Частное учреждение высшего образования
«Институт государственного администрирования»

.

Кафедра математики и информационных технологий

УТВЕРЖДАЮ
Проректор по учебной работе
_____ П.Н. Рузанов
«29» мая 2025 г.

РАБОЧАЯ ПРОГРАММА УЧЕБНОЙ ДИСЦИПЛИНЫ

Объектно-ориентированное программирование

Направление подготовки
09.03.01 Информатика и вычислительная техника

Направленность
«Искусственный интеллект и машинное обучение»

ПРОГРАММА БАКАЛАВРИАТА

Квалификация
Бакалавр

Форма обучения

Очная

Москва 2025

Рабочая программа учебной дисциплины ***Объектно-ориентированное программирование*** разработана на основании федерального государственного образовательного стандарта высшего образования – бакалавриата по направлению подготовки 09.03.01 Информатика и вычислительная техника (бакалавриат), утвержденного приказом Министерства образования и науки Российской Федерации от 19.09.2017 № 929, учебного плана по основной профессиональной образовательной программе высшего образования – программе бакалавриата по направлению подготовки 09.03.01 Информатика и вычислительная техника (бакалавриат), с учетом следующих профессиональных стандартов, сопряженных с профессиональной деятельностью выпускника:

− 06.001 «Программист»;

− 06.004 «Специалист по тестированию в области ИТ»

− 06.011 «Администратор баз данных»;

− 06.015 «Специалист по информационным системам».

− 06.016 «Руководитель проектов в области информационных технологий»

− 06.019 «Технический писатель (специалист по технической документации в области ИТ)

Рабочая программа учебной дисциплины разработана рабочей группой в составе:

Рабочая программа дисциплины (модуля) обсуждена и утверждена на заседании кафедры математики и информационных технологий.

Протокол №

Заведующий кафедрой _____

(подпись)

Аннотация рабочей программы
по дисциплине «Объектно-ориентированное программирование»

Цель освоения дисциплины состоит в приобретении базовых знаний и навыков программирования, проектирования и разработки приложений с применением объектно-ориентированного подхода, изучение платформы ООП, стандартной библиотеки классов, основ многопоточного и распределенного программирования, безопасности программных систем, использующих технологию ООП.

В процессе изучения дисциплины решаются следующие задачи:

− формирование представлений об общей методологии и средствах технологии объектно-ориентированного программирования;

− углубленная подготовка студентов в области применения технологии объектно-ориентированного программирования.

В ходе изучения дисциплины у обучающегося формируются следующие компетенции:

№ п-п	Содержание формируемых компетенций	Индекс компетенции
Профессиональные (ПК)		
2	ПК-6 Способен разрабатывать стратегии тестированияи управление процессом тестирования, разрабатывать документы для тестирования и анализировать качествопокрытия	ПК-6

ПК-6 Способен разрабатывать стратегии тестированияи управление процессом тестирования, разрабатывать документы для тестирования и анализировать качествопокрытия	Знает: ПК-6.1: освоение основных методов разработки стратегии тестирования и управления процессом тестирования, разработки документов для тестирования и анализа качества покрытия Умеет ПК-6.2: навык самостоятельно разрабатывать стратегии тестирования и управление процессом тестирования, разрабатывать документы для тестирования и анализировать качество покрытия. Владеет ПК-6.3: владение принципами и методамиразработки стратегии тестирования и управления процессом тестирования, разработки документов для тестирования и анализа качества покрытия

1. Перечень планируемых результатов обучения по дисциплине (модулю), соотнесенных с планируемыми результатами освоения образовательной программы

1.1. Цель преподавания дисциплины

Сформировать знания о принципах объектно-ориентированного подхода и, в частности, объектно-ориентированное программирование (ООП) являются ключевой теорией и практикой при конструировании современного программного обеспечения. Целью курса является развитие уверенных знаний и практик у обучающихся в использовании главных идей объектного подхода. Приобретение обучающимися знаний об объектно-ориентированном подходе в программировании, освоение возможностей языков ООП с концентрацией на решении объектно-ориентированных проблем.

1.2. Задачи изучения

− Владеть терминологическим базисом объектных методов.

− Уметь применять на практике методы объектной декомпозиции.

− Формирование у обучающихся объектно-ориентированного мышления;

− Формирование навыков использования объектно-ориентированной методологии программирования при разработке программных продуктов;

− Изучение техники объектно-ориентированного анализа;

− Владеть методами объектно-ориентированного программирования.

1.3. Компетенции обучающегося, формируемые в результате освоения дисциплины (модуля)

Категория компетенции	Код и наименование компетенции	Индикаторы достижения компетенции
Профессиональные компетенции (ПК)	ПК-6. Способен разрабатывать требования и проектировать программноеобеспечение	***знать:*** основные парадигмы объектных методов; приёмы и синтаксис для создания и использования классов и объектов; основные принципы и шаблоны распределения обязанностей между объектами, нотацию UML в аспекте моделирования классов и связей; ***уметь:*** записывать простейшие алгоритмы на алгоритмическом языке программирования; создавать простейшие программы (объекты, классы) в среде C++, иллюстрирующие основы технологии объектно-ориентированного программирования (компонентного проектирования программ), проектировать и реализовывать простые прикладные приложения на языке C#; ***владеть:*** навыками работы в визуальной среде программирования C++; навыками структурной алгоритмизации и технологии структурного программирования, синтаксисом языка C# для создания объектно-ориентированных приложений и компонентов; читать и составлять модели классов для различных предметных областей; применять шаблоны Grasp при проектировании классов;

2. Место дисциплины в структуре образовательной программы

2.1. Перечень дисциплин, освоение которых студентами необходимо для изучения данной дисциплины

- Алгоритмы и структуры данных;

- Технологии программирования.

2.2. Перечень дисциплин, изучение которых базируется на материале данной дисциплины

- Методы и средства проектирования информационных систем и технологий;
- Производственная (практика по получению профессиональных умений и опыта профессиональной деятельности);
- Государственная итоговая аттестация.

3. Структура и содержание дисциплины:

Общая трудоемкость дисциплины составляет 8 зачетных единиц, 288 часов.

3.1. Объем дисциплины и виды учебной работы

Семестр	Всего часов	Итого контактные часы	В том числе					СРС	Контроль	КП, КР, РГР, контр. раб, реферат	Экзамен	Зачет
			Лек	Лаб	Пр	ИЗ	АК					
6	108	54	12	12	30			54	-	РГР	-	+
7	108	54	12	12	30			54		РГР	-	+
ИТОГО	**216**	**108**	**24**	**24**	**60**			**108**		**2РГР**	**-**	**+**

3.1.2. Наименование тем, их содержание, объем в часах лекционных занятий (по семестрам)

№ темы	Наименование темы	Основное содержание темы	Количество часов
6 семестр			
1	1. История языка программирования C++. Структура программы на C++. Необъектно-ориентированные расширения C++. Обзор среды программирования.	Работы Б. Страуструпа по созданию C. Простая программа на языке C («Hello, World!») с использованием потокового ввода-вывода. Перегрузка имён функций, параметры функций по умолчанию, определение типов (typedef), обработка исключений (try-catch). Создание проектов, компиляция и отладка программ в Visual Studio.	3
2	2. Классы и объекты в C++. Специальные функции класса.	Понятие класса и объекта. Синтаксис определения класса: члены-данные и члены-функции (методы) класса. Спецификаторы доступа: private, protected, public. Создание экземпляров (объектов) класса: статическое (на стеке) и динамическое размещение объектов. Конструктор класса. Виды конструкторов: по умолчанию, копирования, инициализации. Правила определения конструкторов. Деструктор класса. Правила определения деструкторов.	3

3	3. Перегрузка операторов в классах. Приведение типов.	Необходимость в перегрузке операторов класса. Унарные и бинарные операторы. Перегрузка логических и арифметических операторов. Оператор приведения типа.	3
4	4. Наследование в языке C++. Полиморфизм наследования в языке. Множественное наследование в. Параметрический полиморфизм C++.	Понятие наследования. Одиночное наследование. Переопределение методов. Виртуальные методы. Абстрактные методы и классы. Интерфейсы. Понятие множественного наследования. Проблемы множественного наследования. Виртуальное наследование классов. Понятие параметризованных классов (template). Инстанцирование параметризованных классов. Примеры применения.	3
		Итого	12
		7 семестр	
1	Сфера применения, преимущества и недостатки объектных методов. Объект как фундаментальное понятие объектно-ориентированных методов.	Объектно-ориентированные методы: анализ, проектирование, программирование, определение объекта, класса, основные термины и определения	2
2	Класс и его структура C#/ Методы и свойства. Поведение.	Определение поведения и структуры класса, синтаксис описания класса (C#)	2
3	Основные парадигмы объектных методов	Абстрагирование, инкапсуляция, иерархия, типизация, параллелизм, сохраняемость	2
4	Наследование и полиморфизм C#	Наследование. Одиночное наследование. Проблемы множественного наследования. Виды полиморфизма по Вегнеру, специальный и универсальный полиморфизмы, полиморфизм наследования, проблема среза, использование полиморфизма	2
5	Статические и виртуальные функции C#. Абстрактные классы и интерфейсы C#	Знакомство с модификаторами virtual и override, отличия статистических и виртуальных функций. Перегрузка для обеспечения универсального полиморфизма. Абстрактный метод, абстрактный класс, интерфейс, отличие наследования интерфейсов от множественного наследования, использования интерфейсов для обобщения поведения классов разных иерархий	1
6	Основы RUP и UML. NET Framework и CLR	RUP и UML. Итеративная разработка. Место артефактов UML в процессе разработки. Диаграммы классов. Концептуальные модели классов. Виды связей. NET Framework. Назначение и составляющие. CLR и его преимущества. Управляемое выполнение. Управляемый код и язык C#. углубленное изучение конструкций языка C#	1
7	Компонентно-ориентированное программирование	Компоненто-ориентированное программирование, преимущества компонентов, требования к компонентам.	1
8	Введение в шаблоны проектирования	Шаблоны проектирования, состав описания шаблонов, классификация, краткое знакомство с шаблонами GRASP, GoF. Шаблон MVC, модель, представление, контроллер.	1
		ИТОГО	12

3.1.3. Наименование тем (вопросов), выделенных для самостоятельной работы студентов

№ тем	Наименование темы (вопроса)	Основное содержание темы (вопроса)	Объем в часах	Литература

6 семестр				
1	История языка программирования C++. Структура программы на C++. Необъектно-ориентированные расширения C++. Обзор среды программирования.	Обзор среды программирования. Горячие клавиши и дополнительные возможности. Создание проектов, компиляция и отладка программ	10	ОЛ-3,4
2	Классы и объекты в C++. Специальные функции класса.	Виртуальные функции. Дружественные функции. Дружественные классы.	10	ОЛ-3,4
3	Перегрузка операторов в классах. Приведение типов.	Способы перегрузки операции: 1) как глобальная функция, 2) как метод класса. Особенности перегрузки глобальной операторной функции. Правила перегрузки операторов. Дружественные функции. Особенности перегрузки операторной функции как метода класса. Рекомендации по выбору способа перегрузки	10	ОЛ-3,4
4	Наследование в языке C++. Полиморфизм наследования в языке. Множественное наследование в. Параметрический полиморфизм C++.	Синтаксис наследования. Права доступа при наследовании. Общее и частное наследование. Иерархии классов. Работа конструктора и деструктора. Примеры множественного наследования. Виртуальные базовые классы. Параметрический полиморфизм. Шаблоны. Параметры шаблона. Шаблон функции – универсальный алгоритм. Вызов шаблонной функции. Специализация шаблонов. Выведение типов. Шаблон класса – универсальный контейнер.	12	ОЛ-3,4
5	Стандартная библиотека шаблонов C++ (stl)	Философия STL: контейнеры, алгоритмы, потоки, итераторы. Понятие компонента Компоненты в COM Компоненты в .NET	12	ОЛ-3,4
		Итого	54	
7 семестр				
4	Наследование и полиморфизм C#	История развития парадигмы полиморфного поведения объектов	14	ОЛ-1,2
6	Основы RUP и UML. NET Framework и CLR	Другие модели нотации UML, этап развития RUP. Глубокое изучение синтаксиса и конструкций языка C#,	14	ОЛ-1,2
7	Компонентно-ориентированное программирование	История развития и подходы к компонентному программированию. DCOM, CORBA и другие подходы к компонентному программированию.	12	ОЛ-1,2
8	Введение в шаблоны проектирования	Приёмы объектно-ориентированного проектирования. Паттерны проектирования	14	ОЛ-1,2 ДЛ-1,2
		ИТОГО	54	

3.1.4. Практические занятия, их содержание и объем в часах (по семестрам)

Номер темы	Наименование практических занятий (семинаров)	Основное содержание практических занятий (семинаров)	Количество часов
4 семестр			

1	История языка программирования C++. Структура программы на C++. Необъектно-ориентированные расширения C++. Обзор среды программирования.	Начало работы с среде программирования. Консоль. Создание проекта графического приложения. Обзор файлов проекта. Создание простейшей программы	6
2	Классы и объекты в C++. Специальные функции класса.	Разработка и использование классов. Примеры простых программ с использованием классов.	8
3	Перегрузка операторов в классах. Приведение типов.	Совместное использование функций и методов. Сигнатуры и правила отождествления вызова. Перегрузка операторов. Специальные методы классов, конструкторы и деструкторы. Автоматическая и динамическая память. Примеры простых программ с перегрузкой операторов.	8
4	Наследование в языке C++. Полиморфизм наследования в языке. Множественное наследование в. Параметрический полиморфизм C++.	Наследование классов. Переопределение методов. Виртуальные и статические методы. Примеры простых программ с иерархией классов. Создание консольного приложения, состоящего из нескольких файлов. Реализация шаблона класса - контейнера.	8
		Итого	30
	5 семестр		
2	Класс и его структура C#/ Методы и свойства. Поведение.	Написание простого набора классов, создание и использование объектов	4
3	Основные парадигмы объектных методов	Использования инкапсуляции для сокрытия реализации класса. Создание простой иерархии классов	4
4	Наследование и полиморфизм C#	Применение полиморфного поведения для управления большим количеством схожих объектов	4
5	Статические и виртуальные функции C#. Абстрактные классы и интерфейсы C#	Применение виртуальных функций. Использование абстрактных классов и интерфейсов	4
6	Основы RUP и UML. NET Framework и CLR	Построение и чтение диаграмм классов. Примеры применения конструкций языка C#	4
7	Компонентно-ориентированное программирование	Создание простого компонента	4
8	Введение в шаблоны проектирования	Применение шаблона Controller на практике	6
		ИТОГО	30

3.1.5. Лабораторные занятия, их наименование и объем в часах

Номер работы	Наименование лабораторной работы	Объем в часах
	6,7 семестр	
1	Разработка приложения для создания и редактирования товарной накладной с использованием модели классов.	4
2	Построение приложения для продажи авиабилетов	4
3	Построение приложения для обеспечения работы заправочной станции	4
	ИТОГО	16

3.2. Перечень тем курсовых проектов (работ)

№ п-п	Наименование проекта (работы)
	Не предусмотрены рабочим учебным планом

3.3. Перечень тем РГР

№ п-п	Наименование проекта (работы)
1	Варианты расчетно-графических работ

3.4. Перечень тем рефератов

№ п-п	Наименование проекта (работы)
	Не предусмотрены рабочим учебным планом

3.5. Перечень тем контрольных работ

№ п-п	Наименование проекта (работы)
	Не предусмотрены рабочим учебным планом

3.6 Интерактивные образовательные технологии, используемые при проведении учебных занятий

Семестр	Вид занятий (лекции, практические, лабораторные)	Тема	Формируемая компетенция	Интерактив	Количество часов

4. Перечень учебно-методического обеспечения для самостоятельной работы обучающихся по дисциплине, основной и дополнительной учебной литературы, необходимой для освоения дисциплины

4.1. Основная и дополнительная литература

№ п-п	Автор и наименование	Вид пособия	Год издания	Кол-во экз. в библиотеке
Основная литература				
ОЛ-1	Объектно-ориентированное программирование с примерами на C#: Учебное пособие / Хорев П.Б. - М.: Форум, НИЦ ИНФРА-М, 2016. - 200 с.	У	2016	http://znanium.com/catalog/product/529350
ОЛ-3	Объектно-ориентированное программирование на C++ : учебник / И. В. Баранова, С. Н. Баранов, И. В. Баженова [и др.]. - Красноярск : Сиб. федер. ун-т, 2019. - 288 с.	У	2019	https://znanium.com/catalog/product/1819676
ОЛ-4	Ашарина, И.В. Объектно-ориентированное программирование в C++: лекции и упражнения : учеб. пособие для вузов / И.В. Ашарина. - 2-е изд., перераб. и доп. - Москва : Горячая линия - Телеком, 2017. - 336 с.	УП	2017	https://znanium.com/catalog/product/1040247
Дополнительная литература				
ДЛ-1	Хорев, П. Б. Объектно-ориентированное программирование с примерами на C# : учебное пособие / П.Б. Хорев. — Москва : ФОРУМ : ИНФРА-М, 2023. — 200 с.	УП	2023	https://znanium.com/catalog/document?id=424788#ant
ДЛ-2	Зюзев, А. М. Объектно ориентированное программирование : учебно-методическое пособие / А. М. Зюзев, К. Е. Нестеров. - Екатеринбург : Изд-во Уральского ун-та, 2019. - 116 с.	УП	2019	https://znanium.com/catalog/document?id=422382#ant

4.2. Методические пособия и указания

№№ п-п	Наименование	Год издания (состава)	Кол-во экз.

5. Программное обеспечение и Интернет-ресурсы

5.1. Перечень ресурсов информационно-телекоммуникационной сети "Интернет", необходимых для освоения дисциплины

№	Интернет-ресурс	Характеристика
1	http://intuit.ru	Национальный Открытый Университет «ИНТУИТ» — организация, предоставляющая с помощью собственного сайта услуги дистанционного обучения по нескольким образовательным программам, многие из которых каса-ются информационных технологий. Сайт содержит несколько сотен открытых образовательных курсов, по прохождении которых можно бесплатно получить электронный сертификат.
2	http:// znanium.com	Электронная библиотечная система : содержит электронные версии книг издательства Инфра-М и других ведущих издательств учебной литературы, так и электронные версии периодических изданий по естественным, техническим и гуманитарным наукам

5.2. Перечень информационных технологий, используемых при осуществлении образовательного процесса по дисциплине, включая перечень программного обеспечения и информационных справочных систем (при необходимости)

Программное обеспечение, в т.ч.:

– для выполнения технологических расчетов и письменных работ: «Microsoft Office», «Microsoft Excel»;
– для компьютерной демонстрации презентаций: «Microsoft PowerPoint»;
– PlantUML
– Visual Studio Code, Microsoft Visual Studio

6. Фонд оценочных средств для проведения промежуточной аттестации обучающихся по дисциплине представлен в Приложении.

7. Описание материально-технической базы, необходимой для осуществления образовательного процесса по дисциплине

Учебная аудитория для проведения занятий лекционного типа, занятий семинарского типа (практических занятий), групповых и индивидуальных консультаций, для текущего контроля и промежуточной аттестации.

Приложение

ФОНД ОЦЕНОЧНЫХ СРЕДСТВ ПО ДИСЦИПЛИНЕ

Объектно-ориентированное программирование

Направление подготовки: 09.03.01 Информатика и вычислительная техника

Профиль подготовки: "Искусственный интеллект и машинное обучение "

1. Перечень компетенций и этапы их формирования

Код и наименование компетенции	**Этапы формирования компетенции (семестр/ раздел/ тема дисциплины)**	**Дескрипторные характеристики компетенции (основные признаки)**
ПК-6 Способен разрабатывать стратегии тестированияи управление процессом тестирования, разрабатывать документы для тестирования и анализировать качествопокрытия	Семестр 6. Темы 1-5 Семестр 7. Темы 1-8	*Знать*: принципы объектно-ориентированного подхода, сфера применения, преимущества и недостатки объектных методов, основные парадигмы объектных методов: абстрагирование, инкапсуляция, иерархия, типизация, параллелизм, сохраняемость, приёмы и синтаксис для создания и использования классов и объектов, свойства и атрибуты класса, методы, модификаторы доступа, private, public, protected, принципы структурного и объектно-ориентированного программирования, терминологию (понятийный аппарат) объектно-ориентированного анализа, проектирования, программирования: поведение, идентичность, иерархия, абстракция, Основы RUPи UML. NETFrameworkи CLR, Компонентно-ориентированное программирование, шаблоны UML, Модель классов и ее компоненты, параметрический полиморфизм *Уметь*: применять полученные теоретические знания и практические навыки проектирования программных продуктов, создавать описания пользовательских классов на языке C#, использовать объекты и классы в программном коде на языке C#, создавать классы и структуры *Владеть*: терминологическим базисом объектных методов, современной средой разработки прикладных приложений (Visual Studio Code), методами объектно-ориентированного анализа при проектировании программных продуктов для различных предметных областей *Иметь навык:* работы и создания консольных и графических приложений Visual Studio Code *Быть способным*. создавать пользовательские интерфейсы Visual Studio Code с использованием различных элементов пользовательского интерфейса. применять шаблоны одиночка,

		контроллер при создании собственных программ
ПК-6 Способен разрабатывать стратегии тестирования и управление процессом тестирования, разрабатывать документы для тестирования и анализировать качествопокрытия	Семестр 4. Темы 1-5 Семестр 5. Темы 1-8	*Знать*: синтаксис создания классов и структур, основные компоненты приложения компонент пользовательского интерфейса Visual Studio Code, свойства объектов, события компонент, методику анализа и проектирования объектно-ориентированных программ, основные понятия; синтаксис и семантику конструкций языка программирования; методы обработки и способы реализации основных структур данных в объектно-ориентированных программных средах; шаблоны проектирования, основные парадигмы объектных методов; приёмы и синтаксис для создания и использования классов и объектов; основные принципы и шаблоны распределения обязанностей между объектами *Уметь*: проектировать и реализовывать простые прикладные приложения на языке C#, создавать абстрактные классы и виртуальные методы, использовать наследование и полиморфизм объектов, использовать методы инкапсуляции при реализации классов, использовать свойства для получения доступа к закрытым полям класса, записывать простейшие алгоритмы на алгоритмическом языке программирования; создавать простейшие программы (объекты, классы) в среде C++, иллюстрирующие основы технологии объектно-ориентированного программирования (компонентного проектирования программ). *Владеть*: использование виртуальных методов в наследуемых классах, использование статических классов и методов, навыками работы в визуальной среде программирования C++; навыками структурной алгоритмизации и технологии структурного программирования *Иметь навык:* разрабатывать простые интерфейсы на языке C# *Быть способным:* отличить интерфейс и абстрактный класс

2. Паспорт фонда оценочных средств

№ п/п	Контролируемые дидактические единицы (разделы, темы) дисциплины	Код контролируемой компетенции	Уровень	Наименование оценочного средства	Представление оценочного средства в фонде
1	Тема 1-8	ПК-6	Пороговый	Опрос	Вопросы для собеседования
				Лабораторные работы	Задания для лабораторных работ
				Тест	банк тестовых задания
				Расчетно-графическая работа	задания для расчетно-графических работ
			Повышенный	Метод проектов	банк заданий
2	Семестр 4 Тема 1-5	ПК-6	Обязательный	Зачет	Банк вопросов к зачету
3	Семестр 5 Тема 1-8	ПК-6	Обязательный	Экзамен	Банк вопросов к экзамену

3. Показатели и критерии оценивания компетенций, описание шкал оценивания

Код компетенции	Показатели сформированности	Шкала оценивания	Критерии оценивания
ПК-6	*Знать* принципы объектно-ориентированного подхода, основные парадигмы объектных методов: абстрагирование, инкапсуляция, иерархия, типизация, параллелизм, сохраняемость, приёмы и синтаксис для создания и использования классов и объектов, свойства и атрибуты класса, методы, модификаторы доступа, private, public, protected,	*Пороговый уровень (обязательный)*	*Знать:* принципы объектно-ориентированного подхода, основные парадигмы объектных методов: абстрагирование, инкапсуляция, приёмы и синтаксис для создания и использования классов и объектов, свойства и атрибуты класса, методы, модификаторы доступа, private, public, protected, поведение, идентичность, иерархия, абстракция.
		Повышенный уровень (по отношению к пороговому уровню)	*Знать:* , типизация, параллелизм, сохраняемость
	Уметь создавать описания пользовательских классов на языке Delphi или C#, использовать объекты и классы в программном коде на языке Delphi или C#, создавать классы и	*Пороговый уровень (обязательный)*	*Уметь* создавать описания пользовательских классов на языке C#, использовать объекты и классы в программном коде на языке C#, создавать классы и структуры, создавать пользовательские интерфейсы Visual Studio или

	структуры, создавать пользовательские интерфейсы Visual Studio или Embarcadero RAD Studio	*Повышенный уровень (по отношению к пороговому уровню)*	*Уметь* создавать описания пользовательских классов на языке C#, использовать объекты и классы в программном коде на языке C#, создавать классы и структуры, создавать пользовательские интерфейсы
	Владеть терминологическим базисом объектных методов, современной средой разработки прикладных приложений (Visual Studio или Embarcadero RAD Studio), работы и создания консольных и графических приложений	*Пороговый уровень (обязательный)*	*Владеть* терминологическим базисом объектных методов, современной средой разработки прикладных приложений (Visual Studio Code)
		Повышенный уровень (по отношению к пороговому уровню)	*Владеть: навыками* работы и создания консольных и графических приложений
ПК-6	*Знать*: синтаксис создания классов и структур, основные компоненты приложения компонент пользовательского интерфейса Visual Stuido, свойства объектов, события компонент, основы RUP и UML. NET Framework и CLR, компонето-ориентированное программирование, основные принципы и шаблоны распределения обязанностей между объектами, проблемы множественного наследования, назначение и составляющие, шаблоны фабрика, CLR и его преимущества. Управляемое выполнение. Управляемый код и язык C#, шаблоны проектирования, состав описания шаблонов, классификация, GoF, , фабричный метод	*Пороговый уровень (обязательный)*	*Знать* синтаксис создания классов и структур, свойства объектов, события компонент, основы RUP и UML. NET Framework и CLR, компонето-ориентированное программирование, основные принципы и шаблоны распределения обязанностей между объектами, проблемы множественного наследования, назначение и составляющие, шаблоны фабрика
		Повышенный уровень (по отношению к пороговому уровню)	*Знать* основные компоненты приложения компонент пользовательского интерфейса Visual Studio Code , CLR и его преимущества. Управляемое выполнение. Управляемый код и язык C#, шаблоны проектирования, состав описания шаблонов, классификация, GoF, , фабричный метод
	Уметь проектировать и реализовывать простые прикладные приложения на языке C#, создавать абстрактные классы и виртуальные методы, использо-	*Пороговый уровень (обязательный)*	*Уметь*. проектировать и реализовывать простые прикладные приложения на языке Delphi или C#, создавать абстрактные классы и виртуальные методы, использовать наследование и полиморфизм объектов

	вать наследование и полиморфизм объектов, использовать методы инкапсуляции при реализации классов, использовать свойства для получения доступа к закрытым полям класса, использовать наследование, одиночное наследование, специальный и универсальный полиморфизмы, использовать шаблон MVC, модель, представление, контроллер	*Повышенный уровень (по отношению к пороговому уровню)*	*Уметь* использовать методы инкапсуляции при реализации классов, использовать свойства для получения доступа к закрытым полям класса, использовать наследование, одиночное наследование, специальный и универсальный полиморфизмы, использовать шаблон MVC, модель, представление, контроллер
	Владеть использование виртуальных методов в наследуемых классах, использование статических классов и методов, способами проектирования основных шаблонов: фабрика, фабричный метод, навыками разбивать приложение на слои согласно шаблону MVC	*Пороговый уровень (обязательный)*	*Владеть* использование виртуальных методов в наследуемых классах, способами проектирования основных шаблонов: фабрика
		Повышенный уровень (по отношению к пороговому уровню)	*Владеть* использование статических классов и методов, способами проектирования основных шаблонов: фабричный метод, навыками разбивать приложение на слои согласно шаблону MVC

4. Компетентностно-ориентированные задания (КОЗ)

Данные КОЗ представляют собой комплексные задания, предназначенные для контроля уровня успеваемости и освоения компетенций у студента по всем темам дисциплины. Основным средством формирования компетентностей выступают компетентностно-ориентированные задания:

- банк заданий;
- банк тестовых заданий;
- задания для расчетно-графических работ;
- задания для лабораторных работ;
- банк вопросов к экзамену;
- банк вопросов к зачету;
- вопросы для собеседования.

I. Задание для лабораторных работ

Проверка сформированности компетенций: ПК-6.

Номер работы	Наименование лабораторной работы	Краткое содержание

1	Разработка приложения для создания и редактирования товарной накладной с использованием модели классов.	Создать приложение, которое позволяет учитывать данные товарных накладных. Накладная – это документ, используемый при передаче товарно-материальных ценностей от одного лица другому. В частности, товарная накладная это документ, предназначенный для оформления операций по отпуску и приёму товаров, который включает: название организации, номер накладной, дату отпуска товара, его наименование, кем отпущен товар, кому отпущен товар, его количество, сорт, цена и другие данные, основание для отпуска товара, подписи материально ответственных лиц в его отпуске и приеме
2	Построение приложения для продажи авиабилетов	Авиакомпания осуществляет рейсы, характеризующиеся: номером рейса, аэропортом вылета и аэропортом назначения, ценой за билет. Билеты продаются на определённый рейс и характеризуются следующими атрибутами: датой отправления, именем пассажира и его паспортными данными. Номер посадочного места и номер выхода на посадку (гейта) вписываются в посадочный талон пассажира после прохождения им регистрации
3	Построение приложения для обеспечения работы заправочной станции	Сеть автозаправочных станций торгуют бензином марок: «АИ-80», «АИ-92», «АИ-95» и дизельным топливом («ДТ»). Каждый вид топлива хранится в предназначенной для него цистерне. При покупке клиент называет марку и объём (в литрах) топлива и расплачивается. При этом системадолжна проверить возможность удовлетво- рения запроса: достаточный объём топлива в хранилище и достаточность суммы покупателя. Если условия соблюдаются, система уменьшает запас топлива в одной из цистерн и фиксирует акт продажи: марку проданного топлива, объём, стоимость, дату/время продажи. Топливо на станции подвозится бензовозами и распределяется по имеющимся цистернам в соответствии с марками топлива
*	Построение простого редактора геометрических фигур с использованием абстрактных классов	Создать векторный графический редактор. Графический редактор позволяет добавлять двумерные геометрические фигуры на форму, а также осуществлять их перемещение в другие координаты. Необходимо реализовать возможность рисовать и перетаскивать прямоугольники и круги различных цветов

Краткие методические указания.

На выполнение одной лабораторной работы отводится не менее двух двухчасовых занятий. После выполнения каждой лабораторной работы студент должен представить отчет о ее выполнении, а также при необходимости, по указаниям преподавателя, выполнить дополнительные практические задания или ответить на вопросы по теме лабораторной работы.

Подробное описание задание представлено здесь:

http://lib.ugtu.net/book/28244/

Задание для работы №1.

1. Добавьте в таблицу столбцы «Цена», «Ставка НДС», «Цена с НДС».
   a. Цена должна рассчитываться произведением «Цены единицы товара» на «Количество».
   b. Значение «Ставка НДС» должна вводиться пользователем.
   c. Значение «Цена с НДС» должна рассчитываться автоматически.
2. Добавьте возможность создания нескольких накладных.
   a. Номер накладной должен создаваться автоматически, путём увеличения значения номера последней накладной на единицу
   b. Должна быть предусмотрена возможность ввода номера накладной пользователем.
3. Предусмотрите вывод на форме строки суммы накладной прописью в формате «<рубли прописью> рублей, <копейки прописью> копеек»
4. Предусмотрите выгрузку данных по накладной в файл
   a. Сформулируйте требования к выходному формату файла
   b. Реализуйте выгрузку в соответствие с требованиями.

Вопросы для обсуждения:

1. В этом примере мы обошлись без создания классов предметной области, ограничившись реализацией всех функций на уровне классов пользовательского интерфейса. Дайте по крайней мере два аргумента в пользу предложенного дизайна приложения и также два аргумента в пользу изменения дизайна, в котором выделен слой классов предметной области.
2. Предложите свою объектную модель предметной области для приложения.
3. Какие недостатки предложенного пользовательского интерфейса приложения вы могли бы назвать? Предложите улучшения.
4. Как должен быть пересмотрен дизайн приложения, если в будущем появится необходимость создания накладных в валюте, отличной от российского рубля? Насколько разработанная вами в п. 2 объектная модель адаптирована к реализации этого требования?
5. Достаточно ли данных вводится в программу для генерации реальных накладных на товар? Сравните с существующими аналогами.
6. Сравните два возможных подхода к выгрузке накладной в файл: выгрузка в файл xml и выгрузка в файл Microsoft Excel. Приведите по крайней мере один довод «за» и один довод «против» использования каждого из форматов.

Задание для работы №2.

1. Напишите код, заполняющий поля ввода «аэропорт вылета», «аэропорт назначения» и «Стоимость билета» (на главной форме) при перемещении по списку рейсов.
2. Разработайте метод, который позволил бы осуществлять автоматическое заполнение номера посадочного места для пассажира. При этом количество мест для самолёта ограничено. Если на заданный рейс все места закончились, программа должна сообщать об этом.

3. Дополните программу возможностью удаления рейсов и проданных билетов (например, в случае возврата билета пассажиром). Помните, что при удалении рейса все проданные на него билеты должны быть исключены из списка.
4. Дополните программу возможностью сохранения и загрузки информации о билетах из файла.
5. Добавьте в программу возможность поиска проданных билетов по различным критериям:

a. По номеру рейса

b. По ФИО пассажира

c. По дате

6. Реализуйте интерфейс пользователя для поиска информации на отдельной форме.

Вопросы для обсуждения

1. В этом примере мы осуществляли доступ к компонентам созданных форм frmFlight , frmTicket и frmRegistration из родительской формы. Чем хорош и чем плох такой спо- соб? Знаете ли вы другой способ получить информацию от дочерних форм?
2. Какие недостатки предложенного пользовательского интерфейса приложения вы могли бы назвать? Предложите меры по улучшению.
3. Как вы думаете, достаточно ли данных вводится в программу для печати действитель- ных посадочных талонов? Что нужно добавить?

Задание для работы №3.

1. Спроектируйте и реализуйте классы «Прямоугольник» и «Окружность», наследующие классу «Фигура». Помните, что они должны иметь переопределенные методы отри- совки, определения принадлежности точки и смещения на передаваемые координаты.
2. Реализуйте все методы класса-контроллера PaintController. Помните, что наши фигуры уже «умеют» большую часть того, что необходимо.
3. Разместите необходимые компоненты (PaintBox, элементы управления) на главной форме. При помощи событий свяжите их с методами контроллера.
4. Спроектируйте и реализуйте класс «Линия». Придумайте для этой фигуры механизм перетаскивания.

Вопросы для обсуждения

1. Какие поля вы включили в свои классы «Прямоугольник» и «Окружность»? Какие методы? Почему?
2. Что нужно изменить в классах фигур, чтобы предусмотреть также цвет заливки фигуры?
3. Представьте, что мы хотим реализовать этот графический редактор без создания классов фигур. С какими проблемами мы столкнемся? Будет ли такая реализация иметь преимущества?

Задание для работы №4.

1. Самостоятельно реализуйте создание отчетной формы по поставкам
2. Самостоятельно реализуйте создание отчетной формы по продажам
3. Самостоятельно реализуйте создание отчетной формы по остаткам топлива

Вопросы для обсуждения

1. В этом примере мы не храним текущие остатки в классе «Цистерна». Назовите плюсы и минусы такого подхода. Что предпочли бы вы: хранить или не хранить текущее зна- чение остатка в цистерне?
2. Какие недостатки предложенного пользовательского интерфейса приложения вы могли бы назвать? Предложите меры по улучшению.

3. Нужно ли изменять структуру классов, чтобы иметь возможность распределить один вид топлива в несколько цистерн и увеличить количество цистерн на заправке? Как нужно будет изменить программу?

II. Вопросы для собеседования (4 семестр)

Проверка сформированности компетенций: ПК-6.

Обучающемуся предлагается пройти собеседование (устно/письменно) по одной из тем из предложенного списка.

Тема 1. Введение в программирование

1. Что такое алгоритм?
2. Свойства алгоритма.
3. Реализации алгоритма.
4. Что такое программа?
5. Какой объект программа для ЭВМ?
6. Что такое императивное программирование.
7. Перечислите языки императивного программирования, которые вы знаете.

Тема 2. Некоторые стили программирования.

1. Декларативное программирование.
2. Функциональное программирование.
3. Языки декларативного и функционального программирования.
3. Императивное программирование как модель машины фон Неймана.
4. Что такое машина фон Неймана? Основные принципы организации универсальной ЭВМ.
5. Процедурно-ориентированное программирование.
6. Алгоримическая декомпозиция задачи.

Тема 3 Введение я язык 'C'

1. Язык 'C' как вершина императивного программирования.
2. Модель памяти для программы на языке 'C'/"C++".
3. Операторы объявления в 'C'/"C++".
4. Операторы цикла в 'C'/"C++".
5. Операции в 'C'/"C++".
6. Организация взаимодействия подпрограмм
7. Передача параметров в подрограммы, возврат параметров.
8. Функции с переменным числом параметров

Тема 4 Некоторые детали 'C'/"C++".

1. Перегрузка функций, декорация имён.
2. Неявное преобразование типов в 'C'.
3. Явное преобразование типов в 'C'.
4. Операторы преобразования типов в "C++".

Тема 5 Введение в ООП.

1. Ограничения процедурно - ориентированного подхода.
2. Стили программирования по Гради Бучу.
3. Что такое "объект"?
4. Две стороны объекта.
5. Что такое тип? Встроенные и пользовательские типы.
6. Что такое ООП. Четыре кита ООП.
7. Краткая история ООП.

8. Перечислите основные недостатки ООП с точки зрения программиста.
9. Соотношение ООП и процедурного программирования.
10. Что такое программирование с использованием абстрактных типов?

Тема 6. Реализация ООП в "C++"

1. Суть процесса разработки класса.
2. Структура данных "class".
3. Абстрагирование.
4. Инкапсуляция.
5. Конструкторы и их свойства.
6. Деструкторы и их свойства.
7. Конструктор копирования.
8. Функции get/set.
9. Сервис по умолчанию.

Тема 7. Перегрузка операций.

1. Перегрузка бинарных операций.
2. Перегрузка унарных операций.
3. Перегрузка операции присвоения.
4. Перегрузка операций сравнения.
5. Перегрузка операций индексация.
6. Перегрузка логических операций.
7. Реализация собственной программы преобразования типов.

III. Банк заданий

Проверка сформированности компетенций: ПК-6.

В рамках практических работ студентам предлагается решить следующие задачи:

Задание 1.

Построить программу для работы со структурами Дата-Время. Программа должна обеспечивать простейшие функции для работы с данными структурами: увеличение/уменьшение на 1 день, час, минуту, секунду и т.д., изменение значений, вывод значений. Составить демонстрационную программу. Для реализации демонстрационной программы использовать отдельный модуль.

Программу построить с использованием проекта. Посмотреть работу программы в отладчике, обратить внимание на представление данных. Построить программу без отладочной информации. Обратить внимание на размер программы. Посмотреть, как выглядит оттранслированный код.

Задание 2.

Построить программу для работы со структурами-строками. Структура должна включать следующие поля: массив для хранения строки, его длину, время создания строки. Программа должна обеспечивать простейшие функции для работы с данными структурами: изменение строки, вывод строки, нахождение подстроки в строке и т.д. Составить демонстрационную программу.

Для реализации демонстрационной программы использовать отдельный модуль. Программу построить с использованием проекта. Посмотреть работу программы в отладчике, обратить внимание на представление данных. Построить программу без отладочной информации. Обратить внимание на размер программы. Посмотреть, как выглядит оттранслированный код.

Задание 3.

Построить программу для работы со структурами-окнами. Структура должна включать соответствующие поля: размер окна, его положение на экране, цвет, наличие рамки, цвет ее и т.д.

Программа должна обеспечивать простейшие функции для работы с данными структурами: отображение окна, удаление окна, изменение цветов... Составить демонстрационную программу.

Для реализации демонстрационной программы использовать отдельный модуль. Программу построить с использованием проекта. Посмотреть работу программы в отладчике, обратить внимание на представление данных. Построить программу без отладочной информации. Обратить внимание на размер программы. Посмотреть, как выглядит оттранслированный код.

Задание 4.

Построить программу для работы со структурами-многочленами. Структура должна включать соответствующие поля: порядок, набор коэффициентов. Программа должна обеспечивать простейшие функции для работы с данными структурами: вычисление значения многочлена для данного параметра, вывод многочлена в удобной форме и т.д. Составить демонстрационную программу.

Для реализации демонстрационной программы использовать отдельный модуль. Программу построить с использованием проекта. Посмотреть работу программы в отладчике, обратить внимание на представление данных. Построить программу без отладочной информации. Обратить внимание на размер программы. Посмотреть, как выглядит оттранслированный код.

Задание 5.

Построить программу для работы со структурами-квадратными матрицами. Структура должна включать соответствующие поля: порядок, набор коэффициентов. Программа должна обеспечивать простейшие функции для работы с данными структурами: транспонирование матрицы, расчет определителя, вывод матрицы в удобной форме и т.д. Составить демонстрационную программу.

Для реализации демонстрационной программы использовать отдельный модуль. Программу построить с использованием проекта. Посмотреть работу программы в отладчике, обратить внимание на представление данных. Построить программу без отладочной информации. Обратить внимание на размер программы. Посмотреть, как выглядит оттранслированный код.

Задание 6.

Построить программу для работы со структурами-правильными дробями. Структура должна включать соответствующие поля: числитель, знаменатель. Программа должна обеспечивать простейшие функции для работы с данными структурами: сложение, вычитание, умножение, деление, вывод дроби в удобной форме и т.д. Составить демонстрационную программу. Для реализации демонстрационной программы использовать отдельный модуль.

Программу построить с использованием проекта. Посмотреть работу программы в отладчике, обратить внимание на представление данных. Построить программу без отладочной информации. Обратить внимание на размер программы. Посмотреть, как выглядит оттранслированный код.

Задание 7.

Построить программу для работы со структурами-целыми произвольной точности. Структура должна включать соответствующие поля: длину и значение числа. Программа должна обеспечивать простейшие функции для работы с данными структурами: сложение, вычитание, умножение, деление, вывод числа в удобной форме в разных системах счисления и т.д. Составить демонстрационную программу. Для реализации демонстрационной программы использовать отдельный модуль.

Программу построить с использованием проекта. Посмотреть работу программы в отладчике, обратить внимание на представление данных. Построить программу без отладочной

информации. Обратить внимание на размер программы. Посмотреть, как выглядит оттранслированный код.

IV. Банк тестовых заданий

Проверка сформированности компетенций: ПК-6.

Обучающемуся предлагается решить тест.

1. Как называются элементы класса, которые относятся ко всем экземплярам объек- тов класса
    a. Статические
    b. Динамические
    c. Константные
    d. Защищенные
2. Истинно ли следующее утверждение: структура и класс имеют схожий синтаксис?
    a. Да
    b. Нет
3. В каких случаях надо иметь в классе конструктор копирования:
    a. Для уничтожения объектов из памяти
    b. Для создания дружественной функции
    c. Для выполнения операции присвоение одного элемента другому
    d. Кода нужно передать элементы класса
4. High Cohesion (сильное сцепление) - это ОО принцип, наиболее ассоциирующийся с:
    a. сокрытием реализации
    b. тем, что класс спланирован с единственным и конкретным назначением
    c. разрешением одному объекту быть видимым как разные типы
    d. тем, как много классы знают про другие только через их API
5. Что из перечисленного является недостатком ООП парадигмы?
    a. Невозможность повторного использования кода
    b. Невозможность абстракции
    c. Недостаточная гибкость в создании иерархии
    d. Поддержка языком ООП требует дополнительных ресурсов
6. Драконы умеют летать (как, например, птицы) и ползать (как, например, ящерицы). С точки зрения ООП, примером чего является данная ситуация (выберите наиболее точный вариант)?
    a. Наследование
    b. Множественное наследование
    c. Инкапсуляция
    d. Композиция
7. Как в терминах ООП называется объект, который не может быть изменён после создания?
    a. Internal object
    b. Abstract object
    c. Immutable object
    d. Sealed object
8. В чем главное отличие процедуры от функции?
    a. Процедура может быть запущена только один раз
    b. Функция запускается из основной программы, а процедура запускается сама

c. Функция возвращает значение и может участвовать в качестве операнда в выражении
d. Процедура возвращает значение и может участвовать в качестве операнда в выражении

9. Что является преимуществом объектно-ориентированного подхода перед структурированным?
a. 1. В ООП не используются процедуры и функции
b. 2. ООП позволяет объединить состояние объектов и их поведение
c. 3. ООП не поддерживает повторное использование компонентов
d. 4. ООП поддерживает разработку программ сверху-вниз

10. Какая разница между идентичностью (identity) и равенством (equality) объектов в ООП?
a. Идентичность означает, что две ссылки указывают на один и тот же объект, а равенство - что они содержат одинаковые данныеИдентичность означает, что объекты являются экземплярами одного и того же класса, а равенство - что они содержат одинаковые данные
b. Идентичность означает, что у объектов одинаковые поля, а равенство - что они содержат одинаковые данные
c. Идентичность означает, что у объектов есть общий неабстрактный предок, а равенство - любой общий предок

11. Как называется функция, которая вызывает саму себя?
a. Рекурсивной
b. Конструктором
c. Деструктором
d. Подставляемой
e. Цикличной

12. Почему следует называть объекты и компоненты понятными именами?
a. Это облегчает чтение кода.
b. Это позволяет избежать путаницы при обращении к объектам.
c. Это облегчает поиск ошибок в коде.
d. Это облегчает повторное использование вашего кода.

13. Наследование класса позволяет
a. перегрузить (перекрыть) методы родительского класса;
b. использовать открытые (public) методы родительского класса;
c. использовать защищенные (protected) методы родительского класса;
d. использовать внутренние (private) методы родительского класса.

14. Укажите несуществующее свойство объекта.
a. Состоянием
b. Поведением
c. Сохраняемостью
d. Уникальной идентичностью.

15. Что определяет инкапсуляция:
a. помогает отделить внутреннее устройство класса от его интерфейса;
b. обеспечивает возможность полиморфизма;
c. помогает понизить связность ваших классов;
d. помогает повысить зацепление ваших классов

V. Задания для расчетно-графических работ

Проверка сформированности компетенций: ПК-6. Краткие методические указания.

Сроки выполнения расчетно-графических работ объявляются преподавателем. Как правило, срок выполнения РГР — 10 календарных дней после окончания прохождения соответствующей РГР теме, но не позднее, чем за 10 дней до зачета (окончания семестра).

Расчетно-графическая работа сдается преподавателю в электронном виде в виде файла, обязательно подготовленного в виде распечатанного машинописного текста, оформленного в соответствии с требованиями ГОСТ.

Результаты проверки сообщаются преподавателем на очередном лабораторном занятии или по электронной почте на адрес группы.

В случае отметки «к защите» работа защищается обучающимся во время лабораторных занятий. В случае отметки «на доработку» обучающийся устраняет недостатки и повторно сдает исправленную работу.

При защите расчетно-графической работы студент должен уметь объяснить логику решения задачи и алгоритм работы программы, а также ответить на дополнительные вопросы преподавателя по теме РГР.

После защиты расчетно-графических работ обучающийся допускается к сдаче зачета по дисциплине.

6 семестр:

1. Отобразить файловую структуру диска в виде дерева. (Использовать компоненту C++Builder – TreeView.)
2. Класс «текст на экране». В заданном месте (клик мышкой) под заданным углом выводится строка текста. Пользователь имеет возможность передвигать и поворачивать этот текст. Таких строк на экране может быть много. Цвет текста также может меняться.
3. Класс «символьное число». Реализовать арифметические операции с целыми числами, представленными в символьном виде.
4. Класс “отрезок”.
5. Класс “прямая”.
6. Класс “треугольник”.
7. Программа триангуляции поверхности.
8. Реализовать класс «матрица».
9. Реализовать класс «обобщённый индекс»: пересчёт индекса одномерного массива объёмом N в любой заранее заданный k-мерный индекс и наоборот.
10. Реализовать класс «байт в файле»: для любого последовательного файла прямой доступ до любого байта в файле; его можно читать, писать, выполнять вставки после этого байта и перед ним.
11. Реализовать программу, «перетаскивающую» любой выделенный текст с одного места формы на другое, указываемое кликом мышки.
12. Класс “дерево”.
13. Шаблон класса “список”.
14. Шаблон класса “стек”.
15. Класс “комплексные числа”.
16. Введён растр с картой. Написать программу, позволяющую обводить границы на растре.
17. Введён рисунок (растр). Провести на нём отрезок. Повернуть рисунок так, чтобы отрезок стал вертикальным/горизонтальным.
18. Класс “граф”.

19. Класс “множество”.
20. Реализовать программу перевода арифметического выражения в польскую запись.

7 семестр:

Расчетно-графическая работа 1: «Разработка объектно-ориентированного компонента для игры Terrarium 1.2 с заданными свойствами и поведением».

Задачи на РГР:

1. Описать требования к животному в виде прецедентов.
   1.1 Описать прецеденты в текстовом виде
   1.2 Построить диаграммы последовательностей системы (SSD)
2. Спроектировать классы и наделить их поведением
   a. Отобразить классы на диаграмме классов.
   b. Описать алгоритмы поведения с использованием:
      i. словесное описание алгоритма;
      ii. UML.
3. Реализовать модуль с животным
4. Собрать статистику по итогам жизни животного.

Условные обозначения:

M – много пунктов

C – среднее количество пунктов

H – низкое количество пунктов Должно выполняться неравенство:

M > C > H;

∑(пункты за каждое свойство) < 100

Примечание:

Для тестирования поведения животного в экосистеме выбираются шаблоны животных из поставки Terrarium 1.2. Например, если вы разрабатываете травоядное животное, то для тестирования взаимодействия с хищником скомпилируйте и поместите в экосистему хищника и растение из примеров, входящих в дистрибутив. Для тестирования хищника поместите в экосистему травоядное животное.

Комплексные задания

Для комплексного проекта на двух студентов выбирается хищник и травоядный. Работа над животными идёт совместно; происходит обмен алгоритмами поведения и настройка животных на максимально эффективное сосуществование в режиме соперничества.

Например, для комплексного проекта подходят животные со следующим поведением:

Питание	**Энергия**	**Камуфляж**	**Зрение**	**Скорость**	**Описание**
Хищник	M	H	C	C	Обеспечьте способность особей к совместной охоте на жертву
Травоядный	M	H	C	C	Обеспечьте способность особей к совместному запутыванию хищника с целью истощения его запасов энергии

Расчетно-графическая работа 2: Изучение полиморфизма

Полиморфизм часто называется третьим столпом объектно-ориентированного программирования после инкапсуляции и наследования. Полиморфизм — слово греческого происхождения, означающее «многообразие форм» и имеющее несколько аспектов.

Во время выполнения объекты производного класса могут обрабатываться как объекты базового класса в таких местах, как параметры метода и коллекции или массивы. Когда это происходит, объявленный тип объекта перестает соответствовать своему типу во время выполнения.

Базовые классы могут определять и реализовывать виртуальные *методы*, а произ-водные классы — переопределять их, т. е. предоставлять свое собственное определение и реализацию. Во время выполнения, когда клиент вызывает метод, CLR (Common Language Runtime — общеязыковая исполняющая среда) — исполняющая среда для байт-кода — выполняет поиск типа объекта во время выполнения и вызывает перезапись виртуального метода. Таким образом, в исходном коде можно вызвать метод на базовом классе и привести версию производного класса метода, который необходимо выполнить.

Виртуальные методы позволяют работать с группами связанных объектов универсальным способом. Представим, например, приложение, позволяющее пользователю создавать различные виды фигур на поверхности для рисования. Во время компиляции вы еще не знаете, какие именно виды фигур создаст пользователь. При этом приложению необходимо отслеживать все различные типы создаваемых фигур и обновлять их в ответ на движения мыши. Для решения этой проблемы можно использовать полиморфизм, выполнив два основных действия.

Задание:

Создать иерархию классов, в которой каждый отдельный класс фигур является производным из общего базового класса.

Применить виртуальный метод для вызова соответствующего метода на любой производный класс через единый вызов в метод базового класса.

Для начала создайте базовый класс с именем Figure и производные классы, например Rectangle, Circle и Triangle. Присвойте классу Figure виртуальный метод с именем Draw и переопределите его в каждом производном классе для рисования конкретной фигуры, которую этот класс представляет. Создайте объект List<Figure> и добавьте в него круг, тре- угольник и прямоугольник.

```
using System.Collections.Generic; // подключение коллекций

Создание базового класса и виртуального метода class
   Figure
   {

      public Figure()
      {
      }
      public virtual void draw(TextBox txt)
      {

         txt.Text = "This is Figure";
      }
   }
```

Создание производного класса и переопределение метода class

```
Circle : Figure
{public override void draw(TextBox txt)
   {txt.Text = "This is Circle";
   }
}
```

// В texbox выводится сообщение о выбранной фигуре
List<Figure> listF – объявление коллекции фигур. listF =
new List<Figure> - создание коллекции
listF.Add(new Figure()) – добавление фигуры в коллекцию

Обновить функцию draw в каждом производном классе, по щелчку на форме (или добавить на форму объект PictureBox) в указанном месте появлялась выбранная фигура.

Переписать базовый класс Figure на абстрактный

Добавить в абстрактный класс, абстрактные методы расчета периметра и площади. Добавить в производные классы поля для хранения информации об создаваемых объектах (ширина, длина и так далее в зависимости от создаваемого объекта). Сделать невозможным редактирование полей вне тела классов.

Написать соответствующие конструкторы для производных классов

! переопределить в производных классах абстрактные методы расчета периметра и площади объектов

При рисовании объектов также должны выводится в текстовые поля значения периметра и площади

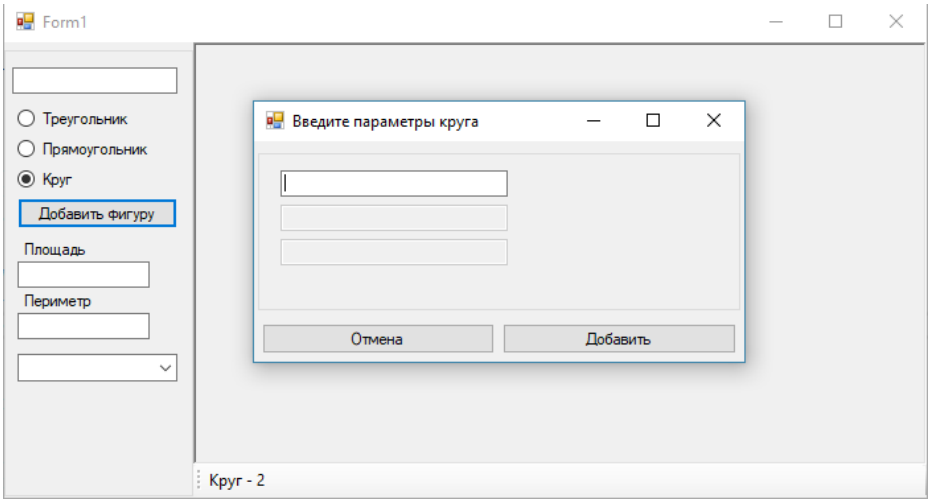
Реализуйте добавление произвольного количества фигур и разных типов в коллекцию, используя элементы пользовательского интерфейса (для выбора типа фигуры можно использовать, например, элемент radiobutton). Добавлять объект по нажатию на кнопку.

Добавьте в проект новую форму, с помощью которой реализуйте возможность определять основные параметры создаваемых фигур и параметры рисования (наличие заливки цветом фигуры, выбор цвета линии и заливки, толщина линии). Добавьте необходимые поля в классы фигур.

Пример инициализации и вызова второй формы (например, по щелчку кнопки) для ввода параметров круга

```
fCircle formCircle = new fCircle();
formCircle.Text = "Введите параметры круга";
formCircle.ShowDialog();
```

Реализуйте на новой форме защиту от ввода лишних данных (например, для круга – 1 параметр (радиус) , прямоугольник – 2, и т. д.



Вызов цветовой палитры можно реализовать через компонент ColorDialog на панели инструментов. В следующем примере Button элемента управления Click обработчик событий открывает ColorDialog компонента. Если цвет — выбран и пользователь нажимает OK, Button цвет фона элемента управления имеет значение на выбранный цвет.

```
private void button1_Click(object sender, System.EventArgs e)
{

  if(colorDialog1.ShowDialog() == DialogResult.OK)
  {

    button1.BackColor = colorDialog1.Color;
  }
}
```

Используя статические поля добавьте счетчик создаваемых фигур по каждому типу и выведите их на форму, например, используя объект ToolStrip на панели инструментов.

Элемент ToolStrip представляет панель инструментов. Каждый отдельный элемент на этой панели является объектом ToolStripItem.

Панель ToolStrip может содержать объекты следующих классов

ToolStripLabel: текстовая метка на панели инструментов, представляет функциональность элементов Label и LinkLabel

ToolStripButton: аналогичен элементу Button.Также имеет событие Click, с помощью которого можно обработать нажатие пользователя на кнопку

ToolStripSeparator: визуальный разделитель между другими элементами на панели инструментов

ToolStripToolStripComboBox: подобен стандартному элементу ComboBox

ToolStripTextBox: аналогичен текстовому полю TextBox

Измените базовый (если необходимо) и производные классы таким образом, чтобы внутри классов хранилась информации о пространственной координате (Point) фигуры на форме, по которой происходит прорисовка фигуры.

На следующем шаге измените способ добавления фигуры в список (на данном этапе он выполняется по нажатию на кнопку). Фигура в список должна добавляться вместе с координатой расположения на PictureBox. В таком случае, код с кнопки необходимо пе- ренести на событие MouseClick объекта PictureBox, где вместе с параметрами создаваемой фигуры в конструктор должна передаваться и координата клика мышки.

Преобразуйте рисование фигур на форме таким образом, чтобы каждую фигуру из списка можно было нарисовать только один раз, то есть при повторном рисовании предыдущий рисунок фигуры должен быть удален, но другие фигуры на PictureBox должны остаться.

Реализуйте возможность удаления фигур из списка.

Отчет по РГР должен содержать:

- цель работы;
- постановку задачи;
- модель представления знаний в предметной области;
- граф (дерево решений) предметной области
- метод решения задачи (описание основных фактов и правил);
- листинг программы;

VI. Банк вопросов к зачету

Проверка сформированности компетенций: ПК-6.

1. Директива препроцессора #define. Условная компиляция #if, #else, #ifndef, #endif.
2. Базовые принципы объектно-ориентированного программирования.
3. Базовые конструкции объектно-ориентированных программ.
4. Понятие полиморфизма
5. Абстрактные типы данных.
6. Потоковый ввод-вывод в C++. Открытие и закрытие потока. Стандартные потоки ввода-вывода. Использование манипуляторов для ввода/вывода в C++.
7. Операторы для динамического выделения и освобождения памяти (new и delete).
8. Перегрузка бинарного оператора.
9. Перегрузка унарного оператора.
10. Перегрузка условных операций.
11. Организация простейшего ввода/вывода в C++.
12. Перегрузка оператора ().
13. Перегрузка оператора [].
14. Перегрузка оператора ->.
15. Перегрузка оператора new.
16. Перегрузка оператора delete.
17. Особенности перегрузки оператора присваивания
18. Inline-функции класса.
19. Вложенные классы.
20. Static-компоненты данные класса.
21. Static-компоненты функции класса.
22. Использование new и delete для реализации массивов (одномерных и двумерных) объектов.
23. Const-компоненты функции класса.
24. Организация внешнего доступа к локальным компонентам класса (спецификатор friend). friend
25. Указатель this.
26. Ссылки. Параметры ссылки.
27. Конструктор и деструктор. Конструктор по умолчанию.
28. Конструктор копирования.
29. Инициализация компонент-данных объекта. Конструктор с параметрами.
30. Константные объекты и функции класса.
31. Наследование (открытое, защищенное и закрытое).
32. Виртуальные функции.
33. Виртуальное наследование.

34. Множественное наследование.
35. Пространство имен. Пространство имен как директива
36. Абстрактные базовые классы.
37. Перегрузка и переопределение функций.
38. Пространство имен. Пространство имен как объявление.
39. Конструктор explicit.
40. Виртуальные деструкторы.
41. Множественное наследование.
42. Шаблоны функций.
43. Передача в шаблон класса дополнительных параметров. Шаблоны класса и

44. Динамические структуры данных (однонаправленные и двунаправленные

списки).

45. Создание списка, печать, удаление, добавление элементов (на примере однонаправленных и двунаправленных списков).

VII. Банк вопросов к экзамену

Проверка сформированности компетенций: ПК-6.

1. Объектно-ориентированные методы: анализ, проектирование, программирование.
2. Сфера применения, преимущества и недостатки объектных методов.
3. Объект как фундаментальное понятие объектно-ориентированных методов. Дайте определение состояния, поведения и уникальной идентичности объектов.
4. Уникальная идентичность объектов. Идентичность объектов Вселенной. Идентичность объектов в реляционных и объектно-ориентированных моделях. Примеры.
5. Понятие класса в объектно-ориентированных методах. Дайте определение поведения и структуры класса.
6. Понятие класса в объектно-ориентированных методах. Понятие контракта (интерфейса) и реализации класса. Обозначение класса в UML. Примеры.
7. Фундамент объектно-ориентированных методов. Абстрагирование. Пример
8. Фундамент объектно-ориентированных методов. Инкапсуляция. Пример.
9. Фундамент объектно-ориентированных методов. Иерархия. Виды иерархий. Пример.
10. Фундамент объектно-ориентированных методов. Типизация. Языки с сильной и слабой типизацией.
11. Фундамент объектно-ориентированных методов. Параллелизм. Поддержка параллелизма в языках программирования. Поток управления.
12. Фундамент объектно-ориентированных методов. Сохраняемость. Поддержка сохраняемости в языках программирования. Сохраняемость
13. Понятие класса в C#. Отличия класса от структуры. Понятие объекта класса. Синтаксис. Пример.
14. Управление доступом к членам класса. Доступ к элементам класса. Синтаксис. Пример.
15. Конструктор класса. Виды конструкторов. Синтаксис C#. Выделение памяти для нового объекта. Пример.
16. Статические члены класса. Назначение. Синтаксис. Пример.
17. Понятие полиморфизма. Классификация типов полиморфизма по Вегнеру. Примеры.
18. Специальный полиморфизм. Перегрузка имён функций. Явные и неявные преобразования типов. Примеры на C#
19. Универсальный полиморфизм. Наследование. Проблемы классификации. Построение иерархии классов. Одиночное наследование в C#. Примеры.
20. Статические и виртуальные функции. Абстрактные классы. Примеры.

21. Интерфейсы. Отличие наследования интерфейсов от множественного наследования. Примеры.
22. Универсальный полиморфизм. Наследование. Полиморфное поведение при наследовании. Примеры на
23. Универсальный полиморфизм. Наследование. Проблема «среза». Примеры
24. Множественное наследование. Проблемы множественного наследования. Варианты решений проблем множественного наследования.
25. RUP и UML. Итеративная разработка. Место артефактов UML в процессе разработки.
26. RUP и UML. Диаграммы классов. Концептуальные модели классов. Виды связей.
27. Интерфейс. Определение и назначение. Пример интерфейса
28. Наследование интерфейсов. Преимущества и недостатки. Объясните реализацию интерфейсов на примере интерфейса IAny, напишите соответствующий код
29. Абстрактные классы. Определение и назначение. Наследование абстрактных классов.
30. Сравнение абстрактных классов и интерфейсов.
31. Множественное наследование интерфейсов. Проблемы и решения.
32. Шаблоны проектирования. Состав описания шаблонов. Классификация.
33. Шаблоны проектирования. Состав GRASP. Шаблоны «Information Expert» и «Creator».
34. Шаблоны проектирования. Состав GRASP. Шаблоны «High Cohesion» и «Low Coupling».
35. Системные события. Обработка системных событий. Виды контроллеров. Преимущества контроллеров.
36. Шаблон MVC. Модель. Представление. Контроллер.
37. Компоненто-ориентированное программирование. Преимущества компонентов. Требования к компонентам.
38. .NET Framework. Назначение и составляющие. CLR и его преимущества.
39. .NET Framework. Управляемое выполнение. Управляемый код и язык C#.

5. Методические материалы, определяющие процедуры оценивания знаний, умений, навыков и (или) опыта деятельности, характеризующих этапы формирования компетенций

Критерии оценивания выполнения расчетно-графических работ (текущий контроль, формирование компетенций ПК-6):

Баллы	Описание
От 4 до 5 баллов	студент, который выполнил все задания, обосновал выполнение элементов заданий (привел цифровые данные, правильно провел расчеты, привел факты и пр.), оформил работу с учетом ГОСТ и требований кафедры, убедительно, полно и развернуто отвечает на вопросы при защите.
От 1,1 до 3,9 баллов	студент, который выполнил все задания, обосновал выполнение элементов заданий (привел цифровые данные, правильно провел расчеты, привел факты и пр.), оформил работу с учетом ГОСТ и требований кафедры, практически отвечает на вопросы во время защиты.
От 0,1 до 1 баллов	студент, который выполнил не все задания, не обосновал выполнение элементов заданий (не привел цифровые данные, неправильно провел расчеты, не привел факты и пр.), оформил работу с грубыми нарушениями ГОСТ и требований кафедры, практически не отвечает на вопросы во время защиты.
0 баллов	студент, который не выполнил задания

«Зачтено» - набран 1 и более баллов;
«Не зачтено» - набрано менее 1 балла

Критерии оценивания тестирования (текущий контроль, формирование компетенций ПК-6):

Баллы	Процент правильных ответов
От 3 до 4 баллов	получают обучающиеся с правильным количеством ответов на тестовые вопросы 100 – 90 % от общего объема заданных тестовых вопросов
От 2 до 2,9 баллов	получают обучающиеся с правильным количеством ответов на тестовые вопросы 70 –89 %от общего объема заданных тестовых вопросов
От 1 до 1,9 баллов	получают обучающиеся с правильным количеством ответов на тестовые вопросы 50 – 69 % от общего объема заданных тестовых вопросов
От 0 до 1 баллов	получают обучающиеся с правильным количеством ответов на тестовые вопросы менее 50 % от общего объема заданных тестовых вопросов

Критерии оценивания опроса (текущий контроль, формирование компетенций ПК-6):

Баллы	Описание
От 3 до 4 баллов	ответ содержательный, уверенный и четкий; показано свободное владение материалом различной степени сложности; при ответе на дополнительные вопросы выявляется владение материалом; допускаются один-два недочета, которые студент сам исправляет по замечанию преподавателя
От 2 до 2,9 баллов	твердо усвоен основной материал; ответы удовлетворяют требованиям, установленным для оценки «отлично», но при этом допускаются две негрубые ошибки; делаются несущественные пропуски при изложении фактического
	материала; при ответе на дополнительные вопросы демонстрируется понимание требуемого материала с несущественными ошибками
От 1 до 1,9 баллов	обучаемый знает и понимает основной материал программы, основные темы, но в усвоении материала имеются пробелы; излагает его упрощенно, с небольшими ошибками и затруднениями; изложение теоретического материала приводится с ошибками, неточно или схематично; появляются затруднения при ответе на дополнительные вопросы
От 0 до 0,9 баллов	отказ от ответа; отсутствие минимальных знаний по дисциплине; присутствуют грубые ошибки в ответе; практические навыки отсутствуют; студент не способен исправить ошибки даже с помощью рекомендаций преподавателя

Критерии оценивания выполненного задания (текущий контроль, формирование компетенций ПК-6):

Баллы	Описание
От 4 до 6 баллов	задачи решены полностью, в представленном решении обоснованно получен правильный ответ
От 2 до 3,9 баллов	задачи решены полностью, но нет достаточного обоснования или при верном решении допущена вычислительная ошибка, не влияющая на правильную последовательность рассуждений, и, возможно, приведшая к неверному ответу
От 0,1 до 1,9 баллов	задачи решены частично
0 баллов	решение неверно или отсутствует

Критерии оценивания лабораторных работ (текущий контроль, формирование компетенций ПК-6):

Баллы	Описание
От 1,5 до 3 баллов	Лабораторная работа засчитывается студенту, если студент знает основные определения по теме лабораторного занятия, самостоятельно решил предложенную задачу, грамотно составил отчет, выполнил не менее 60% заданий для самостоятельной работы. Допускаются несущественные неточности в оформлении и ответах на вопросы.
От 0 до 1,4 баллов	Лабораторная работа не засчитывается студенту в случаях: наличия ошибок в расчетах, неправильного оформления отчета, искажающего смысл задания, существенных ошибок при ответах на вопросы.

Критерии оценивания зачета (промежуточный контроль, формирование компетенций ПК-6):

Баллы	Описание	Оценка
От 5 до 10 баллов	выставляется при условии, если студент показывает хорошие знания изученного учебного материала; самостоятельно, логично и последовательно излагает и интерпретирует материалы учебного курса; полностью раскрывает смысл предлагаемого вопроса; владеет основными терминами и понятиями изученного курса; показывает умение переложить теоретические знания на предполагаемый практический опыт	«зачтено»
От 0 до 4,9 баллов	выставляется при наличии серьезных упущений в процессе изложения учебного материала; в случае отсутствия знаний основных понятий и определений курса или присутствии большого количества ошибок при интерпретации основных определений; если студент показывает значительные затруднения при	«не зачтено»
	ответе на предложенные основные и дополнительные вопросы; при условии отсутствия ответа на основной и дополнительный вопросы	

Критерии оценивания зачета (промежуточный контроль, формирование компетенций ПК-6):

Баллы	Описание	Оценка
От 2,1 до 4 баллов	выставляется при условии, если студент показывает хорошие знания изученного учебного материала; самостоятельно, логично и последовательно излагает и интерпретирует материалы учебного курса; полностью раскрывает смысл предлагаемого вопроса; владеет основными терминами и понятиями изученного курса; показывает умение переложить теоретические знания на предполагаемый практический опыт	«зачтено»
От 0 до 2 баллов	выставляется при наличии серьезных упущений в процессе изложения учебного материала; в случае отсутствия знаний основных понятий и определений курса или присутствии большого количества ошибок при интерпретации основных определений; если студент показывает значительные затруднения при ответе на предложенные основные и дополнительные вопросы; при условии отсутствия ответа на основной и дополнительный вопросы	«не зачтено»

Итоговый результат освоения дисциплины и компетенций:

6 семестр:

Код компетенции	Уровень освоения	Форма контроля	% выполнения	мах результат, балл	Результат обучающегося
ПК-6	Пороговый	Расчетно-графическая работа	<50 – компетенция не освоена – 0 баллов, ≥50 – компетенция освоена – max балл	5	
		Опрос		4	
	Повышенный	Задание		6	
Всего за семестр				Среднее арифметическое по всем уровням	
				5	
ПК-6	Обязательный	Зачет	Определяется преподавателем в КОЗ	5	
ИТОГОВЫЙ РЕЗУЛЬТАТ				**до 5 баллов**	**не зачет**
				5…10 баллов	**зачет**

7 семестр:

Код компетенции	Уровень освоения	Форма контроля	% выполнения	мах результат, балл	Результат обучающегося
ПК-6	Пороговый	Лабораторные задания	<50 – компетенция не освоена – 0 баллов, ≥50 – компетенция освоена – max балл	3	
		Расчетно-графическая работа		5	
		Тест		4	
	Повышенный	Задание		6	
Всего за семестр				5 Всего за семестр	
ПК-6	Обязательный	ЗСО	Определяется преподавателем в КОЗ	6	
ИТОГОВЫЙ РЕЗУЛЬТАТ				**до 3 баллов**	**неудовлетворительно**
				3…5 баллов	**удовлетворительно**
				6…8 баллов	**хорошо**
				8…10 баллов	**отлично**