

Министерство науки и высшего образования  
Российской Федерации

Федеральное государственное бюджетное  
образовательное учреждение высшего образования  
«Донецкий государственный университет»

Физико-технический факультет  
Кафедра компьютерных технологий

УТВЕРЖДАЮ  
проректор

\_\_\_\_\_ П. А. Машаров  
«17» апреля 2025 г.  
МП

**РАБОЧАЯ ПРОГРАММА ДИСЦИПЛИНЫ**  
**«ОБЪЕКТНО-ОРИЕНТИРОВАННОЕ ПРОГРАММИРОВАНИЕ»**

Укрупненная группа направлений подготовки Программа высшего образования	44.00.00 Образование и педагогические науки Программа бакалавриата
Направление подготовки	44.03.05 Педагогическое образование (с двумя профилями подготовки)
Направленность (профиль) образовательной программы	Математика и информатика
Квалификация	Бакалавр
Форма обучения	Очная

Рабочая программа может быть адаптирована для лиц  
с ограниченными возможностями здоровья и инвалидов

Донецк 2025

Рабочая программа дисциплины «Объектно-ориентированное программирование» для обучающихся по направлению подготовки 44.03.05 Педагогическое образование (с двумя профилями подготовки) (Профиль: Математика и информатика), составлена на основании Федерального государственного образовательного стандарта высшего образования – бакалавриат по направлению подготовки 44.03.05 Педагогическое образование (с двумя профилями подготовки), утвержденного приказом Министерства образования и науки Российской Федерации от 22 февраля 2018 г. № 125 (с изм. и доп.), Порядка организации и осуществления образовательной деятельности по образовательным программам высшего образования – программам бакалавриата, программам специалитета, программам магистратуры, утвержденного приказом Министерства науки и высшего образования Российской Федерации от 06 апреля 2021 г. № 245 (с изм. и доп.), в соответствии с учебным планом, утвержденным Ученым советом ФГБОУ ВО «ДонГУ» для набора 2025 года.

Разработчик:

ст. преподаватель кафедры  
Компьютерных технологий

Т.А. Васищенко

Рабочая программа утверждена на заседании кафедры компьютерных технологий  
Протокол от 10.04.2025 г. № 12

Заведующий кафедрой

Г. В. Аверин

СОГЛАСОВАНО:

Декан факультета математики и  
информационных технологий  
16.04.2025 г.

И. А. Моисеенко

Учебно-методическая комиссия факультета математики и информационных технологий.  
Протокол от 16.04.2025 г. № 3.  
Председатель

Л. И. Селякова

Руководитель основной образовательной  
программы, д-р пед. наук, проф.  
16.04.2025 г.

Е.И. Скафа

## 1. МЕСТО ДИСЦИПЛИНЫ В СТРУКТУРЕ ОБРАЗОВАТЕЛЬНОЙ ПРОГРАММЫ

1.1. Требования к предварительной подготовке обучающихся, предшествующие и сопутствующие дисциплины, на которых основывается изучение данной:

основ информатики и алгоритмического подхода, владение базовыми языками программирования (Python, JavaScript);

дисциплины программы бакалавриата: Программирование на языках высокого уровня, Информатика и информационные технологии, Алгоритмы и структуры данных.

1.2. Дисциплины, курсовые работы и практики, для которых освоение данной дисциплины необходимо как предшествующее:

## 2. ОПИСАНИЕ ДИСЦИПЛИНЫ

### 2.1. Общая характеристика

Наименование показателя	Значение показателя
Название образовательной программы	44.03.05 Педагогическое образование (с двумя профилями подготовки) (Профиль: Математика и информатика)
Шифр и название в соответствии с учебным планом	Б1.Б.М8.8 Объектно-ориентированное программирование
Часть образовательной программы	Базовая часть
Количество зачетных единиц / всего часов	3,5 / 126

В случае предъявления от обучающегося или его родителя (законного представителя) заявления на обучение по адаптированной образовательной программе высшего образования, подкрепленного заключением психолого-медико-педагогической комиссии (ПМПК) или медико-социальной экспертизы (МСЭ) с рекомендациями создания индивидуальной программы реабилитации и абилитации (ИПРА), данная рабочая программа может быть адаптирована с учетом индивидуальных особенностей здоровья обучающегося.

### 2.2. Распределение часов по формам и периодам обучения

Форма обучения	курс	семестр	Общее количество часов					Форма контроля
			лекционные	лабораторные	практические	самостоятельной работы + контроль	всего	
Очная	3	5	26	39	–	61	126	экзамен
Очная, всего			26	39	–	61	126	

## 3. ЦЕЛИ ДИСЦИПЛИНЫ

Углубленная подготовка в области анализа (вещественного и комплексного); овладение методами вычисления интегральных преобразований; овладение современным математическим аппаратом для дальнейшего использования в науке и приложениях; формирование у студентов научного подхода.

**4. КОМПЕТЕНЦИИ ОБУЧАЮЩЕГОСЯ, ФОРМИРУЕМЫЕ В РЕЗУЛЬТАТЕ  
ОСВОЕНИЯ КОМПОНЕНТА ОБРАЗОВАТЕЛЬНОЙ ПРОГРАММЫ, ИХ ИНДИКАТОРЫ  
И ПЛАНИРУЕМЫЕ РЕЗУЛЬТАТЫ ОБУЧЕНИЯ**

Компетенции	Индикаторы	Результаты обучения
ОПК-8. Способен осуществлять педагогическую деятельность на основе специальных научных знаний	ОПК-8.12. Способен участвовать в разработке основных и дополнительных образовательных программ, разрабатывать отдельные их компоненты (в том числе с использованием информационно-коммуникационных технологий)	ОПК-8.12.1 Знает технологии проектирования и кодирования структуры данных и алгоритмов в рамках объектно-ориентированной и функциональной парадигмы на языках высокого уровня. ОПК-8.12.2 Уметь писать C#-скрипты для работы с файловой системой и базами данных. ОПК-8.12.3 Владеет навыком создания приложений с оконным интерфейсом.
	ОПК 8.19. Проектирует практико-ориентированные учебные материалы по математике и информатике с учетом индивидуальных особенностей всех категорий обучающихся, в том числе с использованием цифровых инструментов.	ОПК 8.19. 1 Знает принципы ООП и механизмы повторного использования кода. ОПК 8.19. 2 Уметь применять навыки разработки программных приложений для организации средств и способ повышения творческих способностей обучающихся. ОПК 8.19. 3 Владеет навыками построения приложений для корректной работы с памятью в C++ и типами данных в Python: строками, списками, кортежами, словарями, множествами.

**5. ПРОГРАММА ДИСЦИПЛИНЫ**

Темы	Вопросы темы
<b>Содержательный модуль 1.</b> <b>Основы объектно-ориентированного программирования</b>	
Тема 1 «Основы языка C#»	Основные понятия языка. Состав языка. Алфавит и лексемы. Идентификаторы. Ключевые слова. Операторы и разделители. Комментарии Типы данных. Классификация типов. Встроенные типы. Целые и вещественные числа. Значимые и ссылочные типы. Переменные. Константы. Операторы и выражения. Преобразования типов. Побитовые операторы. Логические операторы. Сокращенная запись арифметических операторов. Ввод-вывод. Класс Console. Методы Write, WriteLine. Форматированный вывод. Метод ReadLine и приведение к типу. Блоки кода. Области видимости. Управление ходом программы. Условный оператор. Оператор выбора Циклы. Цикл с постусловием. Цикл с предусловием. Цикл for. Ключевые слова break, pause continue.

	<p>Функции. Методы. Параметры методов. Передача параметров по значению, по ссылке. Выходной параметр. Возвращаемое значение.</p>
Тема 2 «Создание графического приложения»	<p>Основные понятия: GUI, элементы управления, события. Классы форм и окон в C#.</p> <p>Размещение элементов управления на форме. Стандартные диалоговые окна (открытие/сохранение файлов, выбор цвета и т.д.).</p> <p>Создание собственных диалоговых окон</p>
Тема 3 «Элементы управления»	<p>Кнопки, текстовые поля, метки, изображения и другие элементы управления.</p> <p>Свойства элементов управления.</p> <p>Обработка событий элементов управления.</p> <p>Создание и настройка меню.</p> <p>Добавление элементов на панели инструментов.</p> <p>Связывание элементов меню и кнопок с действиями.</p>
Тема 4 «Объектно-ориентированное программирование. Общие принципы»	<p>Понятие класса и объекта.</p> <p>Структура класса: поля, методы, свойства, конструкторы.</p> <p>Создание и использование объектов. Уровни доступа к членам класса (public, private, protected).</p> <p>Скрытие реализации и защита данных. Иерархия классов.</p> <p>Наследование методов и свойств.</p> <p>Полиморфизм: переопределение методов и виртуальные методы. Абстрактные классы и интерфейсы.</p> <p>Скрытие реализации и предоставление общего интерфейса. Связи между объектами.</p> <p>Составные объекты. Разработка модели предметной области.</p> <p>Создание многократно используемых компонентов.</p> <p>Обеспечение гибкости и масштабируемости кода.</p>
Тема 5 «Работа с файловой системой»	<p>Основы работы с файлами:</p> <p>Создание, открытие, чтение и запись файлов.</p> <p>Работа с каталогами и папками.</p> <p>Обработка ошибок при работе с файлами.</p> <p>Текстовые файлы:</p> <p>Чтение и запись текстовых строк и символов.</p> <p>Работа с кодировками текстовых файлов.</p> <p>Посимвольный и построчный ввод/вывод.</p> <p>Двоичные файлы:</p> <p>Чтение и запись байтовых данных.</p> <p>Работа с различными типами данных (int, double, string) в двоичных файлах.</p> <p>Сериализация объектов в файлы и десериализация из файлов.</p> <p>Каталоги и папки:</p> <p>Создание, удаление и переименование каталогов.</p> <p>Перемещение и копирование файлов и каталогов.</p> <p>Поиск файлов в каталогах.</p> <p>Работа с файлами большого размера.</p> <p>Многопоточная работа с файлами.</p> <p>Обеспечение безопасности при работе с файлами.</p> <p>Простые программы для работы с текстовыми файлами (чтение, запись, поиск).</p>

	Программы для копирования файлов и каталогов. Системы хранения данных, основанные на файлах.
Тема 6 «Сборка мусора, управление памятью и указатели»	Система управления памятью в C#: выделение и освобождение памяти. Локальные переменные, поля объектов и управляемые объекты. Сборка мусора (Garbage Collection) в C#: автоматическое освобождение памяти. Указатели в C#: ссылки на объекты в памяти. Операторы new и delete для выделения и освобождения памяти. Опасности ручного управления памятью: утечки памяти и висячие указатели. Алгоритмы сборки мусора: поколения, маркировка и трассировка. Факторы, влияющие на производительность сборки мусора. Настройка сборщика мусора в C#. Разница между управляемыми и неуправляемыми объектами. PInvoke: использование неуправляемых объектов из кода C#. Interop: взаимодействие с неуправляемыми библиотеками. Безопасные ссылки: автоматическая проверка null. Небезопасные ссылки: ручная проверка null. Производительность безопасных и небезопасных ссылок. Фиксированные ссылки: закрепление объектов в памяти. Опасности использования фиксированных ссылок. Альтернативы фиксированным ссылкам.
<b>Содержательный модуль 2.</b>	
<b>Дополнительные главы объектно-ориентированного программирования</b>	
Тема 7 «Основы паттернов проектирования»	Паттерны проектирования ООП. ООП в различных языках программирования. Расширенные темы ООП (метапрограммирование, аспектно-ориентированное программирование).
Тема 8 «Принципы SOLID»	Обзор принципов SOLID: S (Single Responsibility): Принцип единственной ответственности. O (Open-Closed): Принцип открытости/закрытости. L (Liskov Substitution): Принцип подстановки Лисков. I (Interface Segregation): Принцип разделения интерфейсов. D (Dependency Inversion): Принцип инверсии зависимостей. Понимание каждого принципа: S: Класс или модуль должен иметь только одну причину для изменения. O: Классы должны быть открыты для расширения, но закрыты для изменения. L: Подтипы должны быть взаимозаменяемы со своими базовыми типами. I: Клиенты не должны зависеть от реализации интерфейсов, а должны зависеть только от интерфейсов. D: Модули высокого уровня не должны зависеть от модулей низкого уровня; оба должны зависеть от абстракций. Преимущества применения SOLID: Повышение гибкости и масштабируемости кода. Улучшение модульности и понятности кода. Снижение вероятности ошибок и облегчение тестирования.

	<p>Создание более устойчивых к изменениям систем.</p> <p>Реализация SOLID в C#:</p> <p>S: Использование интерфейсов и абстрактных классов для разделения ответственности.</p> <p>O: Использование наследования для расширения функциональности без изменения существующего кода.</p> <p>L: Обеспечение соответствия подтипов базовым типам по контрактам.</p> <p>I: Разделение больших интерфейсов на меньшие специализированные интерфейсы.</p> <p>D: Использование внедрения зависимостей и инъекции зависимостей.</p> <p>Примеры применения SOLID:</p> <p>Разработка многократно используемых компонентов.</p> <p>Создание больших и сложных программных систем.</p> <p>Обеспечение надежности и maintainability кода.</p>
Тема 9 «Введение в базы данных»	<p>Основы баз данных:</p> <p>Что такое база данных?</p> <p>Модели данных: реляционная, иерархическая, сетевая, объектно-ориентированная.</p> <p>Основные понятия: таблица, строка, столбец, первичный ключ, внешний ключ.</p> <p>СУБД (системы управления базами данных):</p> <p>Назначение СУБД.</p> <p>Популярные СУБД: MySQL, PostgreSQL, SQL Server, Oracle.</p> <p>Основные функции СУБД: создание, изменение, удаление данных; обеспечение целостности данных; защита данных; предоставление доступа к данным.</p> <p>Язык SQL (Structured Query Language):</p> <p>Назначение SQL.</p> <p>Базовые операторы SQL: SELECT, INSERT, UPDATE, DELETE.</p> <p>Составление простых и сложных запросов к базе данных.</p> <p>Агрегатные функции SQL: SUM, AVG, COUNT, MIN, MAX.</p> <p>Работа с базами данных из C#:</p> <p>ADO.NET (Access to Data Objects .NET): набор классов для работы с базами данных из C#.</p> <p>Подключение к базе данных.</p> <p>Выполнение SQL-запросов.</p> <p>Обработка результатов запросов.</p> <p>Создание приложения с использованием базы данных:</p> <p>Простой пример приложения C# для работы с базой данных (например, телефонная книга).</p> <p>Разработка интерфейса пользователя.</p> <p>Подключение к базе данных.</p> <p>Выполнение запросов к базе данных.</p> <p>Отображение результатов запросов в интерфейсе пользователя.</p>
Тема 10 «Введение в сетевое программирование»	<p>Основы сетевого программирования:</p> <p>Сети и протоколы: Модели OSI и TCP/IP, основные протоколы (HTTP, FTP, TCP, UDP).</p> <p>Сетевые адреса: IP-адреса, доменные имена, порты.</p> <p>Сокеты: Программирование сокетов в C#, основы клиент-серверного взаимодействия.</p>

	<p>Работа с TCP/IP в C#:</p> <p>Классы Socket, TcpClient, TcpListener для работы с TCP-соединениями.</p> <p>Отправка и получение данных по сети.</p> <p>Асинхронное сетевое программирование.</p> <p>Протокол HTTP и веб-разработка:</p> <p>Основы протокола HTTP: методы запросов (GET, POST, PUT, DELETE) и ответы сервера.</p> <p>Работа с HTTP-клиентами в C#: библиотека HttpClient.</p> <p>Отправка HTTP-запросов и обработка ответов.</p> <p>Потребление REST API.</p> <p>Сетевое программирование с использованием библиотек:</p> <p>Библиотека System.Net для базовых сетевых операций.</p> <p>Библиотека WebSockets для работы с WebSockets.</p> <p>Библиотека HttpClient для работы с HTTP.</p> <p>Примеры сетевых приложений:</p> <p>Простой HTTP-клиент.</p> <p>Чат-клиент/сервер.</p> <p>Простой веб-сервер.</p>
Тема 11 «Основы рисования в C#»	<p>Графические библиотеки в C#:</p> <p>System.Drawing: базовая библиотека для рисования простых фигур (линии, круги, прямоугольники).</p> <p>GDI+: расширенная библиотека для рисования более сложных объектов, растровых изображений и работы с текстом.</p> <p>WPF: графическая подсистема Windows Presentation Foundation, позволяющая создавать современные пользовательские интерфейсы с использованием векторной графики.</p> <p>Рисование простых фигур:</p> <p>Использование классов Pen, Brush и Graphics для рисования линий, фигур и заливки.</p> <p>Преобразование координат и масштабирование.</p> <p>Работа с цветом.</p> <p>Рисование с помощью GDI+:</p> <p>Создание и использование графических объектов (изображения, кисти, перья).</p> <p>Работа с растровыми изображениями (загрузка, сохранение, изменение).</p> <p>Отображение текста на графических объектах.</p> <p>Рисование в WPF:</p> <p>Использование элементов управления DrawingBrush и Image для отображения графики.</p> <p>Создание векторных фигур с помощью классов Path и Geometry.</p> <p>Работа с анимацией в WPF.</p> <p>Примеры приложений:</p> <p>Простой графический редактор для рисования линий и фигур.</p> <p>Просмотрщик изображений с возможностью масштабирования и поворота.</p> <p>Приложение для создания простых векторных иллюстраций.</p>
Тема 12 «Цветовые модели»	<p>Основы цветовых моделей:</p> <p>Что такое цветовая модель?</p> <p>Понятие цветового пространства.</p> <p>Основные цветовые модели: RGB, CMYK, HSV, HSL.</p>

	<p>Модель RGB:</p> <p>Представление цвета как комбинации красного, зеленого и синего цветов.</p> <p>8-битное и 16-битное представление цвета в RGB.</p> <p>Работа с цветовыми каналами.</p> <p>Модель CMYK:</p> <p>Представление цвета как комбинации голубого, пурпурного, желтого и черного цветов.</p> <p>Использование CMYK в полиграфии.</p> <p>Преобразование между RGB и CMYK.</p> <p>Модель HSV:</p> <p>Представление цвета как комбинации оттенка, насыщенности и яркости.</p> <p>Интуитивно понятное управление цветом.</p> <p>Преобразование между RGB и HSV.</p> <p>Модель HSL:</p> <p>Представление цвета как комбинации оттенка, насыщенности и светлоты.</p> <p>Сходство с HSV.</p> <p>Преобразование между RGB и HSL.</p> <p>Работа с цветом в C#:</p> <p>Класс Color для представления цвета в RGB.</p> <p>Библиотека System.Drawing.Color для работы с цветами в графических приложениях.</p> <p>Преобразование между различными цветовыми моделями.</p> <p>Примеры использования:</p> <p>Разработка графических интерфейсов с использованием палитр цветов.</p> <p>Обработка изображений и применение цветовых фильтров.</p> <p>Создание игр и 3D-приложений.</p>
--	--

Курс дисциплины «Объектно-ориентированное программирование» предусматривает следующие формы организации учебного процесса:

1. лекции;
2. лабораторные занятия;
3. самостоятельная работа студента.

По источнику передачи и восприятия учебной информации используются словесные (лекция, беседа), наглядные (слайды, иллюстрации, коды программ), практические (исследования, упражнения, лабораторные работы) методы.

По характеру познавательной деятельности студентов используются объяснительно-иллюстративные и репродуктивные методы, проблемное преподавание, частично-поисковый и исследовательский методы.

В зависимости от основной дидактической цели и задач используются методы устного изложения знаний, закрепление учебного материала, самостоятельной работы студентов по осмыслению и усвоению нового материала, работы по применению знаний на практике и выработке умений и навыков, проверки и оценки знаний, умений и навыков.

Используются следующие методы контроля:

1. устный контроль (экспресс-опрос на лекциях);
2. проверка конспектов;
3. защита лабораторных работ;
4. проверка самостоятельных работ;
5. модульные контрольные работы;
6. итоговый контроль (экзаменационные билеты).

## 6. СТРУКТУРА И СОДЕРЖАНИЕ ДИСЦИПЛИНЫ

### 6.1. Форма обучения – очная, курс – 3, семестр – 5

Наименования разделов и тем	Количество часов				
	Лекц.	Лабор.	Практ.	СРС+К	Всего
<b>Содержательный модуль 1.</b>					
<b>Основы объектно-ориентированного программирования</b>					
Тема 1 «Основы языка C#»	2	3	0	5	10
Тема 2 «Создание графического приложения»	2	3	0	5	10
Тема 3 «Элементы управления»	2	3	0	5	10
Тема 4 «Объектно-ориентированное программирование. Общие принципы»	2	3	0	5	10
Тема 5 «Работа с файловой системой»	2	3	0	5	10
Тема 6 «Сборка мусора, управление памятью и указатели»	2	3	0	5	10
<b>ИТОГО Содержательный модуль 1</b>	<b>12</b>	<b>18</b>	<b>0</b>	<b>30</b>	<b>60</b>
<b>Содержательный модуль 2.</b>					
<b>Дополнительные главы объектно-ориентированного программирования</b>					
Тема 7 «Основы паттернов проектирования»	3	4	0	6	13
Тема 8 «Принципы SOLID»	3	4	0	5	12
Тема 9 «Введение в базы данных»	2	4	0	5	11
Тема 10 «Введение в сетевое программирование»	2	3	0	5	10
Тема 11 «Основы рисования в C#»	2	3	0	5	10
Тема 12 «Цветовые модели»	2	3	0	5	10
<b>ИТОГО Содержательный модуль 2</b>	<b>14</b>	<b>21</b>	<b>0</b>	<b>31</b>	<b>66</b>
<b>ИТОГО ЗА СЕМЕСТР</b>	<b>26</b>	<b>39</b>	<b>0</b>	<b>61</b>	<b>126</b>

## 7. ОЦЕНОЧНЫЕ МАТЕРИАЛЫ (СРЕДСТВА) ДЛЯ ТЕКУЩЕГО КОНТРОЛЯ УСПЕВАЕМОСТИ, ПРОМЕЖУТОЧНОЙ АТТЕСТАЦИИ

### 7.1 Контрольные вопросы

#### 1. Основы ООП:

1. Что такое объектно-ориентированное программирование (ООП)?
2. Какие основные принципы ООП?
3. Что такое класс и объект в ООП?
4. Как происходит инкапсуляция данных в C#?

5. Что такое наследование и полиморфизм в C#?
6. Как реализовать абстракцию в C#?
7. Каковы преимущества использования ООП?
8. В каких случаях целесообразно использовать ООП?

## **2. Работа с классами и объектами:**

1. Как объявить класс в C#?
2. Как создать объект класса?
3. Как определить поля и свойства класса?
4. Как реализовать методы класса?
5. Как использовать модификаторы доступа (public, private, protected)?
6. Как использовать статические члены класса?
7. Как работать с конструкторами и деструкторами классов?
8. Как реализовать наследование классов?
9. Как использовать виртуальные и переопределяемые методы?
10. Как реализовать интерфейсы в C#?
11. Как использовать полиморфизм в C#?
12. Как работать с абстрактными классами?

## **3. Работа с коллекциями:**

1. Какие типы коллекций доступны в C#?
2. Как работать с массивами в C#?
3. Как работать со списками (List<T>) в C#?
4. Как работать с словарями (Dictionary<TKey, TValue>) в C#?
5. Как работать с хеш-таблицами (HashSet<T>) в C#?
6. Как работать с очередями (Queue<T>) и стеками (Stack<T>) в C#?
7. Как использовать LINQ (Language Integrated Query) для работы с коллекциями?

## **4. Работа с файлами и текстом:**

1. Как работать с файлами в C#?
2. Как читать и записывать текстовые файлы?
3. Как работать с двоичными файлами?
4. Как использовать сериализацию объектов в C#?

## **5. Работа с исключениями:**

1. Что такое исключение в C#?
2. Как обрабатывать исключения в C#?
3. Как использовать операторы try-catch-finally?
4. Как генерировать собственные исключения?

## **6. Работа с потоками:**

1. Что такое поток в C#?
2. Как работать с синхронными и асинхронными потоками?
3. Как использовать задачи (Task) и асинхронные методы в C#?
4. Как обеспечить потокобезопасность в C#?

## 7. Дополнительные темы:

1. Как работать с делегатами и событиями в C#?
2. Как использовать отражение (Reflection) в C#?
3. Как работать с атрибутами в C#?
4. Как тестировать C#-код?
5. Какие инструменты и библиотеки используются для разработки C#-приложений?
6. Как развертывать C#-приложения?

### 7.2 Темы письменных работ (типы задач)

#### 1. Задача "Телефонная книга":

Создайте класс Contact для хранения информации о контакте (имя, фамилия, телефонный номер, адрес электронной почты). Реализуйте методы добавления, удаления, поиска и редактирования контактов. Создайте приложение, которое позволяет пользователю добавлять, удалять, искать и редактировать контакты в телефонной книге.

#### 2. Задача "Шахматная доска":

Создайте класс ChessBoard для представления шахматной доски. Реализуйте методы размещения фигур на доске, проверки допустимости ходов и определения победителя. Создайте приложение, которое позволяет пользователю играть в шахматы друг против друга или против компьютера.

#### 3. Задача "Банковский счет":

Создайте класс BankAccount для представления банковского счета. Реализуйте методы внесения и снятия средств, проверки баланса счета, перевода средств на другие счета. Создайте приложение, которое позволяет пользователю открывать новый счет, вносить и снимать средства, переводить средства на другие счета, просматривать историю операций.

#### 4. Задача "Склад товаров":

Создайте класс Product для представления товара (название, цена, количество на складе). Реализуйте методы добавления, удаления, поиска и редактирования товаров. Создайте приложение, которое позволяет пользователю добавлять, удалять, искать и редактировать товары на складе, а также просматривать информацию о наличии товаров.

#### 5. Задача "Библиотека книг":

Создайте класс Book для представления книги (название, автор, жанр, количество страниц). Реализуйте методы добавления, удаления, поиска и редактирования книг. Создайте приложение, которое позволяет пользователю добавлять, удалять, искать и редактировать книги в библиотеке, а также просматривать информацию о книгах.

#### 6. Задача "Расписание занятий":

Создайте класс Lesson для представления занятия (название предмета, время проведения, аудитория). Реализуйте методы добавления, удаления, поиска и редактирования занятий. Создайте приложение, которое позволяет пользователю создавать расписание занятий на неделю, добавлять, удалять, искать и редактировать занятия, а также просматривать расписание.

### **7. Задача "Чат-клиент/сервер":**

Создайте простое приложение чата, где один пользователь (сервер) может принимать сообщения от нескольких пользователей (клиентов). Используйте сокеты для сетевого взаимодействия.

### **8. Задача "Просмотр изображений":**

Создайте приложение для просмотра изображений. Пользователь должен иметь возможность загружать изображения из локального хранилища, просматривать их, увеличивать и уменьшать, а также переходить к следующему/предыдущему изображению.

### **9. Задача "Рисовалка":**

Создайте простое приложение для рисования. Пользователь должен иметь возможность рисовать линии, фигуры (прямоугольники, круги) с помощью мыши или клавиатуры, а также менять цвет кисти.

### **10. Задача "Игра в угадайку":**

Создайте игру "Угадай число". Компьютер загадывает число, а пользователь должен угадать его, делая попытки. После каждой попытки компьютер сообщает, больше или меньше загаданное число введенного пользователем.

## **7.3 Темы докладов (рефератов)**

### **1. Основы ООП:**

- Обзор принципов ООП: инкапсуляция, наследование, полиморфизм, абстракция.
- Преимущества использования ООП.
- Сравнение ООП с другими парадигмами программирования.

### **2. Работа с классами и объектами:**

- Определение класса в C#.
- Создание и использование объектов класса.
- Поля, свойства и методы классов.
- Модификаторы доступа (public, private, protected).
- Статические члены класса.
- Конструкторы и деструкторы.

### **3. Наследование и полиморфизм:**

- Определение иерархии классов.
- Виды наследования: одноуровневое, многоуровневое, иерархическое, множественное.
- Виртуальные и переопределяемые методы.
- Абстрактные классы и интерфейсы.
- Полиморфизм в действии: позднее связывание, метод-заместитель.

#### **4. Работа с коллекциями:**

- Типы коллекций в C#: массивы, списки, словари, хеш-таблицы, очереди, стеки.
- Операции с коллекциями: добавление, удаление, поиск, сортировка.
- LINQ (Language Integrated Query) для работы с коллекциями.

#### **5. Работа с файлами и текстом:**

- Чтение и запись текстовых файлов.
- Работа с двоичными файлами.
- Сериализация объектов в C#.

#### **6. Работа с исключениями:**

- Обработка исключений в C#.
- Операторы try-catch-finally.
- Генерирование собственных исключений.

#### **7. Работа с потоками:**

- Синхронные и асинхронные потоки.
- Задачи (Task) и асинхронные методы.
- Обеспечение потокобезопасности в C#.

#### **8. Дополнительные темы:**

- Делегаты и события.
- Отражение (Reflection).
- Атрибуты.
- Тестирование C#-кода.
- Инструменты и библиотеки для разработки C#-приложений.
- Развертывание C#-приложений.

#### **9. Прикладные задачи:**

- Разработка телефонной книги.
- Создание шахматной игры.
- Реализация системы управления банковскими счетами.
- Разработка приложения для складского учета.
- Создание библиотечного каталога.
- Планировщик занятий.
- Чат-клиент/сервер.
- Просмотрщик изображений.
- Приложение для рисования.

- Игра "Угадай число".

#### **10. Перспективные направления:**

- ООП в веб-разработке (ASP.NET, ASP.NET MVC).
- ООП в разработке мобильных приложений (Xamarin).
- ООП в разработке игр (Unity).
- ООП в высоконагруженных системах.

#### **7.4 Образец содержания экзаменационного билета**

**1. Опишите принцип инкапсуляции в C# и приведите пример его реализации.**

**2. Объясните, что такое наследование классов в C#, и какие его преимущества.**

**3. Практическое задание: Создайте класс Student для хранения информации о студенте (имя, фамилия, номер группы, курс, список оценок).**

- Реализуйте методы добавления, удаления и поиска оценок.
- Добавьте метод, который вычисляет среднюю оценку студента.
- Создайте экземпляр класса Student и заполните его данными.
- Выведите на экран информацию о студенте, включая его среднюю оценку.

В случае ведения учебного процесса с использованием электронного обучения и дистанционных образовательных технологий, содержание билета может отличаться от приведенного.

#### **8. РАСПРЕДЕЛЕНИЕ БАЛЛОВ, КОТОРЫЕ ПОЛУЧАЮТ ОБУЧАЮЩИЕСЯ**

Общая оценка знаний обучающихся по дисциплине проводится по 100-балльной шкале исходя из максимума, приведенного в таблице ниже.

Организационно-учебная работа в аудитории оценивается на основе таких критериев как посещаемость занятий, своевременное и качественное выполнение домашних заданий, активность во время проведения лекционных и практических занятий (участие в обсуждении текущего и пройденного материала, решение задач и т.п.).

Самостоятельная работа оценивается на основе предоставленных на проверку выполненных домашних, индивидуальных заданий с учетом своевременности их предоставления и соответствия требованиям к их выполнению.

Количество баллов за контрольную работу вычисляется как сумма баллов за все входящие в её состав задания. Каждое задание оценивается исходя из максимально возможного количества баллов с учетом правильности выполнения задания, полноты приводимых обоснований.

По результатам работы в семестре обучающийся, набравший не менее 60 баллов, имеет право получить оценку. Те, кто претендует на более высокий балл, проходят промежуточную аттестацию. Максимальное количество баллов на промежуточной аттестации – 100. Общее количество баллов за семестр вычисляется как максимальная из полученных за семестр и на промежуточной аттестации и выставляется согласно принятому порядку.

## 8.1. Семестр 5

Номера разделов	Виды работ	Максимальное количество баллов
1	Организационно-учебная работа в аудитории	15
	Самостоятельная работа	10
	Контрольные работы по практике	5
	Контрольная работа по теоретическому материалу	5
	Доклад/реферат	5
	<b>Итого по 1 разделу</b>	40
2	Организационно-учебная работа в аудитории	15
	Самостоятельная работа	15
	Контрольная работа по теоретическому материалу	10
	<b>Итого по 2 разделу</b>	40
<b>ИТОГО</b>		80
<b>Экзамен</b>		20
<b>Общий итог за семестр</b>		100

## Соответствие баллов оценке

Количество баллов из 100	ECTS	Оценка по пятибалльной шкале	
		Экзамен, дифференцированный зачет	Зачет
90-100	A	отлично	зачтено
80-89	B	хорошо	зачтено
75-79	C		зачтено
70-74	D	удовлетворительно	зачтено
60-69	E		зачтено
35-59	FX	неудовлетворительно	не зачтено
0-34	F		не зачтено

## 9. МАТЕРИАЛЬНО-ТЕХНИЧЕСКОЕ ОБЕСПЕЧЕНИЕ УЧЕБНОГО ПРОЦЕССА

Учебные занятия проводятся в Главном корпусе ДонГУ (г. Донецк, пр. Гурова, 6), в Учебно-практическом вычислительном центре ДонГУ (г. Донецк, пр. Гурова, 6, корпус 12).

Для проведения лекций требуется аудитория, оборудованная меловой или маркерной доской / сенсорным экраном / мультимедийный проектор с экраном и ноутбуком, комплект учебной мебели для студентов, рабочее место преподавателя.

Для проведения практических занятий требуется аудитория, оборудованная меловой или маркерной доской / сенсорным экраном / мультимедийный проектор с экраном и ноутбук, комплект учебной мебели для студентов, рабочее место преподавателя.

Для проведения лабораторных занятий требуется аудитория, оборудованная маркерной доской или сенсорным экраном / мультимедийный проектор с экраном и ноутбук, персональные компьютеры, комплект учебной мебели для студентов, рабочее место преподавателя, выход в Интернет – проводной или с использованием Wi-Fi.

Для самостоятельной работы используются текстовые и электронные ресурсы Научной библиотеки университета и других электронных библиотечных баз данных, учебно-методическое обеспечение, представленное в аудиториях Главного корпуса (ауд. 511, 605, 610).

Обучающиеся имеют возможность использовать учебные материалы по дисциплине, размещенные на платформе Moodle Центра дистанционного образования ФГБОУ ВО «ДонГУ». При изучении дисциплины применяются электронное обучение и дистанционные образовательные технологии.

С использованием ресурсов платформы дистанционного образования осуществляется текущий контроль знаний обучающихся на основе тестирования и проверки результатов самостоятельной работы.

## 10. РЕКОМЕНДУЕМАЯ ЛИТЕРАТУРА

### Основная литература:

1. Биллиг, В.А. Основы программирования на C#: учебный курс / В.А. Биллиг. – Москва: Интернет-Университет Информационных Технологий, 2006. – 485 с.: ил. – (Основы информационных технологий). – Режим доступа: по подписке. – URL: <http://biblioclub.ru/index.php?page=book&id=233695>. – ISBN 5-94774-401-5. – Текст: электронный.

2. Котов, О.М. Язык C#: краткое описание и введение в технологии программирования: учебное пособие / О.М. Котов; Министерство образования и науки Российской Федерации, Уральский федеральный университет им. первого Президента России Б. Н. Ельцина. – Екатеринбург: Издательство Уральского университета, 2014. – 209 с.: ил., табл., схем. – Режим доступа: по подписке. – URL: <http://biblioclub.ru/index.php?page=book&id=275809>. – Библиогр. в кн. – ISBN 978-5-7996-1094-4. – Текст: электронный.

3. Разработка приложений на C# с использованием СУБД PostgreSQL: учебное пособие / И.А. Васюткина, Г.В. Трошина, М.И. Бычков, С.А. Менжулин; Министерство образования и науки Российской Федерации, Новосибирский государственный технический университет. – Новосибирск: Новосибирский государственный технический университет, 2015. – 143 с.: схем., табл., ил. – Режим доступа: по подписке. – URL: <http://biblioclub.ru/index.php?page=book&id=438432>. – Библиогр. в кн. – ISBN 978-5-7782-2699-9. – Текст: электронный.

### Дополнительная литература:

1. Агуров, П. В. C#. Сборник рецептов [Электронный ресурс] / П. В. Агуров. - СПб.: БХВ-Петербург, 2007. - 432 с.: ил. - ISBN 5-94157-969 <http://znanium.com/bookread.php?book=489414>

2. Зиборов В. В. Visual C# 2010 на примерах.? СПб.: БХВ-Петербург, 2011.? 423 с. - ISBN 978-5-9775-0698-4. <http://www.znanium.com/bookread.php?book=355304>

3. Робисон, У. C# без лишних слов [Электронный ресурс] / У. Робинсон; Пер. с англ. - М.: ДМК Пресс, 2008. - 352 с.: ил. - (Серия "Для программистов"). - ISBN 5-94074-177-0. <http://znanium.com/bookread.php?book=408655>

## 11. ИНФОРМАЦИОННЫЕ РЕСУРСЫ

1. **Национальная электронная библиотека (НЭБ):** федеральная государственная информационная система / Министерство Культуры РФ; Российская государственная библиотека. – Москва, 2019- . – URL: <https://rusneb.ru/> (дата обращения: 01.09.2023). – Режим доступа: свободный, подписка. Необходима установка программного обеспечения. – Текст: электронный.

2. **eLIBRARY.RU:** научная электронная библиотека: сайт. – Москва, 2000- . – URL: <https://elibrary.ru> (дата обращения: 01.09.2023). – Режим доступа: для авторизов. пользователей. – Текст: электронный.

3. Научная электронная библиотека **«КиберЛенинка»**: сайт / Ассоциация «Открытая наука». – Москва, 2014- . – URL: <https://cyberleninka.ru/>. – Режим доступа: свободный. – Текст: электронный.
4. Электронно-библиотечная система **«Лань»**: [сайт]. – URL: <https://e.lanbook.com> (дата обращения: 01.09.2023). – Режим доступа: для авторизов. пользователей. – Текст: электронный.
5. **ЭБС Юрайт**: электронная библиотечная система: сайт. – Москва, 2013. – URL: <https://biblio-online.ru> (дата обращения: 01.09.2023). – Режим доступа: для авторизов. пользователей. – Текст: электронный.
6. **Электронно-библиотечная система ДонГУ**: сайт / ФГБОУ ВО «ДонГУ». – Донецк, 2016- . – URL: <http://library.donnu.ru/> (дата обращения: 01.09.2023). – Режим доступа: свободный. – Текст: электронный.
7. **Электронный каталог** Научной библиотеки ДонГУ: раздел сайта / НБ ДонГУ. – Текст: электронный // ЭБС ДонГУ: сайт. – URL: <http://library.donnu.ru/catalog/> (дата обращения: 01.09.2023). – Режим доступа: поиск свободный, электронные документы – для пользователей ДонГУ.
8. **Электронный архив ДонГУ**: раздел сайта / НБ ДонГУ. – Текст: электронный // ЭБС ДонГУ: сайт. – URL: <http://repo.donnu.ru/> (дата обращения: 01.09.2023). – Режим доступа: свободный.

## 12. ПРОГРАММНОЕ ОБЕСПЕЧЕНИЕ

1. Windows 7 PRO (корпоративная лицензия ДонГУ № 46484614)
2. Microsoft Office (корпоративная лицензия ДонГУ № 46472919)
3. Microsoft Visual Studio (лицензия программы Dream Spark для высших учебных заведений)
4. Антивирус Касперского, Adobe Acrobat Reader, xPDF (лицензии GPL, Apache, BSD для свободного программного обеспечения).
5. IDE Visual Studio Community (версии 2017, 2019).