипы других членов рассмотрим позднее.

В отличие от классов с помощью интерфейсов нельзя определять объекты. На основе интерфейса объявляются новые классы, и при этом используется механизм наследования. Говорят, что класс, построенный на базе интерфейса, реализует данный интерфейс. Таким образом, интерфейс — это только описание тех возможностей, которые будут у класса, когда он реализует интерфейс. Другими словами, интерфейс определяет средства взаимодействия с внешним миром объектов реализующего его класса.

На основе одного интерфейса могут быть созданы несколько разных классов, и у каждого из этих классов будет набор всех

Рисунок 21 — «Тема 5. Создание интерфейса для графических операций»

В этой теме лекционного материала «Создание интерфейса для графических операций» состоит из определений:

- среда CLR с реализацией интерфейса графических устройств (GDI);
- интерфейс GDI+ для отображения графики в формах и элементах управления Windows»
- пространство имен System.Рисунок;
- методы рисования на устройстве отображения;
- классы Rectangle и Point;
- 2D графика — растровую и векторную;
- создание графического объекта;
- графические объект для рисования линий и фигур, отображения текста и изображений и управления ими;
- способы создания графических объектов;
- создание графического объекта;

- объект PaintEventArgs при обработке события Paint формы или элемента управления;
- метод CreateGraphics элемента управления или формы;
- пространство имен System. Paint, включая класс Graphics;
- PaintEventArgs в обработчике событий Paint;
- создания обработчика событий PaintEventHandler для элементов управления или объекта PrintPage для PrintDocument [7, с. 378].

В примере показано, как создавать ссылку на объект Graphics из объекта PaintEventArgs события Paint. Изображен на рисунке 22.

```
Private void Form1_Paint(object sender,
 System.Windows.Forms.PaintEventArgs pe)
{ // Declares the Graphics object and sets it to the Graphics object
  // supplied in the PaintEventArgs.
    Graphics g = pe.Graphics;
    // Insert code to paint the form here.}
```

Рисунок 22 — Программный код создания ссылки на объект

Метод CreateGraphics

Создание объекта Graphics с помощью метода CreateGraphics

Вызовите метод CreateGraphics формы или элемента управления, на котором необходимо отобразить графику, отображен на рисунке 23.

```
Graphics g;
// Sets g to a graphics object representing the drawing surface of the
// control or form g is a member of.
G = this.CreateGraphics();
```

Рисунок 23 — Программный код вызова элемента направления

Рисование фигур и изображений и управление ими.

- создание объекта Graphics его можно использовать для рисования Класс Pen;
- класс Brush a [13, с. 183].

2.3 Описание лабораторного блока

Лабораторный блок состоит из пяти лабораторных работ и раздела «Файлы к лабораторным работам», каждая лабораторная работа относится к отдельной теме дисциплины.

Лабораторные работы представлены в виде примера выполнения индивидуальных заданий. К каждой лабораторной работе представлены 10 индивидуальных заданий для проверки усвоения материала.

Лабораторная работа 1. Создание графического класса

В данной лабораторной работе описывается создание графического класса Дом в отдельном модуле. Этот класс имеет поля x, y – отвечающие за местоположение рисунка Дом, поле w – ширина рисунка Дом и h – высота рисунка Дом.

Для создания класса Дом надо щелкнуть правой кнопкой по значку проекта выполнить команду «Добавить» — «Class» на навигационной панели или вписать текст вручную.

Создание имени класса «Dom1» изображен на рисунке 24.

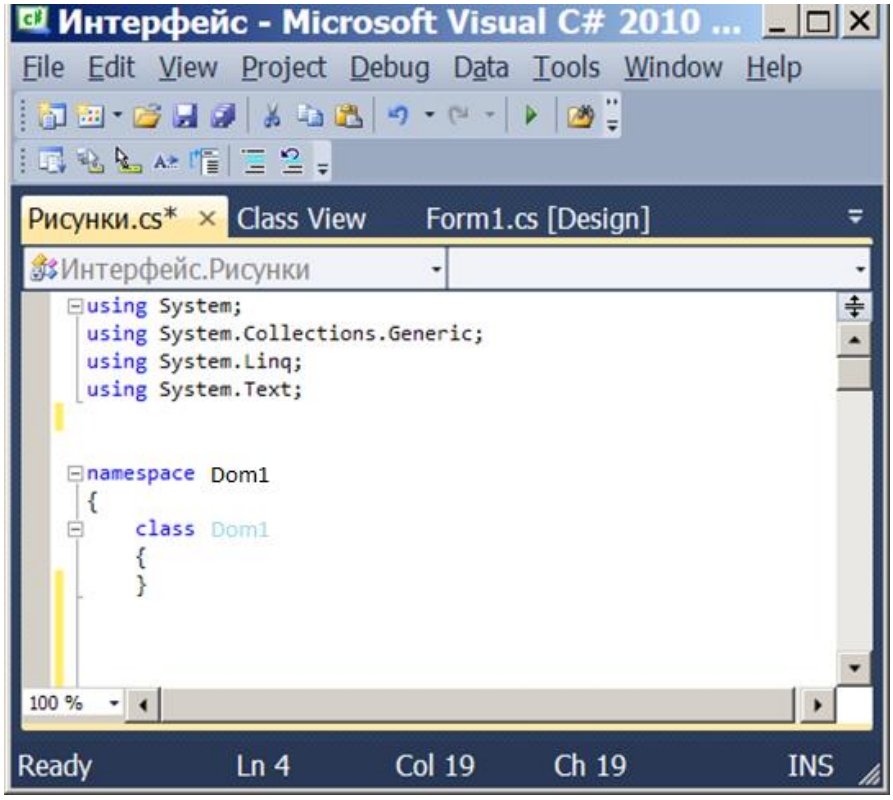


Рисунок 24 — Указание класса

В этой лабораторной работе рассказывается о добавлении строки using System.Drawing; и описание класса «Dom1» (рисунок 25).

```
class Яхта
    {
      int x, y, w, h;
      public Дом()
       {
         x = 0; y = 0;
         w = 100; h = 100;
       }
       public Дом(int x, int y, int w, int h)
        {
         this.x = x;
         this.y = y;
         this.w = w;
         this.h = h;
        }
        public void Draw(Graphics g)
        {
          Pen Перо = new Pen(Color.Black);
          Brush Кисть = new SolidBrush(Color.Yellow);
          g.DrawRectangle(Перо, x, y, w, h);
          Перо.Color = Color.Blue;
          g.DrawRectangle(Перо, x, y + (int)(0.3 * h), w, h - (int)(0.3 * h));
          g.DrawLine(Перо, x, y + (int)(0.3 * h), x + (int)(0.5 * w), y);
          g.DrawLine(Перо, x + (int)(0.5 * w), y, x+w, y + (int)(0.3 * h));
          g.FillRectangle(Кисть, x + (int)(0.1 * w), y + (int)(0.5 * h), (int)(0.4 * w), (int)(0.3 *
h));
          g.FillRectangle(new SolidBrush(Color.Blue), x + (int)(0.6 * w), y + (int)(0.5 * h), (int)(0.3
* w), (int)(0.5 * h));
            Перо.Dispose();
            Кисть.Dispose();

        }
    }
```

Рисунок 25 — Программный код «Dom1»

После в этой лабораторной работе рассказывается о информации проекте эти классы вызываются, как это отображено на рисунке 27.

```
public partial class Form1 : Form
{
    яхта яхта 1; Graphics ris; лодка дерев1; public Form1()
    {
        InitializeComponent(); яхта 1 = new яхта(20, 20, 180, 180);
        дерев1 = new лодка(200, 20, 280, 180); ris = this.CreateGraphics();
    }
    private void Form1_Paint(object sender, PaintEventArgs e)
    { яхта 1.Drow(ris); дерев1.Drow(ris); }
}
```

Рисунок 27 — Программный код процедуры вызова классов

Также в этой лабораторной работе рассказывается об использовании интерфейсов. В модуле «Рисунки» создадим интерфейс Iris и реализуем этот интерфейс в классах «Яхта» и «Лодка», это отображено на рисунке 28.

```
interface Iris
{ void Draw(Graphics g); void Clear(Graphics g, Color c); }
class яхта : Iris
{ }
class лодка : Iris
{ }
```

Рисунок 28 — Программный код интерфейса

В котором отображен список, содержащий объекты классов яхта и лодка. Программный код изображен на рисунках 29, 30, 31.

```
namespace Интерфейс
{
    public partial class Form1 : Form
    {
        List<Iris> ris=new List<Iris>();
         Дом дом;
         Дерево дерево;
         Graphics g;

         public Form1()
         {
             InitializeComponent();
             g=panel1.CreateGraphics();

         }
```

Рисунок 29 — Программный код интерфейса

```
private void button1_Click(object sender, EventArgs e)
{
    if (dom.Checked)
    {
        дом = new Дом((int)X.Value, (int)Y.Value, (int)W.Value, (int)H.Value);
        ris.Add(дом);
    }
    if (derevo.Checked)
    {
        дерево = new Дерево((int)X.Value, (int)Y.Value, (int)W.Value, (int)H.Value);
        ris.Add(дерево);
    }

    panel1.Refresh();
    N.Maximum = ris.Count;

}
```

Рисунок 30 — Программный код кнопки

```
    private void panel1_Paint(object sender, PaintEventArgs e)
    {
        if (ris.Count>0)
        {
            foreach (Iris t in ris)
                t.Draw(g);
        }
    }


    private void button2_Click(object sender, EventArgs e)
    {
        if (N.Value > 0)
        {
            ris.RemoveAt((int)N.Value - 1);
            N.Maximum = ris.Count;
            panel1.Refresh();

        }
    }
   }
}
```

Рисунок 31 — Программный код панели изображения и кнопки

Форма для этой задачи выглядит следующим образом (рисунок 31).

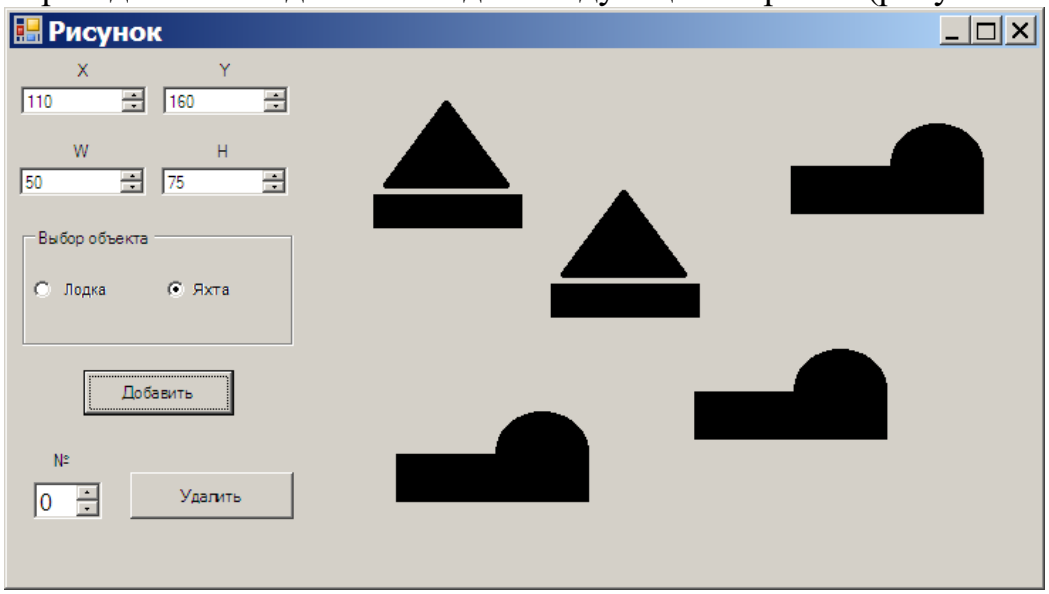


Рисунок 32 — Результат выполнения программы

Лабораторная работа 2. Инкапсуляция.

В данной лабораторной работе описывается о:

1. Классе «товар» сделать все поля закрытыми. Добавлении свойства для доступа к закрытым полям. При задании полей проверять, чтобы цена продажи была не меньше цены закупки, а количество проданного товара не превышало количества закупленного товара.

2. Создании два объекта данного класса. Один — с помощью конструктора без параметра, второй — с помощью конструктора с параметрами. Задать значения для полей этого класса.

Лабораторная работа 3. Наследование.

В данной лабораторной работе описывается о:

1. Описании класса Поставщик, содержащий поля название товара, название поставщика, цена закупки товара, количество закупленного товара, стоимость товара, и класс Товар, содержащий класс Поставщик и поля: ID товара, цена продажи, количество проданного товара, выручка, прибыль.

2. Наследованием классов.

3. Описании классов преподаватель и студент.

У этих классов имеются общие поля, методы и свойства. Конечно, в этой ситуации надо создавать базовый класс, объединяющий общие поля и методы. В результате получим следующий код:

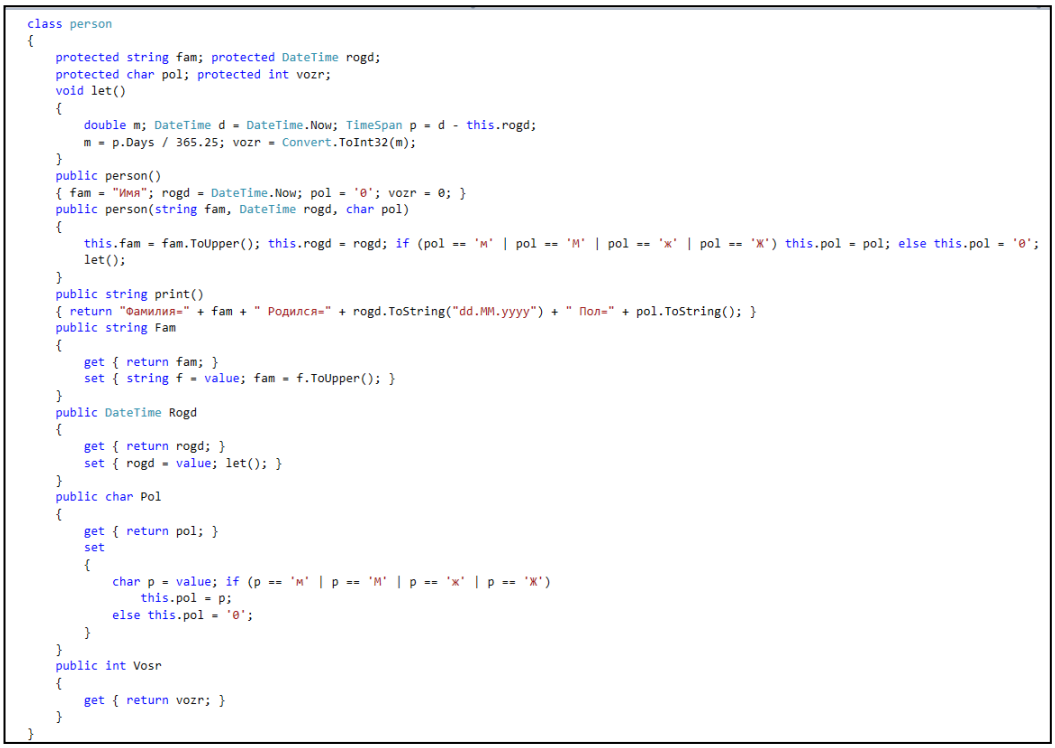
В этой лабораторной работе рассматриваются примеры программного кода базового класса person, изображен на рисунке 33. В этом примере отображено создание глобальных классов с созданием производных классов от класса «Person». В этом классе рассматриваются поля: фамилия, имя, отчество, дата рождения, пол. Эти поля имеют разные форматы переменных.

Fam имеет формат текстовых переменных.

Pol имеет формат char.

Rogd имеет формат даты.

Vozr имеет формат числовых переменных.



```
class person
{
    protected string fam; protected DateTime rogd;
    protected char pol; protected int vozr;
    void let()
    {
        double m; DateTime d = DateTime.Now; TimeSpan p = d - this.rogd;
        m = p.Days / 365.25; vozr = Convert.ToInt32(m);
    }
    public person()
    { fam = "Имя"; rogd = DateTime.Now; pol = '0'; vozr = 0; }
    public person(string fam, DateTime rogd, char pol)
    {
        this.fam = fam.ToUpper(); this.rogd = rogd; if (pol == 'м' | pol == 'М' | pol == 'ж' | pol == 'Ж') this.pol = pol; else this.pol = '0';
        let();
    }
    public string print()
    { return "Фамилия=" + fam + " Родился=" + rogd.ToString("dd.MM.yyyy") + " Пол=" + pol.ToString(); }
    public string Fam
    {
        get { return fam; }
        set { string f = value; fam = f.ToUpper(); }
    }
    public DateTime Rogd
    {
        get { return rogd; }
        set { rogd = value; let(); }
    }
    public char Pol
    {
        get { return pol; }
        set
        {
            char p = value; if (p == 'м' | p == 'М' | p == 'ж' | p == 'Ж')
                this.pol = p;
            else this.pol = '0';
        }
    }
    public int Vosr
    {
        get { return vozr; }
    }
}
```

Рисунок 33 — Программный код класса Person

Также имеется программный код производные классы Stud и Prep изображены на рисунке 34.

```
    class stud : person
    {
        string grup; string vid; public stud() : base() { grup = "Группа"; vid = ""; }
        public stud(string fam, DateTime rogd, char pol, string grup, string vid)
         : base(fam, rogd, pol)
        { this.grup = grup; this.vid = vid; }
        public string Grup
        {
            get { return grup; }
            set { grup = value; }
        }
        public string Vid
        {
            get { return vid; }
            set { vid = value; }
        }
        public new string print()
        { return base.print() + " Группа: " + grup + " Вид обучения - " + vid; }
    }
    class prep : person
    {
        string kaf; string dolgn; public prep() : base()
        { kaf = ""; dolgn = "должность"; }
        public prep(string fam, DateTime rogd, char pol, string kaf, string dolgn)
        : base(fam, rogd, pol) { this.kaf = kaf; this.dolgn = dolgn; }
        public string Kaf
        { get { return kaf; } set { kaf = value; } }
        public string Dolgn
      get { return dolgn; }
    set { dolgn = value; } }
 public new string print()
{ return base.print() + " Кафедра: " + kaf + " Должность - " + dolgn; } } }
```

Рисунок 34 — Программный код классов Stud и Prep

В теме говорится о проекте, где эти классы вызываются, как изображено на рисунке 35 [6, с. 28].

```
class Program
{
    static void Main(string[] args)
    {
        person men = new person("Петрова", Convert.ToDateTime("02.05.2000"), 'ж');
        Console.WriteLine(men.print()); Console.ReadKey();
        stud st = new stud("Иванов", Convert.ToDateTime("09.05.1994"), 'ж', "ЭКТ-303", "заочное"); Console.WriteLine(st.print());
        Console.ReadKey();
        prep pr = new prep("Сидоров", Convert.ToDateTime("26.09.1970"), 'м', "МТ", "доцент"); Console.WriteLine(pr.print());
        Console.ReadKey();
    }
}
```

Рисунок 35 — Программный код вызова программ

Лабораторная работа 4. Полиморфизм.

В данной лабораторной работе описывается:

1. Создание класса Поставщик, содержащий поля: Название поставщика, Название товара, Цена товара. Доступ к цене товара выполняется с помощью индексатора. Индексом является название товара;

2. Программный код класса вызова консоли, который изображен на рисунке 36 [14, с. 131].

```
using System; using System.Collections.Generic;
using System.Linq; using System.Text;
using System.Threading.Tasks; namespace InheritancePolymorphism
    {
        class PersonDemo
        {
            static void Main(string[] args)
            {
                // Объект: Thomas Edison.
                // Создан с помощью Constructor с 2-мя параметрами класса Person.
                Person edison = new Person("Thomas Edison", 1847);
                Console.WriteLine("Info:");
                Console.WriteLine("Name: " + edison.Name);
                Console.WriteLine("Born Year: " + edison.BornYear);
                Console.WriteLine("Place Of Birth: " + edison.PlaceOfBirth);
                // Объект: Bill Gates
                // Создан с помощью Constructor с 3-мя параметрами класса Person.
                Person billGates = new Person("Bill Gate", 1955, "Seattle, Washington");
                Console.WriteLine("-----------------------------------");
                Console.WriteLine("Info:");
                Console.WriteLine("Name: " + billGates.Name);
                Console.WriteLine("Born Year: " + billGates.BornYear);
                Console.WriteLine("Place Of Birth: " + billGates.PlaceOfBirth);
                Console.ReadLine();
            }
        }
```

Рисунок 36 — Программный код класса вызова консоли

Лабораторная работа 5. Интерфейс.

В данной лабораторной работе описывается о создании два класса «Яхта» и «Лодка» в отдельном модуле. Для этого щелкнув правой кнопкой по значку проекта выполняем команду Добавить — Class. Указываем имя класса (рисунок 38).

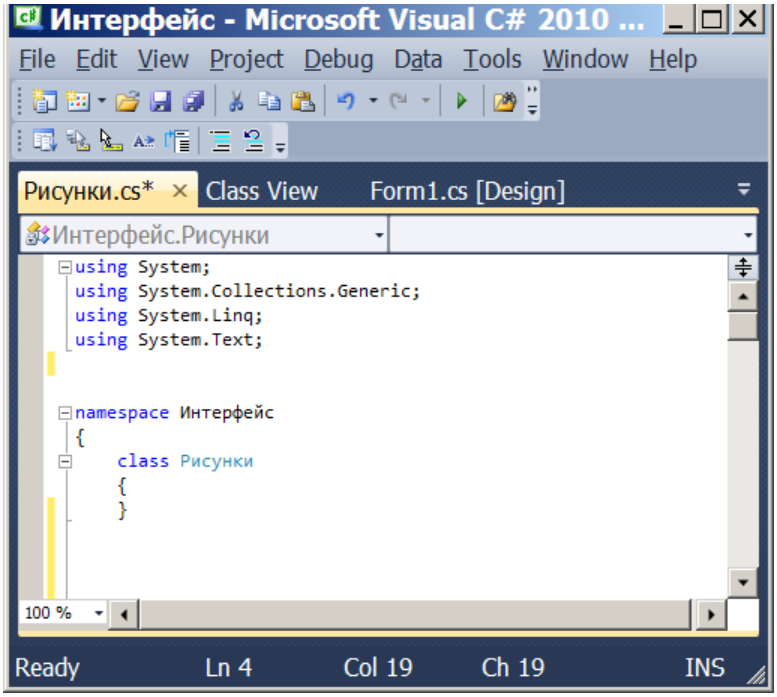


Рисунок 38 — Программный код интерфейса

В лабораторной работе говорится о добавлении строчки using System.Drawing; и описание класса яхта и лодка для создания производственного класса изображены на рисунке 39 [3].

```
class Лодка
{ int x, y, w, h;
    public Лодка ()
    {
        x = 0; y = 0;
        w = 100; h = 100;
    }
    public Лодка (int x, int y, int w, int h)
    {
        this.x = x;
        this.y = y;
        this.w = w;
        this.h = h;

    }
    public void Draw(Graphics g)
    {
        Pen Перо = new Pen(Color.Aquamarine);
        Brush Кисть = new SolidBrush(Color.Green);
        g.FillEllipse(Кисть, x, y, w, (int)(2.0 / 3 * h));
        g.FillRectangle(new SolidBrush(Color.Brown), x + (int)(0.45 * w), y + (int)(2.0 / 3 * h),
(int)(0.1 * w), (int)(1.0 / 3 * h));
        Перо.Dispose();
        Кисть.Dispose();

    }
}
```

Рисунок 39 — Программный код интерфейса класса Лодка

Также отображено обращения вызываемости, как это изображено на рисунке 40.

```
    public partial class Form1 : Form
    {
        яхта яхта 1; Graphics ris; лодкалодка;
public Form1()
        {
            InitializeComponent();
            яхта 1 = new яхта(20, 20, 180, 180);
            лодка1 = new лодка(200, 20, 280, 180);
            ris = this.CreateGraphics();
        }
        private void Form1_Paint(object sender, PaintEventArgs e)
        { яхта 1.Dr}
```

Рисунок 40 — Программный код процедуры вызова

2.4 Описание интерфейса электронного учебного пособия «Объектно-ориентированное программирование графических объектов на языке C#»

Электронное учебное пособие включает пять тем, изучаемых в течение семестра. Программа находится в сети Интернет и запускается с любого компьютера, подключенного к нему.

В ходе создания пособия были использованы: HTML, CSS, JavaScript.

Был создан файл Index.html. В котором находятся ссылки на страницы с теоретическим и практическим материалом.

Начальный лист электронного учебного пособия «Объектно-ориентированное программирование графических объектов на языке C#» приведен на рисунке 41. На начальном тексте расположена информация о назначении электронного учебного пособия и его содержании.

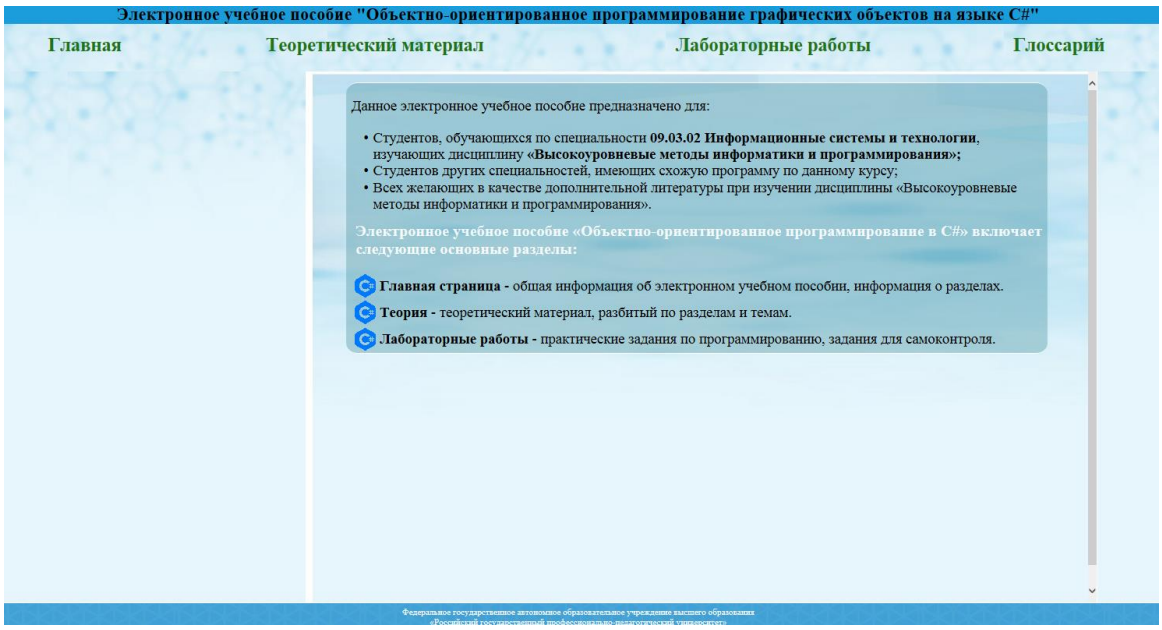


Рисунок 41 — Начальный экран

В правой части формы находится меню. Меню состоит из пунктов: главная, теория, практикум, глоссарий.

Лекция «Тема 1. Создание классов» изображена на рисунке 42.

Рисунок 42 — Теория. Раздел 1 Структура классов

После прочтения лекции в нижней части текста находится ссылка на тестовые задания. Ссылка показана на рисунке 43. Перейдя по этой ссылке,

открывается страница, где можно ответить на десять вопросов для проверки усвоения прочтенного материала по заданной лекции.

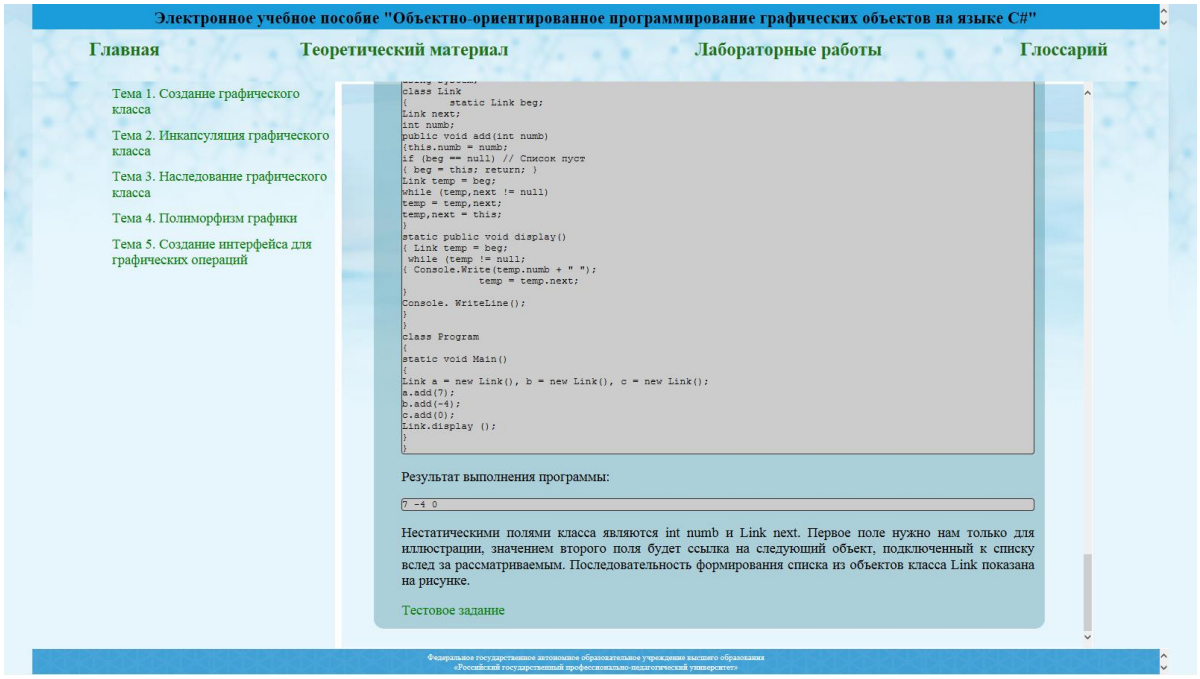


Рисунок 43 — Ссылка на тестовое задание

После прочтения лекции и проверки усвоенного материла предлагается выполнить лабораторные работы. Для перехода к ним нужно нажать пункт меню «Лабораторные работы». На рисунке 43 изображен начальный экран лабораторной работы 1.

Перед выполнением лабораторной работы рекомендуется просмотреть видеоролик по данной работе.

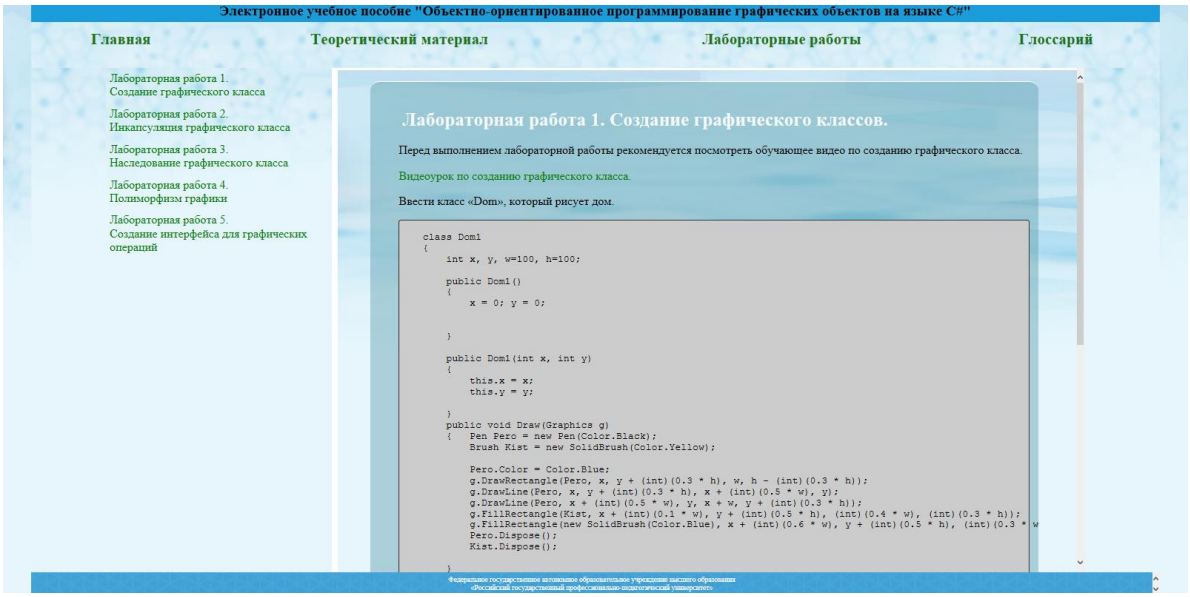


Рисунок 43 — Начальный экран лабораторной работы 1

После перехода по ссылке на видеоролик открывается страница с видео, где рассказывается о пошаговом создании программного кода на языке C#. Страница с видеороликом изображена на рисунке 45.

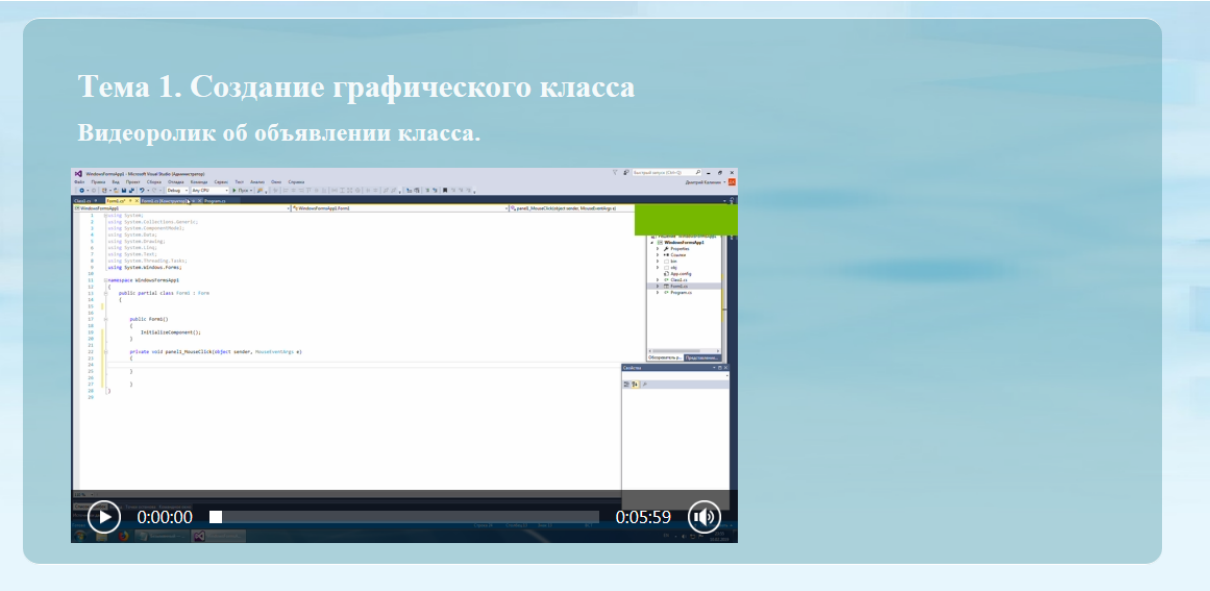


Рисунок 45 — Страница с видеороликом

Откроется страница с десятью вопросами и четырьмя вариантами ответов к каждому, где нужно выбрать только один. Вопросы по теме «Создание классов» отображены на рисунке 46.

Рисунок 46 — Теория. Раздел 1 Структура классов

После выбора ответов нужно нажать на кнопку «Проверить», которая находится под вопросами. После чего выведется сообщение с количеством правильных ответов. Просмотр результатов изображен на рисунке 47.

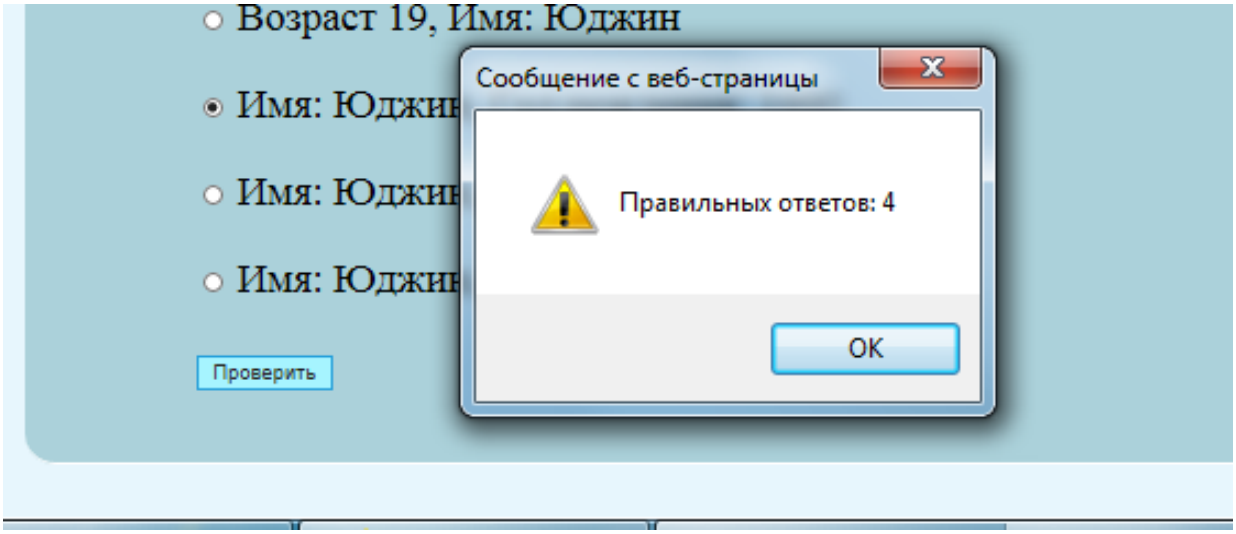


Рисунок 47 — Просмотр результатов

Практическая работа «Лабораторная работа: Создание классов» изображена на рисунке 48.

На странице располагается пример выполнения задания и под примером находится ссылка на десять индивидуальных заданий.

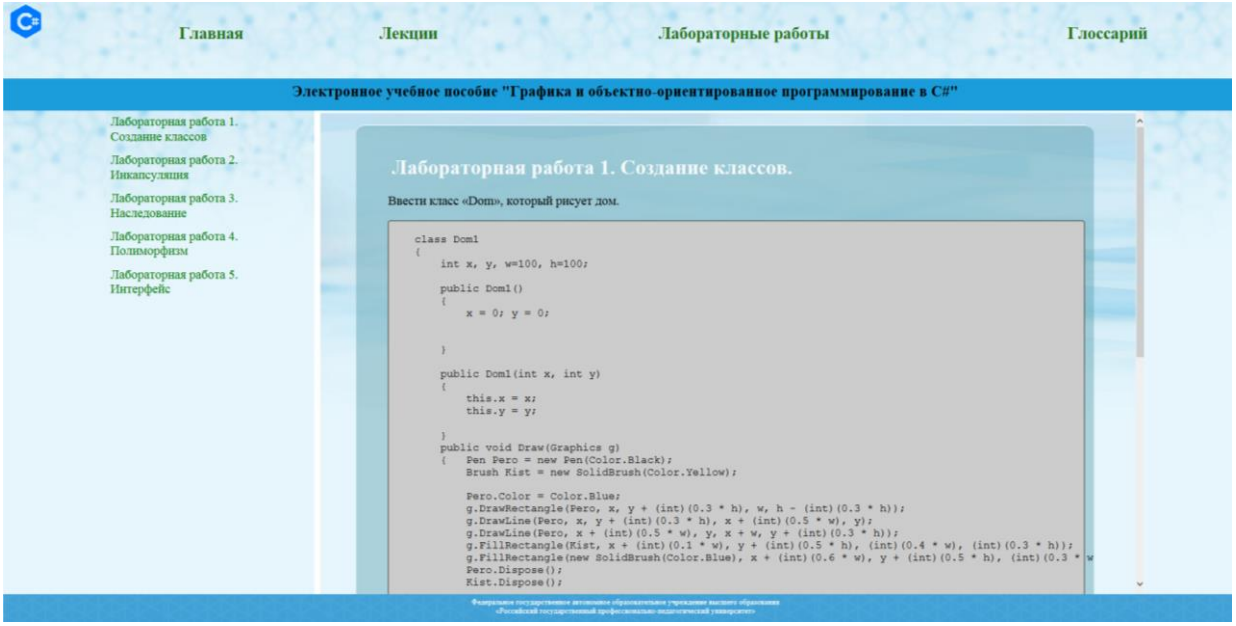


Рисунок 48 — Лабораторная работа 1: Создание классов

После перехода по ссылке «Индивидуальное задание» откроется страница с заданиями. Список индивидуальных заданий изображен на рисунке 49.

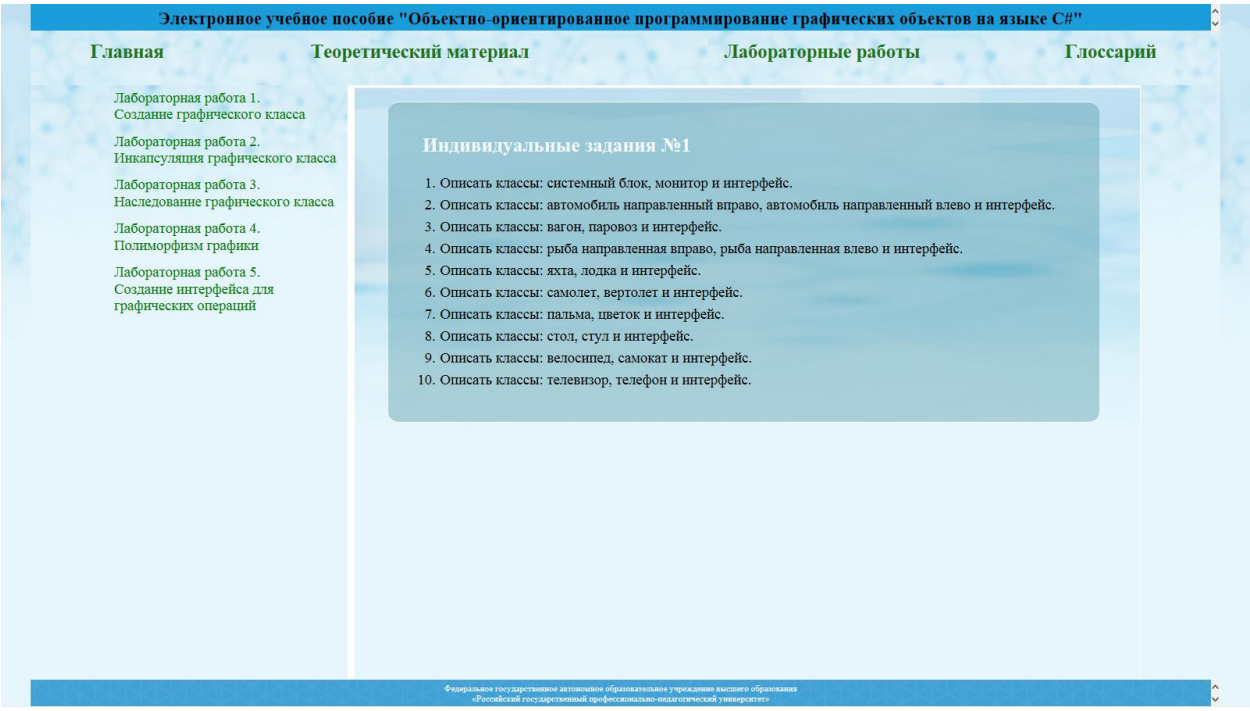


Рисунок 49 — Индивидуальные задания

Пункт меню «Глоссарий» изображен на рисунке 50.

Рисунок 50 — Глоссарий

2.5 Методические рекомендации

Для работы с данным электронным пособием в аудитории необходим персональный компьютер, данное пособие может быть использовано для самостоятельной работы обучающегося.

В первую очередь обучающемуся нужно изучить теоретический блок, в котором содержится информация по каждой разделу. В разделах содержатся темы уроков.

Электронное учебное пособие состоит из разделов и тем: создание класса, инкапсуляция, наследование, полиморфизм, интерфейс.

Изучив первую тему обучающемуся необходимо перейти к лабораторному блок. После лабораторного блока перейти в практический блок.

Практическая часть электронного учебного пособия включает в себя эмуляцию действий, исходя из изученного материала, т.е. начала изучить теорию по конкретной теме, а только после этого переходить к лабораторной работе [18, с. 14].

Обучающийся может выбрать в каком порядке изучить материал, нужно, чтобы он прошел все темы. Если у обучающегося возникли затруднения при выполнении лабораторных работ, то он всегда может просмотреть выполняемые в заданиях последовательности действий, которые имеются в каждой теоретической части, а также в отчетах к лабораторным работам.

По завершении выполнения лабораторных работ с пошаговыми инструкциями обучающийся должен будет перейти практическим заданиям.

Для контроля полученных знаний обучающийся должен будет предоставить преподавателю:

1. Отчет по проделанной лабораторной работе.
2. Скриншоты выполненной практики.

Разнообразие видов контроля в электронном учебном пособии позволяет преподавателю корректно оценить знания обучающегося по пройденному им материалу.

ЗАКЛЮЧЕНИЕ

В ходе проделанной выпускной квалификционной работы были проанализированы источники информации по теме «Высокоуровневые методы информатики и программирования».

Электронное учебное пособие предназначено для студентов направления подготовки 09.03.02 Информатика и технологии медиаиндустрии профиль подготовки «Информационные технологии в медиаиндустрии», изучающих дисциплину «Высокоуровневые методы информатики и программирования» и для студентов других специальностей, изучающих программирование в C#, а также может быть использовано в учреждениях дополнительного образования, обучающих программированию.

Существует достаточно большое количество электронных практикумов, посвященных объектно-ориентированному программированию на C#, практические работы в этих пособиях рассматривают классы, как абстрактные. В представленном электронном учебном пособии все основные черты объектно-ориентированного программирования рассматриваются на основе создания графических классов.

С помощью языка разметки документов реализован интерфейс и функционал электронного учебного пособия «Объектно-ориентированное программирование графических объектов на языке C#».

Была выполнена работа по разработке электронного учебного пособия «Объектно-ориентированное программирование графических объектов на языке C#», предназначенное для студентов направления подготовки 09.03.02 Информатика и технологии медиаиндустрии профиль подготовки «Информационные технологии в медиаиндустрии». В ходе выполнение этой задачи был проведен анализ рабочей программы. Оно содержит пять теоретических материалов с десятью тестовыми заданиями каждый, а также с пя-

тью лабораторными работами, каждая из которых содержит десять индивидуальных заданий.

В процессе исследования были выполнены задачи:

1. Проанализированы литературные и интернет-источники по теме «Высокоуровневые методы информатики и программирования».
2. Проанализирована рабочая программа.
3. Разработана структура электронного учебного пособия «Объектно-ориентированное программирование графических объектов на языке C#».
4. Реализован интерфейс и функционал электронного учебного пособия «Объектно-ориентированное программирование графических объектов на языке C#».

Таким образом, задачи решены, цель достигнута.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. Алгоритмизация, введение в язык программирования C++ в цикле. Курсы ИНТУИТ [Электронный ресурс]. — Режим доступа: http://intuit.valrkl.ru/ course-1345/index.html/ (дата обращения: 09.09.2018).

2. Буч Г. Объектно-ориентированный анализ и проектирование с примерами приложений [Текст] / Г. Буч, Р. А. Максимчук, М. У. Энгл, и др. — Москва: Вильямс, 2008. — 720 с.

3. Васильев А. Н. C#. Объектно-ориентированное программирование. Учебный курс. Электронные издания [Текст]: учебник / А. Н. Васильев. — Санкт-Петербург: Питер, 2012. — 320 с

4. Гамма Э. Приемы объектно-ориентированного проектирования. Паттерны проектирования [Текст] / Э. Гамма, Р. Хелм, Р. Джонсон, Дж. Влиссидес. — Санкт-Петербург: Питер, 2011. — 366 с.

5. Давыдова Н. А. Программирование [Текст]: учебное. пособие / Н. А. Давыдова, Е. В. Боровская. — Москва: Лаборатория знаний, 2015. — 241 с.

6. Красильников И. В. Информационные аспекты разработки и применения в ВУЗе электронных учебных пособий [Текст]: монография / И. В. Красильников. — Москва: РХТУ, 2011. — 114 с.

7. Лав Р. А. Linux. Системное программирование [Текст] / Р. А. Лав. — 2-е издание. — Санкт-Петербург: Питер, 2014. — 448 с.

8. Лаптев В. В. C++. Объектно-ориентированное программирование задачи и упражнения [Текст]: учебное пособие для вузов / В. В. Лаптев, А. В. Морозов, А. В. Бокова. — Санкт-Петербург: Питер, 2007. — 287 с.

9. Макарова С. О. Совершенствование учебно-методического комплекса «Линия компьютера» с применением гипертекстовых технологий [Электронный ресурс]. — Режим доступа: http://bibliofond.ru/view.aspx?-id=446952 (дата обращения: 10.05.2017).

10. Маклафлин Б. Объектно-ориентированный анализ и проектирование [Текст] / Б. Маклафлин, Г. Поллайс, Д. Уэст. — Санкт-Петербург: Питер, 2013. — 608 с.

11. Основные этапы создания электронного учебного пособия [Электронный ресурс]. — Режим доступа: http://wikikurgan.orbitel.ru/images/0/04/Etap.doc (дата обращения: 22.11.2018).

12. Официальный сайт разработчиков программного обеспечения Microsoft [Электронный ресурс]. — Режим доступа: https://docs.microsoft.com/ru-ru/dotnet/csharp (дата обращения: 02.12.2018).

13. Павловская Т. А. C#. Программирование на языке высокого уровня [Текст]: учебник для вузов / Т. А. Павловская. — Санкт-Петербург: Питер, 2014. — 432 с.

14. Петров, С. Б. Основы алгоритмизации и программирования [Текст]: учебное пособие для вузов / С. Б. Петров, С. Н. Ширева. — Екатеринбург: РГППУ, 2011. — 160 с.

15. Писаренко Т. А. Основы дизайна [Текст]: учебное пособие / Т. А. Писаренко, Н. Н. Ставнистый. — Владивосток: ДГУ, 2010. — 113 с.

16. Подбельский В. В. Базовый курс; Язык C# [Текст] / В. В. Подбельский. — Москва: Финансы и статистика; ИНФРА-М, 2015. — 382 с.

17. Полный Курс C# Base [Электронный ресурс]. — Режим доступа: https://glamcoder.ru/video/c-sharp-base-video (дата обращения: 10.11.2018).

18. Рабочая программа дисциплины «Программирование в компьютерных системах». Для студентов всех форм обучения направления подготовки 030300.68 Психология, магистерская программа «Организационная психология», «Юридическая психология» [Текст] / Н. С. Нарваткина. — Екатеринбург: РГППУ, 2012. — 16 с.

19. Рабочая программа по программированию в компьютерных системах [Электронный ресурс]. — Режим доступа: https://life-prog.ru (дата обращения: 20.09.2018).

20. Роберт И. В. Информационные и коммуникационные технологии в образовании [Текст]: учебник / И. В. Роберт, С. В. Панюкова, А. А. Кузнецова и др. — Москва: Дрофа, 2009. — 320 с.

21. Русская документация jQuery [Электронный ресурс]. — Режим доступа: http://www. jquery-docs.ru/ (дата обращения: 21.11.2018).

22. Системное программирование «Техностриm» [Электронный ресурс]. — Режим доступа: https://www.youtube.com/channel/UCmq-EpAsQMcsYaeef4qgECvQ (дата обращения: 09.09.2018).

23. Титова С. В. Ресурсы и службы интернета в преподавании иностранных языков [Текст] / С. В. Титова. — Москва: Эдитус, 2007. — 267 с.

24. Троелсен Э. Язык программирования C# 2010 и платформа .NET 4 [Текст]: учебник / Э. Троелсен. — 5-е издание. — Москва: Вильямс, 2015. — 340 с.

25. Шалкина Т. Н. Электронные учебно-методические комплексы: проектирование, дизайн, инструментальные средства [Текст] / Т. Н. Шалкина, В. В. Запорожко, А. А. Рычкова. — Оренбург: ОГУ, 2010. — 160 с.

26. Эрганова Н. Е. Методика профессионального обучения [Текст]: учебное пособие / Н. Е. Эрганова. — Москва: Академия, 2008. — 160 с.

27. Эрганова Н. Е. Практикум по методике профессионального обучения [Текст]: учебное пособие / Н. Е. Эрганова, М. Г. Шалунова, Л. В. Колясникова. — 2-е издание. — Екатеринбург: РГППУ, 2011. — 89 с.

28. Professorweb.ru Net & Web Programming [Электронный ресурс]. — Режим доступа: https://professorweb.ru/my/csharp/charp_theory/level3/3_1.php (дата обращения: 20.09.2018).

29. Ruby On Rails. vadimstroganov.com [Электронный ресурс]. — Режим доступа: https://vadimstroganov.com (дата обращения: 10.09.2018).

30. Tproger [Электронный ресурс]. — Режим доступа: https://tproger.ru/translations/diving-in-oop-p1/ (дата обращения: 14.11.2018).

ПРИЛОЖЕНИЕ