МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ
ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ
«МОСКОВСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ ТЕХНОЛОГИЙ И УПРАВЛЕНИЯ ИМЕНИ К.Г. РАЗУМОВСКОГО (ПЕРВЫЙ КАЗАЧИЙ УНИВЕРСИТЕТ)»
Смоленский казачий институт промышленных технологий и бизнеса (филиал)

УТВЕРЖДАЮ
Председатель ПЦК
_____Максаков С. А.
«10» апреля 2026 г.

Рабочая программа дисциплины
ОП.03 Основы алгоритмизации и программирования

Закреплена за **ПЦК общепрофессиональных дисциплин; профессиональных модулей специальности 10.02.05 Обеспечение информационной безопасности автоматизированных систем**

Специальность: **10.02.05 ОБЕСПЕЧЕНИЕ ИНФОРМАЦИОННОЙ БЕЗОПАСНОСТИ АВТОМАТИЗИРОВАННЫХ СИСТЕМ**

Направленность

на базе основного общего образования

Квалификация **техник по защите информации**

Форма обучения **очная**

Общая трудоемкость **174 часов**

Часов по учебному плану 174

в том числе:

контактная работа 152

самостоятельная работа 14

часов на контроль 6

Виды контроля в семестрах:

Зачет с оценкой - 3

Экзамен - 4

Распределение часов дисциплины по семестрам

Вид учебной работы	Трудоемкость, часов					
	семестр 3		**семестр 4**		Итого	
	УП	РП	УП	РП	УП	РП
Лекции	32	32	44	44	76	76
Лабораторные	32	32	44	44	76	76
Контактная работа	64	64	88	88	152	152
Сам. работа	4	4	10	10	14	14
Часы на контроль			6	6	6	6

ДОКУМЕНТ ПОДПИСАН ЭЛЕКТРОННОЙ ПОДПИСЬЮ
Сертификат: 00F5 665F CCDB 5046 05DD 6E08 EBCB D00A DC
Действителен: с 21 октября 2025 г. по 14 января 2027 г.
Владелец: Коржикова Надежда Сергеевна
Должность: Директор

Москва, 2026

Программу составил:
Преподаватель Максаков Сергей Анатольевич . _____

Рабочая программа дисциплины
Основы алгоритмизации и программирования

Разработана в соответствии с:
Федеральный государственный образовательный стандарт среднего профессионального образования по специальности 10.02.05 ОБЕСПЕЧЕНИЕ ИНФОРМАЦИОННОЙ БЕЗОПАСНОСТИ АВТОМАТИЗИРОВАННЫХ СИСТЕМ (приказ Минобрнауки России от 09.12.2016 г. № 1553)
06.033. Профессиональный стандарт "СПЕЦИАЛИСТ ПО ЗАЩИТЕ ИНФОРМАЦИИ В АВТОМАТИЗИРОВАННЫХ СИСТЕМАХ", утверждённый приказом Министерства труда и социальной защиты Российско Федерации от 14 сентября 2022 г. N 525н (зарегистрирован Министерством юстиции Российской Федерации 14 октября 2022 г., регистрационный N 70543)

Разработана на основании учебного плана, утвержденного Учёным советом ФГБОУ ВО "МГУТУ им. К.Г. Разумовского (ПКУ)" от 26.03.2026 протокол №9.

СОДЕРЖАНИЕ

1. МЕСТО ДИСЦИПЛИНЫ (УЧЕБНОГО ПРЕДМЕТА) В СТРУКТУРЕ ОБРАЗОВАТЕЛЬНОЙ ПРОГРАММЫ

2. ЦЕЛИ, ЗАДАЧИ И ПЛАНИРУЕМЫЕ РЕЗУЛЬТАТЫ ОСВОЕНИЯ ДИСЦИПЛИНЫ (УЧЕБНОГО ПРЕДМЕТА)

3. СТРУКТУРА И СОДЕРЖАНИЕ ДИСЦИПЛИНЫ (УЧЕБНОГО ПРЕДМЕТА)

4. ФОНД ОЦЕНОЧНЫХ СРЕДСТВ ДИСЦИПЛИНЫ (УЧЕБНОГО ПРЕДМЕТА)

5. УЧЕБНО-МЕТОДИЧЕСКОЕ И ИНФОРМАЦИОННОЕ ОБЕСПЕЧЕНИЕ ДИСЦИПЛИНЫ (УЧЕБНОГО ПРЕДМЕТА)

6. МАТЕРИАЛЬНО-ТЕХНИЧЕСКОЕ ОБЕСПЕЧЕНИЕ ДИСЦИПЛИНЫ (УЧЕБНОГО ПРЕДМЕТА)

7. МЕТОДИЧЕСКИЕ УКАЗАНИЯ ДЛЯ ОБУЧАЮЩИХСЯ ПО ДИСЦИПЛИНЕ (УЧЕБНОГО ПРЕДМЕТА)

8. ОРГАНИЗАЦИЯ ОБРАЗОВАТЕЛЬНОГО ПРОЦЕССА ДЛЯ ЛИЦ С ОГРАНИЧЕННЫМИ ВОЗМОЖНОСТЯМИ ЗДОРОВЬЯ И ИНВАЛИДОВ

1. МЕСТО ДИСЦИПЛИНЫ В СТРУКТУРЕ ОБРАЗОВАТЕЛЬНОЙ ПРОГРАММЫ	
Цикл (раздел) ОП:	ОП

Дисциплина Основы алгоритмизации и программирования является частью неопознанный объект и обязательна для изучения.

Требования к предварительной подготовке обучающегося:	
1	Индивидуальный проект (Информатика)
2	Информатика
3	Профильные учебные предметы
Дисциплины (модули) и практики, для которых освоение данной дисциплины необходимо как предшествующее:	
1	Защита информации в автоматизированных системах программными и программно-аппаратными средствами
2	Программные и программно-аппаратные средства защиты информации
3	Учебная практика (защита информации в автоматизированных системах программными и программно-аппаратными средствами)
4	Производственная практика (защита информации в автоматизированных системах программными и программно-аппаратными средствами)
5	Экзамен по модулю (защита информации в автоматизированных системах программными и программно-аппаратными средствами)

2. ЦЕЛИ, ЗАДАЧИ И ПЛАНИРУЕМЫЕ РЕЗУЛЬТАТЫ ОСВОЕНИЯ ДИСЦИПЛИНЫ (УЧЕБНОГО ПРЕДМЕТА)

Цели освоения: Формирование у студентов фундаментальных знаний, навыков и практической компетенции в области алгоритмизации и разработки алгоритмов для решения разнообразных задач
Задачи: 1. Ознакомление со структурой и принципами работы алгоритмов;
2. Понимание различных видов представлений алгоритмов и их применения;
3. Формирование умения анализировать задачи и выбирать оптимальные алгоритмические решения;
4. Развитие умения программировать и реализовывать алгоритмы на практике;
5. Применение полученных знаний и навыков для решения задач из различных областей
Изучение дисциплины направлено на формирование у обучающихся следующих компетенций:

ОК 02.: Использовать современные средства поиска, анализа и интерпретации информации и информационные технологии для выполнения задач профессиональной деятельности
ОК 02.3: Знать номенклатуру информационных источников, применяемых в профессиональной деятельности; приемы структурирования информации; формат оформления результатов поиска информации, современные средства и устройства информатизации; порядок их применения и программное обеспечение в профессиональной деятельности, в том числе с использованием цифровых средств
ОК 02.4: Уметь определять задачи для поиска информации; определять необходимые источники информации; планировать процесс поиска; структурировать получаемую информацию; выделять наиболее значимое в перечне информации; оценивать практическую значимость результатов поиска; оформлять результаты поиска, применять средства информационных технологий для решения профессиональных задач; использовать современное программное обеспечение; использовать различные цифровые средства для решения профессиональных задач
ПК 2.1.: Осуществлять установку и настройку отдельных программных, программно-аппаратных средств защиты информации
ПК 2.1.1: Знать особенности и способы применения программных и программно-аппаратных средств защиты информации, в том числе, в операционных системах, компьютерных сетях, базах данных
ПК 2.1.2: Уметь устанавливать, настраивать, применять программные и программно-аппаратные средства защиты информации
ПК 2.1.3: Иметь практический опыт установки, настройки программных средств защиты информации в автоматизированной системе

3. СТРУКТУРА И СОДЕРЖАНИЕ ДИСЦИПЛИНЫ

Код занятия	Наименование разделов, тем и содержание занятий /вид занятия/	Семестр / Курс	Часов/ в том числе	Компетен-ции	Виды контроля
	1. Основные принципы алгоритмизации и программирования				
1. 1	Тема 1.1. Основные понятия алгоритмизации Понятие алгоритма и его свойства. Виды алгоритмов. Способы описания алгоритмов. Базовые алгоритмические структуры: линейные, разветвляющиеся, циклические. Базовые типы данных и их характеристика. Разработка алгоритмов сложной структуры: метод	3	2/0	ОК 02.3,ПК 2.1.1	Устный опрос

	последовательной детализации, сборочный метод. Основы алгебры логики. Логические операции и логические функции Знать: понятие алгоритма и его свойства; виды алгоритмов; способы представления алгоритмов; базовые алгоритмические структуры; базовые типы данных; базовые логические операции /Лек/				
1. 2	Лабораторная работа №1. Представление алгоритмов в графическом и табличном виде Уметь: читать и анализировать алгоритмы, представленные в графической (блок-схемы) и табличной форме; преобразовывать текстовое описание алгоритма в графическое (блок-схему) и табличное представление Владеть: программными средствами для представления алгоритмов в графическом и табличном виде; стандартами и ГОСТами при построении блок-схем (например, ГОСТ 19.701-90) /Лаб/	3	4/0	ОК 02.4,ПК 2.1.2,ПК 2.1.3	Отчет по лабораторной работе
1. 3	Тема 1.2. Языки и системы программирования Классификация языков программирования. Поколения языков программирования. Способы реализации языков программирования. Интегрированная среда программирования Знать: классификацию языков программирования (по уровню абстракции, по парадигме программирования, по назначению); поколения языков программирования (с 1 по 5 поколение); способы реализации языков программирования (компилируемые, интерпретируемые, гибридные); понятие интегрированной среды разработки (IDE) и их виды /Лек/	3	2/0	ОК 02.3,ПК 2.1.1	Устный опрос
1. 4	Тема 1.3. Парадигмы программирования Этапы разработки программ. Принципы структурного программирования. Понятия основных элементов ООП: объекты, классы, методы. Свойства ООП: наследование, инкапсуляция, полиморфизм. Концепции модульного программирования Знать: основные парадигмы программирования (императивное, структурное, объектно-ориентированное, функциональное, логическое, процедурное, событийно-ориентированное); этапы разработки программ; принципы структурного программирования; основные элементы и принципы объектно-ориентированного программирования; концепции модульного программирования /Лек/	3	4/0	ОК 02.3,ПК 2.1.1	Устный опрос
1. 5	Самостоятельная работа по теме "Основные парадигмы программирования" /СР/	3	2/0	ОК 02.3,ОК 02.4,ПК 2.1.1,ПК 2.1.2,ПК 2.1.3	Вопросы для самоподготовки
1. 6	Тема 1.4. Принципы отладки и тестового контроля Понятие отладки. Понятие тестового контроля и набора тестов. Проверка граничных условий, ветвей алгоритма, ошибочных исходных данных. Функциональное и структурное тестирование Знать: понятие отладки; этапы отладки; инструменты отладки; понятие тестового контроля; виды тестирования (функциональное,	3	2/0	ОК 02.3,ПК 2.1.1	Устный опрос

	структурное); методы тестирования (черный ящик, белый щик, серый ящик); документирование тестирования /Лек/				
1. 7	Самостоятельная работа по теме "Виды и методы тестирования" /СР/	3	2/0	ОК 02.3,ОК 02.4,ПК 2.1.1,ПК 2.1.2,ПК 2.1.3	Вопросы для самоподготовки
	2. Языки программирования				
2. 1	Тема 2.1. Характеристика языка C# История и особенности языка C#. Области применения языка C#. Характеристика среды разработки Visual Studio. Знать: историю и особенности языка C#; области применения языка C#; характеристику среды разработки Visual Studio /Лек/	3	2/0	ОК 02.3,ПК 2.1.1	Устный опрос
2. 2	Тема 2.2. Элементы языка. Простые типы данных Алфавит и лексика языка. Идентификаторы и ключевые слова. Структура программы. Типы данных языка программирования. Переменные и их описания. Константы. Операции с переменными и константами. Класс Random и его функции. Организация ввода/вывода данных Знать: алфавит и лексика языка C#; идентификаторы и ключевые слова; структуру программы на C#; типы данных языка C#; переменные и их описание; константы; операции с переменными и константами; функции класса Random; организацию ввода/вывода данных; /Лек/	3	4/0	ОК 02.3,ПК 2.1.1	Устный опрос
2. 3	Лабораторная работа №2. Разработка консольных приложений на ввод и вывод данных. Класс Random и Math Уметь: реализовывать консольный ввод/вывод; форматировать вывод; генерировать псевдослучайные числа в заданном диапазоне; применять математические функции класса Math Владеть: средой разработки для создания консольных приложений с интерактивным вводом -выводом; методами классов Random и Math /Лаб/	3	2/0	ОК 02.4,ПК 2.1.2,ПК 2.1.3	Отчет по лабораторной работе
2. 4	Тема 2.3. Базовые конструкции структурного программирования Организация ветвлений. Операторы циклов (с предусловием, с постусловием, с параметром). Операторы передачи управления Знать: определение и принципы структурного программирования; виды базовых структур (линейная, ветвление, циклы) и их синтаксис; операторы управления (break, continue, return) и их синтаксис /Лек/	3	4/0	ОК 02.3,ПК 2.1.1	Устный опрос
2. 5	Лабораторная работа №3. Разработка консольных приложений с применением условного оператора и оператора выбора Уметь: применять ветвления для решения реальных задач, выбирая оптимальные конструкции	3	4/0	ОК 02.4,ПК 2.1.2,ПК 2.1.3	Отчет по лабораторной работе

	Владеть: синтаксисом условного оператора и оператора выбора для разработки приложений /Лаб/				
2. 6	Лабораторная работа №4. Разработка консольных приложений с применением операторов цикла Уметь: применять операторы циклов для решения реальных задач, выбирая оптимальные конструкции Владеть: синтаксисом циклического оператора для разработки приложений /Лаб/	3	4/0	ОК 02.4,ПК 2.1.2,ПК 2.1.3	Отчет по лабораторной работе
2. 7	Тема 2.4. Работа с массивами и указателями. Структурные типы данных Массивы, общие сведения. Одномерные массивы. Многомерные массивы. Сортировка массивов. Работа с символами. Работа со строками. Структуры. Перечисления Знать: определение массива и указателей; виды массивов; операции над массивами; виды сортировок массивов; определение символа и строка; отличие символа от строк; /Лек/	3	4/0	ОК 02.3,ПК 2.1.1	Устный опрос
2. 8	Лабораторная работа №5. Разработка консольных приложений по обработке массивов Уметь: отличать одномерные и двумерные массивы; выполнять сортировку массивов; выполнять основные операции по работе с массивами (заполнять, выводить на экран, вычислять сумму и т.п., находить элементы по заданному критерию) Владеть: средой программирования для разработки приложений по обработке массивов (одномерных и двумерных) /Лаб/	3	6/0	ОК 02.4,ПК 2.1.2,ПК 2.1.3	Отчет по лабораторной работе
2. 9	Лабораторная работа №6. Разработка консольных приложений по обработке строк Уметь: символы от строк; применять встроенные методы по обработке строк Владеть: средой программирования для разработки приложений по обработке строк /Лаб/	3	4/0	ОК 02.4,ПК 2.1.2,ПК 2.1.3	Отчет по лабораторной работе
2. 10	Тема 2.5. Процедуры и функции Определение процедур и функций. Области видимости. Глобальные и локальные переменные. Обращение к процедурам и функциям. Использование библиотечных функций. Рекурсивное определение функций. Шаблоны функций Знать: понятие методы; отличие процедуры от функции; виды переменных (локальные и глобальные) и их отличие; способы обращения к методам; понятие рекурсии /Лек/	3	4/0	ОК 02.3,ПК 2.1.1	Устный опрос
2. 11	Лабораторная работа №7. Разработка консольных приложений по созданию собственных методов Уметь: отличать функции от процедур; выбирать функцию или метод при решении реальных задач Владеть: средой программирования для разработки приложений по созданию собственных подпрограмм (методов) /Лаб/	3	4/0	ОК 02.4,ПК 2.1.2,ПК 2.1.3	Отчет по лабораторной работе
2. 12	Тема 2.6. Работа с файлами Файловый ввод/вывод. Организация обмена данными между программой и внешними устройствами компьютера. Ввод и вывод текстовой информации. Неформатированный ввод/вывод данных. Дополнительные операции с	3	4/0	ОК 02.3,ПК 2.1.1	Устный опрос

	файлами Знать: понятие файла; отличие типизированного файла от нетипизированного; методы при работе с файлами /Лек/				
2. 13	Лабораторная работа №8. Разработка консольных приложений по работе с файлами Уметь: применять различные методы по обработке файлов при решении реальных задач Владеть: средой программирования для разработки приложений по работе с файлами /Лаб/	3	4/0	ОК 02.4,ПК 2.1.2,ПК 2.1.3	Отчет по лабораторной работе
2. 14	Другие формы контроля. ОК 02.3 Знать номенклатуру информационных источников, применяемых в профессиональной деятельности; приемы структурирования информации; формат оформления результатов поиска информации, современные средства и устройства информатизации; порядок их применения и программное обеспечение в профессиональной деятельности, в том числе с использованием цифровых средств ОК 02.4 Уметь определять задачи для поиска информации; определять необходимые источники информации; планировать процесс поиска; структурировать получаемую информацию; выделять наиболее значимое в перечне информации; оценивать практическую значимость результатов поиска; оформлять результаты поиска, применять средства информационных технологий для решения профессиональных задач; использовать современное программное обеспечение; использовать различные цифровые средства для решения профессиональных задач ПК 2.1.1 Знать особенности и способы применения программных и программно-аппаратных средств защиты информации, в том числе, в операционных системах, компьютерных сетях, базах данных ПК 2.1.2 Уметь устанавливать, настраивать, применять программные и программно-аппаратные средства защиты информации ПК 2.1.3 Иметь практический опыт установки, настройки программных средств защиты информации в автоматизированной системе /ЗаО/	3	0/0	ОК 02.3,ОК 02.4,ПК 2.1.1,ПК 2.1.2,ПК 2.1.3	Вопросы для зачета с оценкой, Итоговое тестирование
	3. Основы объектно-ориентированного программирования				
3. 1	Тема 3.1. Класс - как механизм создания объектов Понятия: класс, объект, свойства объекта, методы. Синтаксис объявления класса. Описание объектов. Спецификаторы доступа (private, public, protected). Описание функций-членов класса. Принцип инкапсуляции Знать: понятие класса; синтаксис описания класса; понятие экземпляра класса; спецификаторы доступа /Лек/	4	10/0	ОК 02.3,ПК 2.1.1	Устный опрос
3. 2	Самостоятельная работа по теме «Структура класса в объектно-ориентированном программировании» /СР/	4	4/0	ОК 02.3,ОК 02.4,ПК 2.1.1,ПК 2.1.2,ПК 2.1.3	Вопросы для самоподготовки
3. 3	Тема 3.2. Принципы наследования и	4	8/0	ОК	Устный опрос

	полиморфизма Механизм наследования для формирования иерархии классов. Формат объявления класса потомка. Режим доступа. Примеры организации классов-наследников Знать: определение наследования и его цели; синтаксис объявления класса-потомка; режимы доступа /Лек/			02.3,ПК 2.1.1	
3. 4	Самостоятельная работа по теме «Наследование и полиморфизм в объектно-ориентированном программировании» /СР/	4	2/0	ОК 02.3,ОК 02.4,ПК 2.1.1,ПК 2.1.2,ПК 2.1.3	Вопросы для самоподготовки
3. 5	Тема 3.3. Понятия деструктора и конструктора Назначение и свойства конструкторов, деструкторов. Их описание. Вызов в программе конструкторов, деструкторов. Примеры программ с конструкторами и деструкторами Знать: определение конструкторов и деструкторов; когда они вызываются /Лек/	4	8/0	ОК 02.3,ПК 2.1.1	Устный опрос
3. 6	Лабораторная работа №9. Разработка приложений по работе с классами Уметь: описывать классы; создавать объекты и их вызывать Владеть: средой программирования для разработки приложений по работе с классами /Лаб/	4	14/0	ОК 02.4,ПК 2.1.2,ПК 2.1.3	Отчет по лабораторной работе
	4. Модульное программирование				
4. 1	Тема 4.1. Понятие модульного программирования Модульное программирование как метод разработки программ. Программный модуль и его основные характеристики. Типовая структура программного модуля. Инкапсуляция в модулях. Порядок разработки программного модуля. Связность модулей. Ошибки периода исполнения и логические ошибки в программах. Обработка ошибок. Исключительные ситуации. Организация обработки исключительных ситуаций Знать: определение и цели модульного программирования; структуру и характеристики модуля; принципы инкапсуляции; этапы разработки модуля; типы связности модулей; классификацию ошибок и механизмы обработки ошибок и исключений /Лек/	4	10/0	ОК 02.3,ПК 2.1.1	Устный опрос
4. 2	Самостоятельная работа по теме «Обработка исключительных ситуаций» /СР/	4	4/0	ОК 02.3,ОК 02.4,ПК 2.1.1,ПК 2.1.2,ПК 2.1.3	Вопросы для самоподготовки
4. 3	Лабораторная работа №10. Модульное программирование. Исключения Уметь: создавать программы с применением модульного программирования; обрабатывать исключения Владеть: средой программирования для обработки исключений /Лаб/	4	8/0	ОК 02.4,ПК 2.1.2,ПК 2.1.3	Отчет по лабораторной работе
4. 4	Тема 4.2. Разработка оконных приложений Среда разработки приложений. Архитектура оконных приложений. Конфигурации для создания консольных и оконных приложений.	4	8/0	ОК 02.3,ПК 2.1.1	Устный опрос

	Разработка приложений как многомодульного проекта Знать: инструменты разработки (функционал IDE, популярные среды (Visual Studio, PyCharm)); архитектуру GUI; типы приложений (консольные, оконные) /Лек/				
4. 5	Лабораторная работа №11. Разработка оконных приложений Уметь: применять разные среды программирования для создания оконных приложений Владеть: основными конструкциями языка программирования для создания оконных приложений в различных средах программирования /Лаб/	4	22/0	ОК 02.4,ПК 2.1.2,ПК 2.1.3	Отчет по лабораторной работе
4. 6	Консультация «Разбор ключевых тем и типовых задач». Организационные моменты проведения экзамена: формат экзамена; время проведения; разрешённые материалы, используемые студентом во время проведения экзамена; критерии оценивания. Перечисление основных, наиболее важных, тем на экзамене. Краткий обзор разделов, которые будут на экзамене: основные принципы алгоритмизации и программирования; языки программирования; основы объектно-ориентированного программирования; модульное программирование. Разбор типовых задач. Примеры задач с пошаговым решением. Ответы на вопросы студентов (студенты могут уточнить сложные моменты и попросить разобрать конкретные задачи) ОК 02.3 Знать номенклатуру информационных источников, применяемых в профессиональной деятельности; приемы структурирования информации; формат оформления результатов поиска информации, современные средства и устройства информатизации; порядок их применения и программное обеспечение в профессиональной деятельности, в том числе с использованием цифровых средств ОК 02.4 Уметь определять задачи для поиска информации; определять необходимые источники информации; планировать процесс поиска; структурировать получаемую информацию; выделять наиболее значимое в перечне информации; оценивать практическую значимость результатов поиска; оформлять результаты поиска, применять средства информационных технологий для решения профессиональных задач; использовать современное программное обеспечение; использовать различные цифровые средства для решения профессиональных задач ПК 2.1.1 Знать особенности и способы применения программных и программно-аппаратных средств защиты информации, в том числе, в операционных системах, компьютерных сетях, базах данных ПК 2.1.2 Уметь устанавливать, настраивать, применять программные и программно-аппаратные средства защиты информации ПК 2.1.3 Иметь практический опыт установки, настройки программных средств защиты информации в автоматизированной системе	4	2/0	ОК 02.3,ОК 02.4,ПК 2.1.1,ПК 2.1.2,ПК 2.1.3	Вопросы для самоподготовки

	/Конс/				
4. 7	Экзамен ОК 02.3 Знать номенклатуру информационных источников, применяемых в профессиональной деятельности; приемы структурирования информации; формат оформления результатов поиска информации, современные средства и устройства информатизации; порядок их применения и программное обеспечение в профессиональной деятельности, в том числе с использованием цифровых средств ОК 02.4 Уметь определять задачи для поиска информации; определять необходимые источники информации; планировать процесс поиска; структурировать получаемую информацию; выделять наиболее значимое в перечне информации; оценивать практическую значимость результатов поиска; оформлять результаты поиска, применять средства информационных технологий для решения профессиональных задач; использовать современное программное обеспечение; использовать различные цифровые средства для решения профессиональных задач ПК 2.1.1 Знать особенности и способы применения программных и программно-аппаратных средств защиты информации, в том числе, в операционных системах, компьютерных сетях, базах данных ПК 2.1.2 Уметь устанавливать, настраивать, применять программные и программно-аппаратные средства защиты информации ПК 2.1.3 Иметь практический опыт установки, настройки программных средств защиты информации в автоматизированной системе /Эк/	4	6/0	ОК 02.3,ОК 02.4,ПК 2.1.1,ПК 2.1.2,ПК 2.1.3	Вопросы к экзамену, итоговое тестирование

Список выбранных образовательный технологий:

Название	Описание
Информационные технологии	Личностно ориентированная технология, способ организации самостоятельной деятельности учащихся, направленный на решение задачи учебного проекта
Компьютерная технология обучения	Основана на использовании информационных технологий в учебном процессе. Реализация данной технологии осуществляется посредством компьютера и иных мультимедийных средств. Использование компьютерных технологий делает учебный процесс не только современным и познавательным, но интересным для обучающихся
Технология контекстного обучения	Контекстное обучение отражает тенденцию соединения обучения с будущей профессиональной деятельностью, интеграцию обучения, науки и производства. Основной единицей работы преподавателей и студентов становится здесь не порция информации, а ситуация в ее предметной и социальной определенности; деятельность обучающихся обретает черты, в которых проявляются
Технология поиска информации (Информационная технология)	Информационная технология неотделима от субъектов образовательной деятельности, она является определяющим фактором технологии работы с информацией, применяемой в образовательной практике
Технология организации самостоятельной работы	Организации самостоятельной работы учащихся на более высоком уровне может способствовать применение технологии проектного и проблемного обучения. Методы самостоятельного приобретения знаний основаны на использовании проблемного обучения

4. ФОНД ОЦЕНОЧНЫХ СРЕДСТВ ДИСЦИПЛИНЫ

Оценочные средства текущего контроля успеваемости:

ВОПРОСЫ ДЛЯ УСТНОГО ОПРОСА

РАЗДЕЛ 1. ОСНОВНЫЕ ПРИНЦИПЫ АЛГОРИТМИЗАЦИИ И ПРОГРАММИРОВАНИЯ

Тема 1.1. Основные понятия алгоритмизации
1. Что такое алгоритм с точки зрения формального определения в информатике?

2. Перечислите и кратко охарактеризуйте основные свойства алгоритма (дискретность, детерминированность, результативность, массовость, понятность).
3. Чем отличаются линейные, разветвляющиеся и циклические виды алгоритмов по характеру выполнения команд?
4. Какие существуют основные способы (формы) описания алгоритмов (словесный, графический, псевдокод, программный)?
5. Как на блок-схеме графически изображаются ветвления (полная и неполная форма условного оператора)?
6. В чем принципиальное отличие цикла с предусловием от цикла с постусловием с точки зрения минимального количества выполнений?
7. Какие базовые типы данных (числовые, символьные, логические) вы знаете, и каковы их основные характеристики (диапазон, допустимые операции)?
8. В чем суть метода последовательной детализации (нисходящего проектирования) при разработке алгоритмов сложной структуры?
9. Чем сборочный метод (восходящее проектирование) отличается от метода последовательной детализации?
10. Какие базовые логические операции входят в алгебру логики (конъюнкция, дизъюнкция, отрицание, импликация, эквивалентность)?
11. Чему равен результат логического умножения (конъюнкции) и логического сложения (дизъюнкции) для различных комбинаций входных значений?
12. Как с помощью таблиц истинности определить значение сложной логической функции?
13. Почему алгоритм не может быть реализован без использования переменных определенных типов данных?
14. Как связаны понятия «базовые алгоритмические структуры» и «свойство дискретности алгоритма»?
15. Приведите пример условия, где для принятия решения в разветвляющемся алгоритме потребуется использование сложной логической функции (с операциями И, ИЛИ, НЕ).

Тема 1.2. Языки и системы программирования
1. По каким основным признакам классифицируют языки программирования (например, машинно-зависимые и машинно-независимые, языки низкого и высокого уровня)?
2. Какие языки относятся к первому поколению языков программирования, и в чем заключалась главная сложность их использования?
3. Чем языки программирования второго поколения (ассемблеры) принципиально отличаются от машинных кодов первого поколения?
4. Какие языки программирования относят к третьему поколению, и какое ключевое нововведение (процедурность, структурированность) они привнесли?
5. Что характерно для языков программирования четвертого и пятого поколений (например, проблемно-ориентированные, визуальные, декларативные языки)?
6. В чем суть компиляции как способа реализации языка программирования, и что является результатом работы компилятора?
7. Как работает интерпретатор, и чем его работа принципиально отличается от работы компилятора с точки зрения скорости выполнения и многократного использования кода?
8. Что такое JIT-компиляция (Just-In-Time), и какие преимущества она дает по сравнению с классической компиляцией и чистой интерпретацией?
9. Какие существуют гибридные способы реализации языков программирования (например, компиляция в байт-код с последующей виртуальной машиной)?
10. Что входит в понятие «интегрированная среда программирования» (IDE), и какие обязательные компоненты она должна включать?
11. Какие дополнительные инструменты (отладчик, профилировщик, система контроля версий, автодополнение кода) обычно присутствуют в современных IDE?
12. Чем интегрированная среда программирования отличается от простого текстового редактора с подсветкой синтаксиса?
13. Почему языки низкого уровня (ассемблер) до сих пор используются в программировании, несмотря на наличие множества языков высокого уровня?
14. Как выбор способа реализации языка (компиляция или интерпретация) влияет на переносимость программы между разными операционными системами?
15. Какие языки программирования обычно реализуются с помощью интерпретаторов, а какие – с помощью компиляторов, и с чем связан такой выбор?

Тема 1.3. Парадигмы программирования
1. Какие основные этапы включает в себя процесс разработки программного обеспечения от постановки задачи до сопровождения готового продукта?
2. Чем отличается этап алгоритмизации от этапа кодирования при разработке программы?
3. Какие принципы лежат в основе структурного программирования (отказ от безусловных переходов, модульность, нисходящее проектирование)?
4. Почему в структурном программировании рекомендуется избегать использования оператора безусловного перехода, и чем его можно заменить?
5. Что такое «объект» в объектно-ориентированном программировании, и какими характеристиками (состояние, поведение, идентичность) он обладает?
6. Чем класс отличается от объекта, и можно ли использовать класс без создания экземпляров?
7. Что называется методом в ООП, и как метод связан с классом и объектом?

8. В чем сущность свойства инкапсуляции, и как она обеспечивает скрытие внутренней реализации объекта?
9. Как работает механизм наследования, и какое соотношение «родитель-потомок» он устанавливает между классами?
10. Что означает полиморфизм в объектно-ориентированном программировании, и как один и тот же метод может вести себя по-разному в разных классах?
11. Чем переопределение методов отличается от перегрузки методов с точки зрения полиморфизма?
12. Какие основные концепции модульного программирования вы знаете (разделение на модули, сокрытие деталей, четко определенные интерфейсы)?
13. Чем модульное программирование отличается от структурного программирования по способу организации кода?
14. Какие преимущества дает разбиение программы на независимо компилируемые модули при коллективной разработке и отладке?
15. Как связаны между собой три основные парадигмы программирования (структурная, объектно-ориентированная, модульная) в реальных языках программирования?

Тема 1.4. Принципы отладки и тестового контроля
1. Что понимают под отладкой программы, и чем отладка принципиально отличается от тестирования?
2. Какие основные этапы включает в себя процесс отладки (обнаружение ошибки, локализация, исправление, проверка)?
3. Что называется тестовым контролем, и какова главная цель составления набора тестов?
4. Почему невозможно провести исчерпывающее (полное) тестирование даже для относительно простой программы?
5. Что такое «проверка граничных условий», и почему именно граничные значения чаще всего являются источником ошибок?
6. Какие примеры граничных условий можно привести для целочисленной переменной, принимающей значения от 1 до 100?
7. Как осуществляется проверка всех ветвей алгоритма, и почему недостаточно протестировать только «счастливый путь»?
8. Что подразумевается под тестированием с ошибочными исходными данными, и зачем вводить в программу заведомо некорректные значения?
9. Чем функциональное тестирование («черный ящик») отличается от структурного тестирования («белый ящик») с точки зрения доступа к внутреннему устройству программы?
10. Какие критерии полноты тестирования используются при структурном тестировании (покрытие операторов, покрытие ветвей, покрытие условий)?
11. Для каких типов задач функциональное тестирование является предпочтительнее структурного, и почему?
12. Как связаны понятия «набор тестов» и «покрытие кода» при оценке качества тестирования?
13. Почему при тестировании важно проверять поведение программы не только на корректных, но и на предельно допустимых и запрещенных входных данных?
14. Что такое тестовое покрытие ветвей, и чем оно отличается от покрытия операторов?
15. Какие ошибки можно выявить с помощью структурного тестирования, которые могут остаться незамеченными при функциональном тестировании?

РАЗДЕЛ 2. ЯЗЫКИ ПРОГРАММИРОВАНИЯ

Тема 2.1. Характеристика языка C#
1. В каком году и какой компанией был создан язык программирования C#, и кто был ведущим разработчиком этого языка?
2. Какое историческое событие или потребность послужила толчком к созданию языка C# вместо дальнейшего развития Java или C++?
3. Какие основные особенности языка C# отличают его от языка C++?
4. Чем платформа .NET Framework (или .NET) принципиально отличается от виртуальной машины Java (JVM) с точки зрения исполнения кода?
5. Что такое управляемый код, и как сборщик мусора влияет на работу программ на C#?
6. В каких основных областях применения используется язык C#?
7. Почему язык C# стал одним из основных инструментов для разработки игр на движке Unity?
8. Какие возможности языка C# делают его удобным для разработки корпоративных приложений?
9. Что представляет собой среда разработки Visual Studio, и какие редакции существуют?
10. Какие основные компоненты входят в интерфейс Visual Studio?
11. Какие возможности предоставляет отладчик Visual Studio для отладки программ на C#?
12. Чем отличается Visual Studio от Visual Studio Code с точки зрения работы с проектами на C#?
13. Что такое IntelliSense в Visual Studio, и как эта функция помогает при написании кода на C#?
14. Какие дополнительные инструменты входят в состав Visual Studio?
15. Как система проектов и решений организована в Visual Studio для разработки на C#?

Тема 2.2. Элементы языка. Простые типы данных
1. Из каких символов состоит алфавит языка C#, и какие лексические элементы выделяются на основе этого алфавита?
2. Какие правила формирования идентификаторов существуют в C#, и какие символы запрещены для

использования в именах переменных?
3. Чем ключевые слова отличаются от идентификаторов, и можно ли использовать ключевые слова в качестве имен переменных?
4. Из каких обязательных и необязательных элементов состоит структура простейшей программы на C#?
5. Какова сигнатура метода Main в консольном приложении C#, и почему именно этот метод является точкой входа в программу?
6. Какие категории типов данных существуют в C#, и в чем принципиальное различие в их хранении в памяти?
7. Какие целочисленные типы данных входят в состав простых типов C#, и какой диапазон значений у типа int?
8. Чем типы с плавающей запятой float и double отличаются по точности и диапазону, и какой тип используется по умолчанию для вещественных литералов?
9. Какие способы объявления переменных существуют в C#, и чем отличается инициализация от простого объявления?
10. Какие виды констант можно использовать в C#, и в чем их различия?
11. Какие арифметические, инкрементальные и декрементальные операции можно применять к переменным и константам в C#?
12. Чем отличается префиксная форма инкремента (++x) от постфиксной (x++) при использовании в выражениях?
13. Как создать объект класса Random, и какие методы этого класса используются для генерации случайных чисел?
14. Как организовать ввод данных с клавиатуры в консольном приложении C# с использованием методов Console.Read() и Console.ReadLine(), и в чем их различие?
15. Какими методами класса Console осуществляется вывод данных, и как с помощью форматных строк управлять внешним видом выводимой информации?

Тема 2.3. Базовые конструкции структурного программирования
1. Какие формы оператора if существуют в C#, и как записывается сокращенная форма условного оператора (тернарная операция)?
2. Для каких целей используется оператор switch в C#, и чем он отличается от каскада вложенных операторов if-else?
3. Какие ограничения на тип выражения в операторе switch существуют в C# по сравнению с некоторыми другими языками программирования?
4. Как работает оператор цикла с предусловием while, и в каком случае тело цикла не будет выполнено ни разу?
5. Чем цикл с постусловием do-while отличается от цикла while, и сколько минимальное количество итераций гарантирует каждый из них?
6. Какой синтаксис имеет цикл с параметром for в C#, и какие три выражения указываются в его заголовке?
7. Каким образом можно организовать бесконечный цикл с помощью конструкции for, и для каких задач это может потребоваться?
8. Для чего предназначен оператор break, и как его применение влияет на выполнение вложенных циклов (выходит только из внутреннего или из всех сразу)?
9. Чем оператор continue отличается от оператора break, и какой эффект он производит при выполнении в теле цикла?
10. Какие операторы передачи управления существуют в C#, и в каких ситуациях применяется оператор goto, несмотря на критику структурного подхода?
11. Как работает оператор return в методах с типом возвращаемого значения void и в методах, возвращающих конкретное значение?
12. Можно ли использовать операторы break и continue внутри конструкции switch, и как они себя ведут в этом контексте?
13. Какие различия в работе цикла foreach по сравнению с циклом for, и для каких типов коллекций он предназначен?
14. Как организовать множественный выбор с помощью оператора switch, если несколько вариантов выбора должны выполнять один и тот же блок кода?
15. Чем отличается досрочный выход из цикла с помощью break от естественного завершения цикла при выполнении условия окончания?

Тема 2.4. Работа с массивами и указателями. Структурные типы данных
1. Что такое массив в C#, и какими свойствами обладают массивы как структурные типы данных?
2. Какими способами можно объявить, создать и инициализировать одномерный массив в C#?
3. Как осуществляется доступ к элементам одномерного массива, и что произойдет при попытке обратиться к индексу, выходящему за границы массива?
4. Чем отличаются прямоугольные многомерные массивы от ступенчатых массивов в C# по способу выделения памяти и синтаксису объявления?
5. Как объявить и инициализировать двумерный прямоугольный массив размерностью 3 строки на 4 столбца, и как обратиться к элементу, расположенному во второй строке и третьем столбце?
6. Какие алгоритмы сортировки массивов поддерживаются встроенными методами класса Array, и для каких типов данных они работают по умолчанию?
7. Как отсортировать массив строк в обратном алфавитном порядке с использованием методов класса Array?
8. Какими способами можно перебрать все элементы многомерного массива с помощью вложенных циклов, и как для этой цели можно использовать цикл foreach?
9. Как в C# представляются символы, и какими методами класса Char можно проверять и преобразовывать

символы?
10. Чем тип string в C# отличается от массива символов char[], и почему строки называют неизменяемыми?
11. Какие основные методы класса String используются для обработки строковых данных?
12. Как с помощью метода Split разбить строку, содержащую слова, разделенные запятыми и пробелами, на массив отдельных слов?
13. Что такое структура в C#, и чем структура принципиально отличается от класса с точки зрения хранения в памяти (стек против кучи)?
14. Как объявить перечисление в C#, и какие преимущества дает использование перечислений вместо числовых констант для представления набора связанных значений?
15. Почему в C# отсутствуют классические указатели в том виде, в котором они существуют в C/C++, и в каких контекстах их использование все же допускается?

Тема 2.5. Процедуры и функции
1. Чем в C# функция отличается от процедуры?
2. Из каких обязательных элементов состоит заголовок метода в C#?
3. Какими способами можно передать параметры в метод в C#?
4. Чем отличаются параметры, передаваемые с ключевым словом ref, от параметров, передаваемых с ключевым словом out, и в каких ситуациях применяется каждый из них?
5. Что такое область видимости переменной в C#, и какие правила определяют доступность переменной в различных частях программы?
6. Чем глобальные переменные (поля класса) отличаются от локальных переменных по времени жизни и области видимости?
7. Как организовать доступ к методу из другого класса, и какие модификаторы доступа регулируют этот доступ?
8. Что такое перегрузка методов, и по каким признакам компилятор различает перегруженные методы?
9. Какие библиотечные функции пространства имен System наиболее часто используются при разработке на C#, и приведите примеры их вызова?
10. Что такое рекурсивная функция, и какое обязательное условие должно присутствовать в любом рекурсивном определении во избежание бесконечного вызова?
11. В чем преимущества и недостатки рекурсивного подхода по сравнению с итерационным (циклическим) решением одной и той же задачи?
12. Что представляют собой шаблоны (обобщенные) методы в C#, и как с помощью синтаксиса <T> объявить метод, работающий с различными типами данных?
13. Как вызвать шаблонный метод, если тип данных не указан явно, а выводится компилятором автоматически?
14. Каким образом локальная переменная, объявленная внутри метода, может иметь то же имя, что и поле класса, и как внутри метода обратиться к полю класса?
15. Что такое стек вызовов, и как он связан с работой процедур и функций, особенно при рекурсивных вызовах?

Тема 2.6. Работа с файлами
1. Какие пространства имен в C# содержат основные классы для организации файлового ввода/вывода?
2. Какие основные классы в C# используются для чтения и записи текстовых файлов?
3. Чем отличается работа с файлами через классы File и FileInfo с точки зрения статических и экземплярных методов?
4. Как открыть существующий текстовый файл для чтения с помощью StreamReader, и какой метод используется для построчного считывания данных?
5. Как записать строку в текстовый файл с помощью StreamWriter, и почему важно вызывать метод Close() или использовать конструкцию using для освобождения ресурсов?
6. Что представляет собой неформатированный (бинарный) ввод/вывод, и какие классы используются для работы с двоичными файлами?
7. Как с помощью BinaryWriter записать в файл целое число, вещественное число и строку, и в каком виде эти данные будут сохранены?
8. Каким образом можно прочитать данные из бинарного файла, если неизвестна точная последовательность типов записанных значений?
9. Какие дополнительные операции с файлами поддерживаются классом File?
10. Для каких целей используются классы Directory и DirectoryInfo при работе с файловой системой?
11. Каким образом организовать обмен данными между программой и файлом, если файл имеет большой размер и не помещается целиком в оперативную память?
12. Как с помощью класса FileStream управлять позицией чтения/записи в файле?
13. Какие режимы открытия файла доступны при создании объекта FileStream, и чем они отличаются?
14. Что произойдет с данными в файле, если открыть его в режиме FileMode.Create, когда файл с указанным именем уже существует?
15. Как с помощью метода File.ReadAllLines или File.WriteAllLines выполнить чтение или запись всех строк текстового файла без использования явного цикла?

РАЗДЕЛ 3. ОСНОВЫ ОБЪЕКТНО-ОРИЕНТИРОВАННОГО ПРОГРАММИРОВАНИЯ

Тема 3.1. Класс - как механизм создания объектов
1. Что такое класс в объектно-ориентированном программировании, и какую роль он выполняет как шаблон или чертеж для создания объектов?

2. Чем объект (экземпляр класса) отличается от класса с точки зрения выделения памяти и хранения данных?
3. Что называется свойствами объекта (полями данных), и чем они отличаются от методов (функций-членов) класса?
4. Каков базовый синтаксис объявления класса в C# (ключевые слова class, фигурные скобки, тело класса)?
5. Как описать (создать) объект класса в C#, и какое ключевое слово используется для выделения памяти под экземпляр класса?
6. Какие спецификаторы доступа (модификаторы) существуют в C#, и что означает каждый из них?
7. В чем разница между спецификатором доступа private и public применительно к полям и методам класса?
8. Для каких целей используется спецификатор доступа protected, и чем он отличается от private с точки зрения наследования?
9. Как объявить и определить функцию-член (метод) внутри класса в C#, и чем ее объявление отличается от объявления обычной функции вне класса?
10. Что такое принцип инкапсуляции, и как он реализуется с помощью комбинации private-полей и public-методов (свойств) в классе?
11. Каким образом внешний код может получить доступ к private-полю класса, если прямой доступ к нему запрещен (через public-методы или свойства get/set)?
12. Что произойдет при попытке обратиться из другого класса к private-полю данного класса без использования public-методов доступа?
13. Как организовать доступ к методам и полям класса через объект (оператор точка), и какие элементы класса доступны, а какие нет в зависимости от спецификаторов?
14. Можно ли создать объект класса, в котором все поля объявлены как private, и как тогда инициализировать такой объект начальными значениями?
15. Чем отличается объявление класса от объявления структуры (struct) с точки зрения инкапсуляции и размещения в памяти (ссылочный тип против значимого типа)?

Тема 3.2. Принципы наследования и полиморфизма
1. Что представляет собой механизм наследования в объектно-ориентированном программировании, и для каких целей он используется при формировании иерархии классов?
2. Каков формат объявления класса-потомка (производного класса) в C#?
3. Какой класс в C# является базовым (родительским) для всех остальных классов, даже если это не указано явно?
4. Как наследуются члены класса с различными спецификаторами доступа при создании класса-потомка?
5. Почему private-поля и private-методы базового класса недоступны напрямую в классе-потомке, и как можно организовать доступ к ним (через protected-члены или public-свойства)?
6. В чем разница между спецификаторами protected и public для членов базового класса с точки зрения доступности в классе-потомке и во внешнем коде?
7. Как вызвать конструктор базового класса из конструктора производного класса с помощью ключевого слова base в C#?
8. Что такое полиморфизм в объектно-ориентированном программировании, и как он связан с возможностью использования объектов разных классов через единый интерфейс?
9. Какие ключевые слова используются в C# для реализации полиморфизма при переопределении методов базового класса?
10. Чем отличается переопределение метода от сокрытия метода в производном классе с точки зрения поведения при вызове через ссылку на базовый класс?
11. Что такое виртуальные методы, и почему базовый класс должен пометить метод как virtual, чтобы потомок мог его переопределить с помощью override?
12. Как организовать пример иерархии классов-наследников (например, базовый класс «Фигура» и производные классы «Круг», «Прямоугольник», «Треугольник») с переопределением метода вычисления площади?
13. Можно ли в C# реализовать множественное наследование классов (наследование от двух и более базовых классов), и если нет, то чем это ограничение компенсируется (интерфейсы)?
14. Какой доступ имеют protected-члены базового класса в классе-потомке, если оба класса находятся в разных сборках (разных проектах)?
15. Как с помощью полиморфизма организовать единый метод, который принимает параметр типа базового класса, но корректно вызывает переопределенные методы объектов любого производного класса?

Тема 3.3. Понятия деструктора и конструктора
1. Каково основное назначение конструктора в C#, и какие свойства его характеризуют?
2. Каково основное назначение деструктора (финализатора) в C#, и чем его роль отличается от роли конструктора?
3. Как правильно описать (объявить) конструктор в классе на C#, и может ли класс содержать несколько конструкторов с разными параметрами (перегрузка конструкторов)?
4. Как правильно описать деструктор (финализатор) в C#, и каков его синтаксис (имя с тильдой, отсутствие параметров, отсутствие модификаторов доступа)?
5. Когда и как происходит автоматический вызов конструктора при создании объекта класса в C#?
6. Когда и как происходит вызов деструктора (финализатора) в C#, и можно ли вызвать деструктор явно из программного кода?
7. Чем отличается конструктор по умолчанию (без параметров) от конструктора с параметрами, и что произойдет, если в классе не объявить ни одного конструктора?

8. Что такое цепочка вызовов конструкторов (использование ключевого слова this для вызова одного конструктора из другого), и для каких целей она применяется?
9. Как с помощью конструктора с параметрами обеспечить обязательную инициализацию полей объекта в момент его создания?
10. Почему в C# невозможно точно предсказать момент вызова деструктора (финализатора), и кто отвечает за его вызов (сборщик мусора, Garbage Collector)?
11. Как с помощью статического конструктора (static constructor) инициализировать статические поля класса, и когда он вызывается?
12. Какие ограничения накладываются на деструкторы в C#?
13. Приведите пример программы, где конструктор используется для выделения ресурсов (например, открытие файла), а деструктор – для их освобождения.
14. В чем разница между деструктором в C++ и финализатором в C# с точки зрения детерминированности времени вызова и управления памятью?
15. Что произойдет, если в классе объявить деструктор, но при этом не освобождать в нем никакие неуправляемые ресурсы, и как это повлияет на производительность?

РАЗДЕЛ 4. МОДУЛЬНОЕ ПРОГРАММИРОВАНИЕ

Тема 4.1. Понятие модульного программирования
1. Что понимают под модульным программированием как методом разработки программ, и в чем его основное отличие от монолитного подхода?
2. Что называется программным модулем, и какие основные характеристики используются для его описания?
3. Из каких типовых элементов состоит структура программного модуля в C#?
4. Как реализуется принцип инкапсуляции на уровне модулей, и какие механизмы C# обеспечивают скрытие внутренней реализации?
5. В чем разница между физической модульностью и логической модульностью в C#?
6. Каков оптимальный порядок разработки программного модуля: сначала спецификация интерфейса или сначала реализация внутренних функций, и почему?
7. Что такое связность модуля, и почему высокая связность (функциональная) считается желательной, а низкая (случайная) – нежелательной?
8. Какие виды связности модулей вы знаете, и в чем заключается худшая из них?
9. Чем отличаются ошибки периода исполнения от логических ошибок в программах с точки зрения проявления и обнаружения?
10. Почему логические ошибки считаются наиболее трудными для обнаружения и исправления по сравнению с синтаксическими ошибками и ошибками периода исполнения?
11. Что такое исключительная ситуация в программировании, и какие стандартные классы исключений существуют в C#?
12. Как организована обработка исключительных ситуаций с помощью конструкции try-catch-finally в C#, и для чего предназначен каждый из этих блоков?
13. В чем разница между перехватом исключения с помощью catch без указания типа и catch с указанием конкретного типа исключения (например, catch (FileNotFoundException ex))?
14. Для каких целей используется блок finally, и в чем его особенность?
15. Каким образом можно создать собственный класс исключения в C#, и в каких ситуациях это целесообразно сделать вместо использования стандартных исключений?

Тема 4.2. Разработка оконных приложений
1. Какие основные среды разработки в Visual Studio используются для создания оконных приложений на C#?
2. Чем Windows Forms отличается от WPF с точки зрения подхода к построению пользовательского интерфейса (императивный против декларативного)?
3. Какую архитектурную модель (обычно MVC, MVVM или событийно-ориентированную) используют оконные приложения, и в чем суть разделения кода на представление, логику и данные?
4. Как в Visual Studio настроить конфигурацию проекта для создания консольного приложени и для создания оконного приложения?
5. Какие различия в точке входа и в ссылочных сборках существуют между шаблонами консольного и оконного приложений?
6. Что представляет собой файл конструктора в Windows Forms, и как он связан с основным файлом кода формы?
7. Каким образом осуществляется многомодульная разработка оконного приложения, когда интерфейс, бизнес-логика и работа с данными вынесены в отдельные проекты (сборки)?
8. Как добавить ссылку на другой проект или сборку в решение Visual Studio, чтобы использовать классы и методы из одного модуля в другом оконном приложении?
9. Какая технология (XAML) используется для описания пользовательского интерфейса в WPF-приложениях, и чем XAML-разметка отличается от программного создания элементов управления в коде C#?
10. Как организована обработка событий в оконных приложениях на C#?
11. Какие основные контейнеры для размещения элементов управления существуют в Windows Forms и WPF?
12. Как осуществляется переход между несколькими окнами (формами) в оконном приложении, и какими методами управляют жизненным циклом окон?
13. В чем разница между модальным и немодальным открытием дочернего окна с точки зрения взаимодействия

пользователя с родительским окном?
14. Как в многомодульном проекте оконного приложения организовать доступ к общим ресурсам из разных модулей?
15. Какие инструменты визуального проектирования предоставляет Visual Studio для ускорения разработки пользовательского интерфейса оконных приложений?

ВОПРОСЫ ДЛЯ САМОПОДГОТОВКИ

РАЗДЕЛ 1. ОСНОВНЫЕ ПРИНЦИПЫ АЛГОРИТМИЗАЦИИ И ПРОГРАММИРОВАНИЯ

Самостоятельная работа по теме "Основные парадигмы программирования"
1. Какие три основные парадигмы программирования обычно выделяют в современной теории программирования, и в чем заключается суть каждой из них?
2. Какие принципы лежат в основе структурного программирования, и почему рекомендуется избегать использования оператора безусловного перехода (goto)?
3. Чем объектно-ориентированное программирование отличается от структурного с точки зрения организации данных и методов их обработки?
4. Что такое инкапсуляция, наследование и полиморфизм? Приведите краткую характеристику каждого из этих свойств ООП.
5. Какой этап разработки программы следует за этапом алгоритмизации, и что происходит на этапе кодирования?
6. Что такое модульное программирование, и какие преимущества дает разбиение программы на независимо разрабатываемые и компилируемые модули?
7. Чем отличается объект от класса в объектно-ориентированном программировании?
8. Как полиморфизм позволяет использовать единый интерфейс для работы с объектами разных классов?
9. Какие основные этапы разработки программного обеспечения вы знаете (от постановки задачи до сопровождения)?
10. Как связаны между собой понятия «класс» и «метод» в объектно-ориентированной парадигме?

Самостоятельная работа по теме "Виды и методы тестирования"
1. Что такое тестирование программного обеспечения, и какова его основная цель?
2. Чем функциональное тестирование (тестирование «черного ящика») отличается от структурного тестирования («белого ящика») с точки зрения доступа к внутреннему устройству программы?
3. Какие критерии полноты тестирования используются при структурном тестировании?
4. Почему при тестировании важно проверять поведение программы не только на корректных, но и на граничных и ошибочных исходных данных?
5. Какие типы ошибок можно выявить с помощью структурного тестирования, которые могут остаться незамеченными при чисто функциональном тестировании?
6. Что называется набором тестов, и какими основными требованиями он должен обладать?
7. Какие примеры граничных условий можно привести для целочисленной переменной, принимающей значения в диапазоне от -50 до 50?
8. Что такое покрытие ветвей, и почему оно считается более строгим критерием тестирования, чем покрытие операторов?
9. Для каких типов задач функциональное тестирование является предпочтительнее структурного, и с чем связан такой выбор?
10. Чем тестирование отличается от отладки по целям, исполнителям и временным рамкам выполнения?

РАЗДЕЛ 3. ОСНОВЫ ОБЪЕКТНО-ОРИЕНТИРОВАННОГО ПРОГРАММИРОВАНИЯ

Самостоятельная работа по теме «Структура класса в объектно-ориентированном программировании»
1. Чем тестирование программного обеспечения принципиально отличается от отладки с точки зрения целей и задач?
2. Какие два основных метода тестирования (по доступу к внутренней структуре программы) выделяют, и в чем заключается разница между «черным ящиком» и «белым ящиком»?
3. Что такое функциональное тестирование, и какие аспекты программы проверяются при его проведении без знания внутреннего устройства кода?
4. Что такое структурное тестирование, и какие критерии полноты тестирования (покрытие операторов, покрытие ветвей, покрытие условий) используются в его рамках?
5. Почему проверка граничных условий считается одним из наиболее эффективных способов выявления ошибок в программе?
6. Приведите не менее двух примеров граничных условий для целочисленной переменной, которая может принимать значения от 0 до 100.
7. Что означает термин «набор тестов», и какими требованиями должен обладать качественный набор тестов?
8. Почему при тестировании важно проверять поведение программы не только на корректных, но и на заведомо ошибочных исходных данных?
9. Что такое покрытие ветвей, и чем оно отличается от покрытия операторов с точки зрения качества тестирования?
10. Какие типы ошибок можно выявить с помощью структурного тестирования, которые могут остаться незамеченными при чисто функциональном тестировании?

Самостоятельная работа по теме "Структура класса в объектно-ориентированном программировании"
1. Из каких основных элементов (разделов) состоит структура класса в объектно-ориентированном программировании?
2. Что такое поля класса, и какие модификаторы доступа могут применяться к ним?
3. Чем методы класса отличаются от полей, и какую роль они выполняют в структуре класса?
4. Что такое конструктор класса, каковы его основные свойства (имя, отсутствие возвращаемого значения, автоматический вызов) и чем он отличается от обычного метода?
5. Какие виды конструкторов (конструктор по умолчанию, конструктор с параметрами, статический конструктор) могут присутствовать в структуре класса?
6. Для чего в структуре класса используются свойства с ключевыми словами get и set, и как они связаны с принципом инкапсуляции?
7. Какую функцию выполняет деструктор (финализатор) в структуре класса, и в каких случаях его объявляют явно?
8. Каким образом принцип инкапсуляции реализуется через структуру класса?
9. Что представляет собой сигнатура метода, и по каким признакам различаются перегруженные методы внутри класса?
10. Как организовано наследование в структуре класса, и какие члены базового класса становятся доступными производному классу в зависимости от спецификаторов доступа?

Самостоятельная работа по теме «Наследование и полиморфизм в объектно-ориентированном программировании»
1. Что такое наследование в объектно-ориентированном программировании, и какую роль оно играет в формировании иерархии классов?
2. Каков синтаксис объявления класса-потомка (производного класса) от базового (родительского) класса?
3. Каким образом наследуются члены класса с различными спецификаторами доступа при создании класса-потомка?
4. Чем отличается спецификатор доступа protected от private с точки зрения доступности членов класса в производных классах?
5. Что такое полиморфизм в объектно-ориентированном программировании, и как он позволяет использовать объекты разных классов через единый интерфейс?
6. Какие ключевые слова используются в C# для реализации полиморфизма при переопределении методов базового класса?
7. В чем разница между переопределением метода и сокрытием метода в производном классе?
8. Почему метод базового класса должен быть помечен ключевым словом virtual, чтобы производный класс мог его переопределить с помощью override?
9. Можно ли в C# реализовать множественное наследование классов (наследование от двух и более базовых классов), и если нет, то каким механизмом это ограничение компенсируется?
10. Как с помощью полиморфизма организовать метод, который принимает параметр типа базового класса, но корректно вызывает переопределенные методы объектов любого производного класса?

РАЗДЕЛ 4. МОДУЛЬНОЕ ПРОГРАММИРОВАНИЕ

Самостоятельная работа по теме «Обработка исключительных ситуаций»
1. Что называется исключительной ситуацией в программировании, и чем она отличается от синтаксической или логической ошибки?
2. Какие стандартные классы исключений (не менее трех) существуют в C#, и в каких ситуациях каждое из них возникает?
3. Как организована обработка исключительных ситуаций с помощью конструкции try-catch-finally в C#?
4. Для каких целей предназначен блок try, и какие операции следует размещать внутри этого блока?
5. Чем отличается блок catch без указания типа исключения от блока catch с указанием конкретного типа (например, catch (DivideByZeroException ex))?
6. Каким образом можно перехватить несколько разных типов исключений в одной конструкции try-catch, и в каком порядке должны следовать блоки catch?
7. Для каких целей используется блок finally, и в чем заключается его особенность?
8. Что произойдет, если исключение возникло в блоке try, но ни один из блоков catch не обработал этот тип исключения?
9. Как создать собственный класс исключения в C#, и от какого базового класса его следует наследовать?
10. В каких ситуациях целесообразно создавать и выбрасывать собственное исключение с помощью ключевого слова throw вместо использования стандартных классов исключений?

ВОПРОСОВ ДЛЯ САМОПОДГОТОВКИ (по разделам)

РАЗДЕЛ 1. ОСНОВНЫЕ ПРИНЦИПЫ АЛГОРИТМИЗАЦИИ И ПРОГРАММИРОВАНИЯ
1. Почему любая последовательность инструкций, даже если она приводит к результату, не всегда может считаться алгоритмом с точки зрения формальных свойств?
2. Какие три базовые алгоритмические структуры лежат в основе любого алгоритма, независимо от его сложности?
3. Чем массовость алгоритма отличается от результативности, и может ли алгоритм быть результативным, но не

обладать массовостью?
4. Каким образом форма записи алгоритма (словесная, графическая, программная) влияет на его однозначность понимания исполнителем?
5. Почему при описании разветвляющегося алгоритма важно явно указывать условие перехода, а не просто набор возможных действий?
6. Какие преимущества и недостатки имеет компиляция как способ реализации языка программирования по сравнению с интерпретацией?
7. Каким образом поколение языка программирования (1-е, 2-е, 3-е, 4-е) связано с его уровнем абстракции от машинного кода?
8. Почему в современных условиях сохраняют актуальность языки второго поколения (ассемблеры), несмотря на существование множества языков высокого уровня?
9. В чем заключается роль таблиц истинности при проверке корректности сложных логических условий в алгоритмах?
10. Какая разница между процедурной (императивной) и декларативной парадигмами программирования в подходе к описанию решения задачи?
11. Почему при тестовом контроле проверка граничных условий часто позволяет обнаружить больше ошибок, чем проверка типичных значений?
12. Каким образом функциональное тестирование («черный ящик») может быть полезно, если тестировщик не имеет доступа к исходному коду программы?
13. Чем локализация ошибки в процессе отладки отличается от её обнаружения, и какой из этих этапов обычно занимает больше времени?
14. Как связаны между собой понятия «дискретность алгоритма» и «возможность его исполнения компьютером в пошаговом режиме»?
15. Почему при разработке алгоритмов сложной структуры метод последовательной детализации считается более предпочтительным, чем сборочный, для начинающих программистов?

РАЗДЕЛ 2. ЯЗЫКИ ПРОГРАММИРОВАНИЯ
1. Каким образом платформа .NET и среда выполнения CLR (Common Language Runtime) обеспечивают независимость программ на C# от конкретной аппаратной платформы?
2. Почему в языке C# для хранения целых чисел используется несколько типов, а не один универсальный целочисленный тип?
3. Какие последствия могут возникнуть при неявном преобразовании типа long в int, и почему компилятор C# требует явного приведения в этом случае?
4. Чем отличается постфиксная форма инкремента (x++) от префиксной (++x) при использовании в составных арифметических выражениях?
5. Каким образом оператор switch в C# позволяет избежать каскада вложенных if-else, и какие ограничения на тип выражения в нем существуют?
6. Почему в цикле с предусловием while тело цикла может не выполниться ни разу, а в цикле с постусловием do-while выполняется как минимум один раз?
7. Какие особенности организации памяти (стек и куча) объясняют различное поведение значимых типов и ссылочных типов в C#?
8. Как с помощью метода Split класса String преобразовать строку с разделителями в массив подстрок, и какой символ чаще всего используется в качестве разделителя?
9. Чем отличаются прямоугольные многомерные массивы от ступенчатых массивов в C# с точки зрения выделения памяти и доступа к элементам?
10. Почему строки в C# являются неизменяемыми, и какие преимущества это дает с точки зрения безопасности и производительности?
11. Каким образом метод Main должен быть объявлен в консольном приложении C#, чтобы он мог принимать аргументы командной строки?
12. Что произойдет с потоком выполнения программы при вызове метода Console.ReadLine(), если пользователь не ввел никаких данных и не нажал клавишу Enter?
13. Как с помощью класса Random сгенерировать случайное целое число в диапазоне от 10 до 20 (включительно), используя метод Next()?
14. Почему при работе с файлами через StreamReader и StreamWriter рекомендуется использовать конструкцию using, а не явно вызывать метод Close()?
15. Каким образом классы File и FileInfo различаются по способу вызова методов (статические против экземплярных), и в каких случаях предпочтительнее каждый из них?

РАЗДЕЛ 3. ОСНОВЫ ОБЪЕКТНО-ОРИЕНТИРОВАННОГО ПРОГРАММИРОВАНИЯ
1. Почему класс в ООП называют не просто типом данных, а «шаблоном» или «чертежом» для создания объектов?
2. Каким образом спецификатор доступа private обеспечивает инкапсуляцию, если внешний код все равно может получить доступ к данным через public-методы?
3. Чем отличается состояние объекта от его поведения, и какие элементы класса отвечают за каждый из этих аспектов?
4. Почему конструктор не может возвращать значение, даже тип void, и каковы причины такого синтаксического ограничения?
5. Каким образом перегрузка конструкторов (несколько конструкторов с разными параметрами) повышает

гибкость создания объектов?
6. Что произойдет, если в классе объявить конструктор с параметрами, но не объявить конструктор по умолчанию, и попытаться создать объект без параметров?
7. Почему деструктор в C# нельзя вызвать явно, и кто определяет момент его вызова в программе?
8. Каким образом ключевое слово base используется в конструкторе производного класса для вызова конструктора базового класса?
9. Почему при наследовании private-поля базового класса не доступны в производном классе напрямую, но могут быть доступны через protected-свойства?
10. Каким образом полиморфизм позволяет вызывать переопределенный метод производного класса через ссылку на базовый класс?
11. Чем отличается переопределение метода от сокрытия метода, и как компилятор определяет, какой метод вызывать в каждом случае?
12. Почему виртуальные методы требуют явного указания ключевого слова, а не являются поведением по умолчанию для всех методов?
13. Какие проблемы могут возникнуть, если в классе объявить деструктор, но при этом оставить ссылки на управляемые ресурсы?
14. Как множественное наследование интерфейсов в C# компенсирует отсутствие множественного наследования классов?
15. Почему конструктор статического класса не может иметь параметров и модификаторов доступа, и когда он вызывается?

РАЗДЕЛ 4. МОДУЛЬНОЕ ПРОГРАММИРОВАНИЕ
1. Чем программный модуль отличается от просто набора функций, собранных в одном файле, с точки зрения независимой компиляции?
2. Как низкая связность модуля влияет на сложность его тестирования и повторного использования в других проектах?
3. Почему в модульном программировании считается хорошей практикой минимизировать количество глобальных переменных, доступных из разных модулей?
4. Каким образом в Visual Studio организована физическая модульность (проекты и сборки) и логическая модульность (пространства имен)?
5. Чем ошибка периода исполнения отличается от логической ошибки, и какой тип ошибки сложнее обнаружить с помощью тестового контроля?
6. Почему перехват всех исключений с помощью catch без указания типа считается плохой практикой в промышленном программировании?
7. Как блок finally гарантирует освобождение ресурсов (например, закрытие файла или соединения с базой данных) даже при возникновении исключения?
8. Каким образом архитектурные паттерны (MVC, MVVM) применяются при разработке оконных приложений для разделения логики и представления?
9. Чем модальное окно отличается от немодального с точки зрения блокировки взаимодействия с родительским окном?
10. Почему в WPF для описания пользовательского интерфейса используется декларативный язык XAML, а не программное создание элементов в C#?
11. Каким образом событийная модель оконных приложений (нажатие кнопки, загрузка формы) организует связь между действиями пользователя и кодом обработчиков?
12. Что произойдет с приложением Windows Forms, если в обработчике события (например, Click кнопки) возникнет необработанное исключение?
13. Как в многомодульном проекте организовать ссылку на общий модуль с бизнес-логикой из двух разных модулей интерфейса (консольного и оконного)?
14. Почему при разработке оконных приложений важно различать поток пользовательского интерфейса (UI thread) и фоновые рабочие потоки?
15. Как инструмент визуального проектирования (дизайнер форм) в Visual Studio автоматически генерирует код для размещенных на форме элементов управления?

Задания для лабораторных работ см. в приложение А.

Оценочные средства промежуточной аттестации:
См. приложение Б
Темы индивидуальных проектов, курсовых работ (проектов), индивидуальных заданий на практику:
Учебным планом не предусмотрено

Описание критериев оценивания успеваемости

НЕДОСТАТОЧНЫЙ УРОВЕНЬ	ПОРОГОВЫЙ УРОВЕНЬ	ПРОДВИНУТЫЙ УРОВЕНЬ	ВЫСОКИЙ УРОВЕНЬ
Обучающийся демонстрирует: - существенные пробелы в знаниях учебного материала; - принципиальные ошибки при ответе на основные вопросы билета, отсутствует знание и понимание основных понятий и категорий; - непонимание сущности дополнительных вопросов в рамках заданий билета; - отсутствие умения выполнять практические задания, предусмотренные программой дисциплины; - отсутствие готовности (способности) к дискуссии и низкая степень контактности.	Обучающийся демонстрирует: - знания теоретического материала; - неполные ответы на основные вопросы, ошибки в ответе, недостаточное понимание сущности излагаемых вопросов; - неуверенные и неточные ответы на дополнительные вопросы; - недостаточное владение литературой, рекомендованной программой дисциплины; - умение без грубых ошибок решать практические задания, которые следует выполнить.	Обучающийся демонстрирует: - знание и понимание основных вопросов контролируемого объема программного материала; - твердые знания теоретического материала; - способность устанавливать и объяснять связь практики и теории, выявлять противоречия, проблемы и тенденции развития; - правильные и конкретные, без грубых ошибок ответы на поставленные вопросы; - умение решать практические задания, которые следует выполнить; - владение основной литературой, рекомендованной программой дисциплины; - наличие собственной обоснованной позиции по обсуждаемым вопросам; - незначительные оговорки и неточности в раскрытии отдельных положений вопросов билета, присутствует неуверенность в ответах на дополнительные вопросы.	Обучающийся демонстрирует: - глубокие, всесторонние и аргументированные знания программного материала; - полное понимание сущности и взаимосвязи рассматриваемых процессов и явлений, точное знание основных понятий в рамках обсуждаемых заданий; - способность устанавливать и объяснять связь практики и теории; - логически последовательные, содержательные, конкретные и исчерпывающие ответы на все задания билета, а также дополнительные вопросы экзаменатора; - умение решать практические задания; - свободное использование в ответах на вопросы материалов рекомендованной основной и дополнительной литературы.
Оценка «незачет», «неудовлетворительно»	Оценка «зачтено/удовлетворительно» ,«удовлетворительно»	Оценка «зачтено/хорошо», «хорошо»	Оценка «зачтено/отлично», «отлично»

ОК 02.: Использовать современные средства поиска, анализа и интерпретации информации и информационные технологии для выполнения задач профессиональной деятельности

Недостаточный уровень:
Знает базовый перечень информационных источников (справочники, ГОСТы), простейшие приемы структурирования (списки, таблицы), единый формат оформления результатов (текстовый документ), основные средства информатизации (ПК, принтер) и порядок их применения, а также стандартное ПО (браузер, текстовый редактор) строго по инструкции

Умеет определять задачи и источники поиска по готовому шаблону, планировать последовательный перебор источников, структурировать информацию в заданную таблицу, выделять значимое по указанному критерию, оформлять результаты в виде стандартного текстового отчета и применять ПО (браузер, текстовый редактор) строго по инструкции

Пороговый уровень:
Знает расширенную номенклатуру профессиональных источников (нормативно-техническая документация, базы данных, электронные каталоги), алгоритмы структурирования (классификация, ранжирование), требования к оформлению результатов (отчет, справка, презентация), современные средства информатизации (МФУ, сканеры) и порядок их применения, а также возможности профессионального ПО (электронные таблицы, справочные системы) для типовых задач

Умеет самостоятельно формулировать задачи поиска, выбирать необходимые профессиональные источники (базы данных, справочные системы), планировать процесс с учетом сроков, структурировать информацию (группировка, ранжирование), выделять наиболее значимое для профессии, оценивать актуальность и достоверность результатов, оформлять их в виде аналитической справки или презентации, а также использовать стандартное ПО и цифровые средства (ПК, принтер, онлайн-сервисы) для решения типовых профессиональных задач

Продвинутый уровень:
Знает альтернативные номенклатуры источников (облачные хранилища, отраслевые порталы, экспертные сети), разные приемы структурирования (ментальные карты, дерево целей, хронологию), вариативные форматы оформления (дашборды, инфографика, базы данных), современные устройства информатизации (планшеты,

Знает альтернативные номенклатуры источников (облачные хранилища, отраслевые порталы, экспертные сети), разные приемы структурирования (ментальные карты, дерево целей, хронологию), вариативные форматы оформления (дашборды, инфографика, базы данных), современные устройства информатизации (планшеты, смарт-терминалы) и различные варианты их применения, а также умеет выбирать подходящее ПО в зависимости от профессиональной ситуации

Умеет вариативно определять задачи поиска под разные контексты (плановый, экстренный, углубленный анализ), выбирать оптимальные источники из альтернативных (официальные, экспертные, облачные хранилища), планировать маршруты поиска с резервными вариантами, применять разные приемы структурирования (ментальные карты, матрицы, хронологию), выделять значимое сравнительным анализом, оценивать практическую значимость результатов для конкретной задачи, выбирать формат оформления (дашборд, чек-лист, инфографика) и подбирать соответствующие цифровые средства под ситуацию

Высокий уровень:
Знает нерегламентированную совокупность источников (API открытых данных, семантические сети, экспертные интервью), интегративные приемы структурирования (динамические модели, многомерные классификации), инновационные форматы оформления (интерактивные отчеты, видеоаннотации, цифровые двойники), перспективные средства информатизации (IoT-устройства, системы технического зрения, промышленные дроны) и нестандартные порядки их применения, а также методы адаптации программного обеспечения (нейросети, SCADA-системы) под изменяющиеся профессиональные условия

Умеет адаптировать поиск в условиях неопределенности (дефицит времени, противоречивые данные), привлекать неочевидные источники (API, экспертные сети, сырые данные), гибко изменять стратегию поиска онлайн, структурировать разнородную и зашумленную информацию в авторские динамические модели, выделять значимое с учетом латентных факторов и рисков, оценивать применимость результатов для принятия решений в нестандартной ситуации, оформлять результаты для конкретной аудитории (интерактивный дашборд, база знаний, видеодемонстрация), а также творчески комбинировать различные цифровые средства и программное обеспечение (включая мобильные устройства, IoT, облачные сервисы) в единый рабочий процесс

ПК 2.1.: Осуществлять установку и настройку отдельных программных, программно-аппаратных средств защиты информации

Недостаточный уровень:
Знает основные виды угроз информации (несанкционированный доступ, вредоносное ПО, перехват данных), базовые понятия защиты информации (аутентификация, шифрование, контроль доступа), а также названия наиболее распространённых программных и программно-аппаратных средств защиты (антивирус, межсетевой экран, VPN).

Имеет практический опыт установки программных средств защиты информации в автоматизированной системе под руководством наставника.

Умеет под руководством преподавателя перечислять функции типового антивируса, находить в интерфейсе операционной системы настройки парольной защиты пользователя и запускать стандартную антивирусную проверку по образцу.

Пороговый уровень:
Имеет практический опыт самостоятельной установки и настройки программных средств защиты информации в автоматизированной системе по типовому регламенту.

Умеет самостоятельно настраивать стандартные средства защиты операционной системы (политики паролей, блокировка учётной записи после неудачных попыток, шифрование дисков встроенными средствами), выполнять базовые настройки персонального межсетевого экрана (разрешение/запрет входящих и исходящих соединений для конкретных приложений), применять антивирусное ПО для сканирования и карантина угроз, а также настраивать права доступа пользователей к таблицам в учебной базе данных (GRANT, REVOKE).

Знает классификацию программных и программно-аппаратных средств защиты информации (антивирусные комплексы, межсетевые экраны, системы обнаружения вторжений, средства криптографической защиты, средства контроля съёмных носителей), особенности их применения в операционных системах семейства Windows и Linux, базовые принципы разграничения доступа к файлам и папкам в файловых системах, а также основные методы защиты баз данных (аутентификация, разграничение прав на уровне таблиц и представлений, резервное копирование).

Продвинутый уровень:
Умеет самостоятельно анализировать существующую конфигурацию защиты информации в операционной системе, компьютерной сети или базе данных с целью выявления недостатков и уязвимостей; выполнять настройку расширенных политик безопасности (запрет выполнения кода из временных папок, ограничение прав приложений с использованием мандатного контроля); настраивать межсетевой экран для защиты серверной части базы данных (разрешение доступа только с определённых IP-адресов, инспектирование протокола); реализовывать шифрование столбцов или всей базы данных с использованием встроенных средств СУБД; организовывать защищённое подключение удалённых пользователей с использованием VPN и сертификатов.

Знает особенности применения различных типов программно-аппаратных средств защиты информации (аппаратные модули доверенной загрузки, криптошлюзы, сетевые экраны следующего поколения NGFW, средства защиты от утечек DLP) в зависимости от архитектуры сети, критичности данных и угроз; способы настройки политик безопасности в операционных системах (групповые политики, мандатный контроль доступа

Знает особенности применения различных типов программно-аппаратных средств защиты информации (аппаратные модули доверенной загрузки, криптошлюзы, сетевые экраны следующего поколения NGFW, средства защиты от утечек DLP) в зависимости от архитектуры сети, критичности данных и угроз; способы настройки политик безопасности в операционных системах (групповые политики, мандатный контроль доступа SELinux/AppArmor); методы защиты баз данных на уровне шифрования, маскирования данных и аудита запросов; а также регламентные мероприятия по обновлению и мониторингу работоспособности средств защиты.

Имеет практический опыт установки, настройки и отладки программных средств защиты информации в автоматизированной системе с учётом специфики угроз и архитектуры АС.

Высокий уровень:
Умеет самостоятельно проектировать архитектуру защиты информации для гетерогенной инфраструктуры, включающей различные операционные системы, сегментированные сети и распределённые базы данных (в том числе на основе облачных решений), с обоснованием выбора каждого средства; разрабатывать политики безопасности для операционных систем и баз данных с учётом специфики обрабатываемой информации и профилей угроз; выявлять и устранять конфликты совместной работы различных средств защиты (например, межсетевого экрана и системы обнаружения вторжений); проводить сравнительный анализ нескольких альтернатив программных или программно-аппаратных средств защиты для конкретной задачи с оценкой затрат, эффективности и эксплуатационных рисков.

Знает современные подходы к построению комплексных систем защиты информации (Zero Trust, концепция минимальных привилегий, защита на основе поведенческого анализа); особенности применения программно-аппаратных средств для защиты виртуальной инфраструктуры и контейнеров; методы интеграции различных средств защиты в единую систему с централизованным управлением и автоматизацией реагирования на инциденты (SOAR, SIEM); правовые и нормативные требования к применению средств защиты информации в государственных и коммерческих системах (аттестация объектов информатизации, сертификация средств защиты); а также перспективные направления развития технологий защиты (постквантовая криптография, аппаратная защита на уровне процессора).

Имеет практический опыт установки, настройки и оптимизации программных средств защиты информации в сложных автоматизированных системах, включая интеграцию разнородных СЗИ, устранение инцидентов, связанных с некорректной настройкой, и документирование конфигураций.

5. УЧЕБНО-МЕТОДИЧЕСКОЕ И ИНФОРМАЦИОННОЕ ОБЕСПЕЧЕНИЕ ДИСЦИПЛИНЫ

5.1. Рекомендуемая литература

5.1.1. Основная литература

Л.1.1	Никифоров С. Н. Прикладное программирование [Электронный ресурс]:. - Санкт-Петербург: Лань, 2022. - 124 с. – Режим доступа: https://e.lanbook.com/book/184156
Л.1.2	Батасова В. С., Гречкина П. В., Горкина А. А., Воробьева И. А., Маран М. М., Старостина Л. А., Чибизова Н. В. Программирование. Сборник задач [Электронный ресурс]:. - Санкт-Петербург: Лань, 2022. - 168 с. – Режим доступа: https://e.lanbook.com/book/189452
Л.1.3	Андрианова А. А., Исмагилов Л. Н., Мухтарова Т. М. Алгоритмизация и программирование. Практикум [Электронный ресурс]:. - Санкт-Петербург: Лань, 2022. - 240 с. – Режим доступа: https://e.lanbook.com/book/186390
Л.1.4	Андрианова А. А., Исмагилов Л. Н., Мухтарова Т. М. Алгоритмизация и программирование. Практикум [Электронный ресурс]:учебное пособие. - Санкт-Петербург: Лань, 2022. - 240 с. – Режим доступа: https://e.lanbook.com/book/206258
Л.1.5	Батасова В. С., Воробьева И. А., Голубева И. В., Гречкина П. В., Маран М. М., Чибизова Н. В., Маран М. М. Программирование в примерах и задачах [Электронный ресурс]:учебное пособие для вузов. - Санкт-Петербург: Лань, 2024. - 260 с. – Режим доступа: https://e.lanbook.com/book/362825
Л.1.6	Акулич И. Л. Математическое программирование в примерах и задачах [Электронный ресурс]:учебное пособие для вузов. - Санкт-Петербург: Лань, 2024. - 348 с. – Режим доступа: https://e.lanbook.com/book/360428
Л.1.7	Барков И. А. Объектно-ориентированное программирование [Электронный ресурс]:учебник для вузов. - Санкт-Петербург: Лань, 2023. - 700 с. – Режим доступа: https://e.lanbook.com/book/329549
Л.1.8	Рацеев С. М. Программирование. Лабораторный практикум [Электронный ресурс]:учебное пособие для спо. - Санкт-Петербург: Лань, 2023. - 104 с. – Режим доступа: https://e.lanbook.com/book/292904
Л.1.9	Рацеев С. М. Программирование. Лабораторный практикум [Электронный ресурс]:учебное пособие для спо. - Санкт-Петербург: Лань, 2026. - 104 с. – Режим доступа: https://e.lanbook.com/book/509356
Л.1.10	Андрианова А. А., Исмагилов Л. Н., Мухтарова Т. М. Алгоритмизация и программирование. Практикум [Электронный ресурс]:учебное пособие для спо. - Санкт-Петербург: Лань, 2026. - 240 с. – Режим доступа: https://e.lanbook.com/book/509336

5.2. Перечень программного обеспечения

Kaspersky Endpoint Security для бизнеса – Расширенный Russian Edition

5.3. Перечень информационных справочных систем

Сайт национального открытого университета "ИНТУИТ"
Федеральный центр информационно-образовательных ресурсов

Российский портал открытого образования
"Электронная библиотека учебников"
ПЛАТФОРМА ОНЛАЙН-ОБРАЗОВАНИЯ «РАЗУМ»

6. МАТЕРИАЛЬНО-ТЕХНИЧЕСКОЕ ОБЕСПЕЧЕНИЕ ДИСЦИПЛИНЫ

Адрес: 215119, Смоленская область, Вяземский район, город Вязьма, ул. Ленина, д. 77: аудитория 18-1302 - Лаборатория
Учебная аудитория для проведения занятий лекционного типа; занятий семинарского типа, лабораторных и практических занятий; для курсового проектирования (выполнения курсовых работ); для проведения групповых и индивидуальных консультаций; для текущего контроля и промежуточной аттестации : Рабочее место преподавателя; маркерная доска; 10 рабочих мест обучающихся, оснащенных ПЭВМ с подключением к сети интернет и обеспечением доступа в электронную информационно-образовательную среду;

Адрес: 215119, Смоленская область, Вяземский район, город Вязьма, ул. Ленина, д. 77: аудитория 18-1310 - Лаборатория
Учебная аудитория для проведения занятий лекционного типа; занятий семинарского типа, лабораторных и практических занятий; для курсового проектирования (выполнения курсовых работ); для проведения групповых и индивидуальных консультаций; для текущего контроля и промежуточной аттестации : Рабочие места обучающихся; Рабочее место преподавателя; Классная доска; 15 рабочих мест обучающихся оснащенные ПЭВМ с подключением к сети интернет и обеспечением доступа в электронную информационно-образовательную среду; Ноутбук переносной; Переносной проектор; Переносной экран; Учебно-наглядные пособия

7. МЕТОДИЧЕСКИЕ УКАЗАНИЯ ДЛЯ ОБУЧАЮЩИХСЯ ПО ДИСЦИПЛИНЕ

Методические рекомендации по организации самостоятельной работы

СРС – планируемая учебная, выполняемая во внеаудиторное (аудиторное) время по заданию и при методическом руководстве преподавателя, но без его непосредственного участия (возможно частичное непосредственное участие преподавателя при сохранении ведущей роли студентов). Целью СРС является овладение фундаментальными знаниями, профессиональными умениями и навыками по профилю будущей специальности, опытом творческой, исследовательской деятельности, развитие самостоятельности, ответственности и организованности, творческого подхода к решению проблем учебного и профессионального уровней. Задачи СРС: систематизация и закрепление полученных теоретических знаний и практических умений студентов; углубление и расширение теоретической подготовки; формирование умений использовать нормативную, правовую, справочную документацию и специальную литературу; развитие познавательных способностей и активности студентов: творческой инициативы, самостоятельности, ответственности и организованности; формирование самостоятельности мышления, способностей к саморазвитию, самосовершенствованию и самореализации; развитие исследовательских умений; использование материала, собранного и полученного в ходе самостоятельных занятий на практических занятиях, при написании курсовых и выпускной квалификационной работ, для эффективной подготовки к текущей и промежуточной аттестации. Функции СРС: развивающая (повышение культуры умственного труда, приобщение к 10 творческим видам деятельности, обогащение интеллектуальных способностей студентов); информационно-обучающая (учебная деятельность студентов на аудиторных занятиях, неподкрепленная самостоятельной работой, становится мало результативной); ориентирующая и стимулирующая (процессу обучения придается ускорение и мотивация); воспитательная (формируются и развиваются профессиональные качества специалиста и гражданина); исследовательская (новый уровень профессионально-творческого мышления).

Самостоятельная работа студентов является обязательным компонентом учебного процесса для каждого студента и определяется учебным планом. Самостоятельная работа определяется как индивидуальная или коллективная учебная деятельность, осуществляемая без непосредственного руководства педагога, но по его заданиям и под его контролем. Самостоятельная работа – это познавательная учебная деятельность, когда последовательность мышления студента, его умственных и практических операций и действий зависит и определяется самим студентом.

Самостоятельная работа студентов способствует развитию самостоятельности, ответственности и организованности, творческого подхода к решению проблем учебного и профессионального уровня, что в итоге приводит к развитию навыка самостоятельного планирования и реализации деятельности. Целью самостоятельной работы студентов является овладение необходимыми компетенциями по своему направлению подготовки, опытом творческой и исследовательской деятельности. На основании компетентностного подхода к реализации профессиональных образовательных программ, видами заданий для самостоятельной работы являются:

- для овладения знаниями: чтение текста (учебника, первоисточника, дополнительной литературы), составление плана текста, графическое изображение структуры текста, конспектирование текста, выписки из текста, работа со словарями и справочниками, ознакомление с нормативными документами, учебно-исследовательская работа, использование аудио- и видеозаписей, компьютерной техники и информационно-телекоммуникационной сети Интернет и др.
- для закрепления и систематизации знаний: работа с конспектом лекции, обработка текста (учебника, первоисточника, дополнительной литературы, аудио и видеозаписей), повторная работа над учебным материалом, составление плана, составление таблиц для систематизации учебного материала, ответ на контрольные вопросы, заполнение рабочей тетради, аналитическая обработка текста (аннотирование, рецензирование, реферирование, конспект-анализ и др.), завершение аудиторных практических работ и оформление отчётов по ним, подготовка мультимедиа сообщений/докладов к выступлению на семинаре (конференции), материалов-презентаций, подготовка реферата, составление библиографии, тематических кроссвордов, тестирование и др.
- для формирования умений: решение задач и упражнений по образцу, решение вариативных задач, выполнение чертежей, схем, выполнение расчетов (графических работ), решение ситуационных (профессиональных) задач, подготовка к деловым играм, проектирование и моделирование разных видов и компонентов профессиональной деятельности, рефлексивный анализ профессиональных умений с использованием аудио- и видеотехники и др.

Самостоятельная работа может осуществляться индивидуально или группами студентов в зависимости от цели, объема, конкретной тематики самостоятельной работы, уровня сложности, уровня умений студентов.

Методические рекомендации по освоению дисциплины (учебного предмета)

Методические рекомендации по подготовке к лекционным занятиям

Просмотрите конспект сразу после занятий. Пометьте материал конспекта лекций, который вызывает затруднения для понимания. Попытайтесь найти ответы на затруднительные вопросы, используя предлагаемую литературу. Если самостоятельно не удалось разобраться в материале, сформулируйте вопросы и обратитесь на текущей консультации или на ближайшей лекции за помощью к преподавателю. Каждую неделю рекомендуется отводить время для повторения пройденного материала, проверяя свои знания, умения и навыки по контрольным вопросам.
Работа с рекомендованной литературой:
При работе с основной и дополнительной литературой целесообразно придерживаться такой последовательности. Сначала прочитать весь заданный текст в быстром темпе. Цель такого чтения заключается в том, чтобы создать общее представление об изучаемом материале, понять общий смысл прочитанного. Затем прочитать вторично, более медленно, чтобы в ходе чтения понять и запомнить смысл каждой фразы, каждого положения и вопроса в целом. Чтение приносит пользу и становится продуктивным, когда сопровождается записями. Это может быть составление плана прочитанного текста, тезисы или выписки, конспектирование и др. Выбор вида записи зависит от характера изучаемого материала и целей работы с ним. Если содержание материала несложное, легко усваиваемое, можно ограничиться составлением плана. Если материал содержит новую и трудно усваиваемую информацию, целесообразно его законспектировать. План – это схема прочитанного материала, перечень вопросов, отражающих структуру и последовательность материала. Конспект – это систематизированное, логичное изложение материала источника. Различаются четыре типа конспектов:
- план-конспект – это развернутый детализированный план, в котором по наиболее сложным вопросам даются подробные пояснения,
- текстуальный конспект – это воспроизведение наиболее важных положений и фактов источника,
- свободный конспект – это четко и кратко изложенные основные положения в результате глубокого изучения материала, могут присутствовать выписки, цитаты, тезисы; часть материала может быть представлена планом,
- тематический конспект – составляется на основе изучения ряда источников и дает ответ по изучаемому вопросу. В процессе изучения материала источника и составления конспекта нужно обязательно применять различные выделения, подзаголовки, создавая блочную структуру конспекта. Это делает конспект легко воспринимаемым и удобным для работы.

Методические рекомендации по подготовке к практическим занятиям

Практические занятия представляют особую форму сочетания теории и практики. Их назначение – углубление проработки теоретического материала предмета путем регулярной и планомерной самостоятельной работы студентов на протяжении всего курса. Процесс подготовки к практическим занятиям включает изучение нормативных документов, обязательной и дополнительной литературы по рассматриваемому вопросу. Непосредственное проведение практического занятия предполагает, например:
- индивидуальные выступления студентов с сообщениями по какому-либо вопросу изучаемой темы;
- фронтальное обсуждение рассматриваемой проблемы, обобщения и выводы;
- решение задач и упражнений по образцу;
- решение вариантных задач и упражнений;
- решение ситуационных производственных (профессиональных) задач;
- проектирование и моделирование разных видов и компонентов профессиональной деятельности.
- выполнение контрольных работ;
- работу с тестами.

При подготовке к практическим занятиям студентам рекомендуется: внимательно ознакомиться с тематикой практического занятия; прочесть конспект лекции по теме, изучить рекомендованную литературу;
составить краткий план ответа на каждый вопрос практического занятия; проверить свои знания, отвечая на вопросы для самопроверки; если встретятся незнакомые термины, обязательно обратиться к словарю и зафиксировать их в тетради. Все письменные задания выполнять в рабочей тетради. Практические занятия развивают у студентов навыки самостоятельной работы по решению конкретных задач.

Методические рекомендации по подготовке к лабораторным работам

Лабораторные работы представляют одну из форм освоения теоретического материала с одновременным формированием практических навыков в изучаемой дисциплине. Их назначение – углубление проработки теоретического материала, формирование практических навыков путем регулярной и планомерной самостоятельной работы студентов на протяжении всего курса. Процесс подготовки к лабораторным работам включает изучение нормативных документов, обязательной и дополнительной литературы по рассматриваемому вопросу. Непосредственное проведение лабораторной работы предполагает:
- изучение теоретического материала по теме лабораторной работы (по вопросам изучаемой темы);
- выполнение необходимых расчетов и экспериментов;
- оформление отчета с заполнением необходимых таблиц, построением графиков, подготовкой выводов по проделанным экспериментам и теоретическим расчетам;
- по каждой лабораторной работе проводится контроль: проверяется содержание отчета, проверяется усвоение теоретического материала. Контроль усвоения теоретического материала является индивидуальным.

Методические указания по выполнению отчёта к лабораторным работам

Основным требованием по выполнению лабораторных и практических работ является полное исчерпывающее описание всей проделанной работы, позволяющее судить о полученных результатах, степени выполнения и профессиональной подготовки студентов.
Методические указания обеспечивают комплексный подход в учебной работе студентов, единство и преемственность требований к оформлению результатов работы на разных этапах обучения. С единых позиций приведены основные требования по структуре, оформлению и содержанию отчета по лабораторным и практическим работам.
Структура отчёта:
- цель работы;
- краткие теоретические сведения;
- ход выполнения работы;
- выводы.
Дополнительными элементы:
- приложения;
- библиографический список.
Требования к содержанию отчёта:
1. Титульный лист
В верхнем поле листа указывают полное наименование учебного заведения.
В среднем поле указывается вид работы, в данном случае лабораторная или практическая работа с указанием курса, по которому она выполнена, и ниже ее название. Название работы приводится без слова тема и в кавычки не заключается.
Далее ближе к правому краю титульного листа указывают фамилию, инициалы и группу учащегося, выполнившего работу, а также фамилию, инициалы преподавателя, принявшего работу.
В нижнем поле листа указывается место выполнения работы и год ее написания (без слова год).
2. Цель работы должна отражать тему работы, а также конкретные задачи, поставленные студенту на период выполнения работы. По объему цель работы в зависимости от сложности и многозадачности работы составляет от нескольких строк до 0,5 страницы.
3. Краткие теоретические сведения. В этом разделе излагается краткое теоретическое описание изучаемой в работе темы. Материал раздела не должен копировать содержание методического пособия или учебника по данной теме, а ограничивается изложением основных понятий, требующихся для дальнейшей обработки полученных результатов. Объем литературного обзора не должен превышать 1/3 части всего отчета.
4. Ход выполнения работы. В данном разделе подробно излагается методика выполнения работы, процесс получения данных и способ их обработки. Если используются стандартные пакеты компьютерных программ для обработки экспериментальных результатов, то необходимо обосновать возможность и целесообразность их применения, а также подробности обработки данных с их помощью.
5. Выводы по работе - кратко излагаются результаты работы, полученные в результате выполнения работы, а также краткий анализ полученных результатов.
Для защиты лабораторной работы студент должен подготовить отчет, провести самостоятельную работу, иметь отметку о проверенном отчете.
Результаты определяются по пятибалльной системе оценок.

Методические рекомендации по выполнению реферата

Реферат – письменная работа объемом 8–10 страниц. Это краткое и точное изложение сущности какого-либо вопроса, темы. Тему реферата студент выбирает из предложенных преподавателем или может предложить свой вариант. В реферате нужны развернутые аргументы, рассуждения, сравнения. Содержание темы излагается объективно от имени автора. Функции реферата. Информативная, поисковая, справочная, сигнальная, коммуникативная. Степень выполнения этих функций зависит от содержательных и формальных качеств реферата и для каких целей их использует. Требования к языку реферата.
Должен отличаться точностью, краткостью, ясностью и простотой.
Структура реферата:
1. Титульный лист
2. Оглавление (на отдельной странице). Указываются названия всех разделов (пунктов плана) реферата и номера страниц, указывающие начало этих разделов в тексте реферата.
3. Введение. Аргументируется актуальность исследования, т.е. выявляется практическое и теоретическое значение данного исследования. Далее констатируется, что сделано в данной области предшественниками, перечисляются положения, которые должны быть обоснованы. Обязательно формулируются цель и задачи реферата.
4. Основная часть. Подчиняется собственному плану, что отражается в разделении текста на главы, параграфы, пункты. План основной части может быть составлен с использованием различных методов группировки материала. В случае если используется чья-либо неординарная мысль, идея, то обязательно нужно сделать ссылку на того автора, у кого взят данный материал.
5. Заключение. Последняя часть научного текста. В краткой и сжатой форме излагаются полученные результаты, представляющие собой ответ на главный вопрос исследования.
6. Приложение. Может включать графики, таблицы, расчеты.
7. Библиография (список литературы). Указывается реально использованная для написания реферата литература. Названия книг располагаются по алфавиту с указанием их выходных данных. Общие требования к построению, содержанию и оформлению».
При проверке реферата оцениваются:
- знание фактического материала, усвоение общих представлений, понятий, идей;
- характеристика реализации цели и задач исследования;

 степень обоснованности аргументов и обобщений;
 качество и ценность полученных результатов;
 использование литературных источников;
 культура письменного изложения материала;
 культура оформления материалов работы.

Методические рекомендации по устному опросу/самоподготовке

После изучения определенной темы по записям в конспекте и учебнику, а также решения достаточного количества соответствующих задач на практических занятиях и самостоятельно студенту рекомендуется, используя лист опорных сигналов, воспроизвести по памяти определения, выводы формул, формулировки основных положений и доказательств. В случае необходимости следует рекомендовать еще раз внимательно разобраться в материале. Иногда недостаточность усвоения того или иного вопроса выясняется только при изучении дальнейшего материала. В этом случае надо вернуться назад и повторить плохо усвоенный материал. Важный критерий усвоения теоретического материала – умение решать задачи или пройти тестирование по пройденному материалу. Однако преподавателю следует помнить, что правильное решение задачи может получиться в результате применения механически заученных формул без понимания сущности теоретических положений.

Методические рекомендации по подготовке к семинарским занятиям

Одним из видов внеаудиторной самостоятельной работы является подготовка к семинарским занятиям. Семинар – форма учебно-практических занятий, при которой студенты обсуждают сообщения, доклады и рефераты, выполненные ими по результатам учебных или научных исследований под руководством преподавателя. Преподаватель в этом случае является координатором обсуждений темы семинара, подготовка к которому является обязательной. Поэтому тема семинара и основные источники обсуждения предъявляются до обсуждения для детального ознакомления, изучения. Цели обсуждений направлены на формирование навыков профессиональной полемики и закрепление обсуждаемого материала. Семинар – это такая форма организации обучения, при которой на этапе подготовки доминирует самостоятельная работа учащихся с
учебной литературой и другими дидактическими средствами над серией вопросов, проблем и задач, а в процессе семинара идут активное обсуждение, дискуссии и выступления учащихся, где они под руководством преподавателя делают обобщающие выводы и заключения. Семинар предназначен для углубленного изучения дисциплины, овладения методологией научного познания, то главная цель семинарских занятий – обеспечить студентам возможность овладеть навыками и умениями использования теоретического знания применительно к особенностям изучаемой отрасли.

Методические рекомендации по подготовке эссе

Одним из видов самостоятельной работы студентов является написание творческой работы по заданной либо согласованной с преподавателем теме. Творческая работа (эссе) представляет собой оригинальное произведение объемом 500-700 слов, посвященное какой-либо значимой классической либо современной проблеме в определенной теоретической и практической области. Творческая работа не является рефератом и не должна носить описательный характер, большое место в ней должно быть уделено аргументированному представлению своей точки зрения студентами, критической оценке рассматриваемого материала и проблематики, что должно способствовать раскрытию творческих и аналитических способностей. Цели написания эссе – научиться логически верно и аргументировано строить устную и письменную речь;
работать над углублением и систематизацией своих философских знаний; овладеть способностью использовать основы знаний для формирования мировоззренческой позиции. Приступая к написанию эссе, изложите в одном предложении, что именно вы будете утверждать и доказывать (свой тезис).

Методические рекомендации по подготовке к докладу

Для подготовки доклада необходимо выбрать актуальную тему. Желательно, чтобы тема была интересна докладчику и вызывала желание качественно подготовить материалы. Подготовка доклада предполагает: определение цели доклада; подбор необходимого материала, определяющего содержание доклада; составление плана доклада, распределение собранного материала в необходимой логической последовательности.
Композиция доклада имеет вступление, основную часть и заключение.
Вступление должно содержать: название доклада; сообщение основной идеи; современную оценку предмета изложения; краткое перечисление рассматриваемых вопросов; интересную для слушателей форму изложения. Основная часть, в которой необходимо раскрыть суть темы, обычно строится по принципу отчёта. Задача основной части: представить достаточно данных для того, чтобы слушатели заинтересовались темой.
Заключение – чёткое обобщение и краткие выводы по излагаемой теме.

Методические рекомендации по подготовке к собеседованию

Собеседование – средство контроля, организованное как специальная беседа преподавателя с обучающимся на темы, связанные с изучаемой дисциплиной, и рассчитанное на выяснение объема знаний обучающегося по определенному разделу, теме, проблеме и т.п.
Цель собеседования: проверка усвоения знаний; умений применять знания; сформированности профессионально значимых личностных качеств.
Подготовка к собеседованию предполагает повторение пройденного материала и приобретение навыка свободного владения терминологией и фактическими данными по определенному разделу дисциплины.

Методические рекомендации по подготовке к тестированию/ итоговому тестированию

Тестирование/итоговое тестирование – это система стандартизированных заданий, определенного содержания и упорядоченных в рамках определенной стратегии предъявления, позволяющая качественно оценить структуру и эффективно измерить уровень знаний, умений и навыков по учебной дисциплине (разделу). Тестирование/итоговое тестирование позволяет оценить у студентов уровень знаний фактического материала, умения логически мыслить, способности к рефлексии и творческому подходу к решению поставленной задачи на каждом этапе их обучения.
При самостоятельной подготовке к тестированию/итоговому тестированию студенту необходимо:
- четко выяснить все условия тестирования/итогового тестирования заранее (сколько тестов будет предложено, сколько времени отводится на тестирование/итоговое тестирование, какова система оценки результатов);
- проконсультироваться с преподавателем по вопросу выбора учебной литературы;
- проработать информационный материал по разделу дисциплине;
- конспективно выписать из соответствующих разделов учебной дисциплины основные понятия, ключевые идеи, теоретические концепции и подходы;
- повторить теоретический материал лекций, самостоятельно изученной учебной и научной литературы, семинарских занятий.
При выполнении тестовых заданий рекомендуется:
- внимательно полностью прочитать вопрос и предлагаемые варианты ответов, выбрать правильные из них (их может быть несколько);
- при затруднении при ответе на сложный для студента вопрос целесообразно переходить к другим вопросам теста. По завершению ответов на все последующие вопросы теста необходимо вернуться к трудному вопросу и дать на него продуманный ответ;
- по окончании тестирования/итогового тестирования следует проверить правильность данных ответов на вопросы теста, чтобы избежать механических ошибок.

Методические рекомендации по подготовке к экзамену

Подготовка к экзамену способствует закреплению, углублению и обобщению знаний, получаемых, в процессе обучения, а также применению их к решению практических задач. Готовясь к экзамену, студент ликвидирует имеющиеся пробелы в знаниях, углубляет, систематизирует и упорядочивает свои знания. На экзамене студент демонстрирует то, что он приобрел в процессе обучения по конкретной учебной дисциплине. Экзаменационная сессия – это серия экзаменов, установленных учебным планом. Между экзаменами интервал 2-4 дня, в течение студент систематизирует уже имеющиеся знания. На консультации перед экзаменом студенты должны
быть ознакомлены с основными требованиями и получить ответы на возникающие в процессе подготовки вопросы. Необходимо ориентировать студентов на систематическую подготовку к занятиям в течение семестра, что позволит использовать время экзаменационной сессии для систематизации знаний.

Методические рекомендации по подготовке к зачету/ зачету с оценкой

В ходе подготовки к зачету/зачету с оценкой студент, в первую очередь, должен систематизировать знания, полученные в ходе изучения дисциплины. К зачету/зачету с оценкой необходимо готовиться целенаправленно, регулярно, систематически и с первых дней обучения по данной дисциплине. В самом начале учебного курса познакомьтесь со следующей учебно- методической документацией:
- программой дисциплины;
- перечнем знаний, умений и навыков, которыми студент должен владеть;
- тематическими планами лекций, семинарских занятий;
- учебниками, учебными пособиями по дисциплине, а также электронными ресурсами;
- перечнем вопросов к зачету/зачету с оценкой.
После этого у обучающихся должно сформироваться четкое представление об объеме и характере знаний и умений, которыми надо будет овладеть по дисциплине. Систематическое выполнение учебной работы на лекциях и лабораторных занятиях позволит успешно освоить дисциплину и создать хорошую базу для сдачи зачета/зачета с оценкой.

Методические рекомендации по выполнению курсовой работы/ курсового проекта

Курсовая работа/курсовой проект - это учебно-исследовательский труд студента, выполняемый по определённой дисциплине под руководством научного руководителя. Её цель - закрепить и углубить теоретические знания, развить навыки самостоятельного анализа, работы с научной литературой и эмпирическими данными, а также сформировать умения структурировать информацию и формулировать обоснованные выводы.
Курсовая работа/курсовой проект – письменная работа объемом 30–50 страниц, выполняется в несколько этапов:
1. Подготовительный этап
2. Сбор и анализ информации
3. Написание работы
Структура курсовой работы/курсового проекта:
1. Титульный лист – по шаблону вуза.
2. Содержание – разделы с номерами страниц.
3. Введение: актуальность, цель, задачи, объект, предмет, методы, значимость, структура.
4. Основная часть:
o Глава 1 (теоретическая) – анализ концепций, выводы.
o Глава 2 (аналитическая/практическая) – анализ объекта, расчёты, данные, выводы.
o Глава 3 (рекомендательная, если нужно) – предложения, прогнозы, выводы.
5. Заключение – выводы по задачам, итоги, перспективы.
6. Список литературы – группируйте по типам, оформляйте по ГОСТ, указывайте только использованные

источники.
7. Приложения (если нужно) – таблицы, графики, анкеты и т. д. (нумеруются, есть ссылки в тексте).
4. Оформление
5. Проверка и доработка
6. Подготовка к защите
• Доклад (5–7 мин.):
• Презентация.

8. ОРГАНИЗАЦИЯ ОБРАЗОВАТЕЛЬНОГО ПРОЦЕССА ДЛЯ ЛИЦ С ОГРАНИЧЕННЫМИ ВОЗМОЖНОСТЯМИ ЗДОРОВЬЯ И ИНВАЛИДОВ

В образовательном процессе используются социально-активные и рефлексивные методы обучения, технологии социокультурной реабилитации с целью оказания помощи в установлении полноценных межличностных отношений с другими студентами, создании комфортного психологического климата в студенческой группе.
Студенты с ограниченными возможностями здоровья, в отличие от остальных студентов, имеют свои специфические особенности восприятия, переработки материала. Подбор и разработка учебных материалов производится с учетом индивидуальных особенностей.
Предусмотрена возможность обучения по индивидуальному графику, при составлении которого возможны различные варианты проведения занятий: в академической группе и индивидуально, на дому с использованием дистанционных образовательных технологий.
Основной формой в дистанционном обучении является индивидуальная форма обучения. Главным достоинством индивидуального обучения для инвалидов и лиц с ограниченными возможностями здоровья является то, что оно позволяет полностью индивидуализировать содержание, методы и темпы учебной деятельности инвалида, следить за каждым его действием и операцией при решении конкретных задач; вносить вовремя необходимые коррекции как в деятельность студента-инвалида, так и в деятельность преподавателя. Дистанционное обучение также обеспечивает возможности коммуникаций не только с преподавателем, но и с другими обучаемыми, сотрудничество в процессе познавательной деятельности.
При изучении дисциплины используются следующие организационные мероприятия:

- использование возможностей сети «Интернет» для обеспечения связи с обучающимися, предоставления им необходимых материалов для самостоятельного изучения, контроля текущей успеваемости и проведения тестирования;

- проведение видеоконференций, лекций, консультаций, и т.д. с использованием программ, обеспечивающих дистанционный контакт с обучающимся в режиме реального времени.
- предоставление электронных учебных пособий, включающих в себя основной материал по дисциплинам, включенным в ОП;

- проведение занятий, консультаций, защит курсовых работ и т.д. на базе консультационных пунктов, обеспечивающих условия для доступа туда лицам с ограниченными возможностями;
- предоставление видеолекций, позволяющих изучать материал курса дистанционно.
- использование программного обеспечения и технических средств, имеющих функции адаптации для использования лицами с ограниченными возможностями.

Актуализация с учетом развития науки, техники, культуры, экономики, техники, технологий и социальной сферы

Руководитель ОП _____

Рабочая программа актуализирована, обсуждена и одобрена на заседании ПЦК _____
Протокол от __ _____ 2026 г. № __
Председатель ПЦК _____

==========

Актуализация с учетом развития науки, техники, культуры, экономики, техники, технологий и социальной сферы

Руководитель ОП _____

Рабочая программа актуализирована, обсуждена и одобрена на заседании ПЦК _____
Протокол от __ _____ 2027 г. № __
Председатель ПЦК _____

==========

Актуализация с учетом развития науки, техники, культуры, экономики, техники, технологий и социальной сферы

Руководитель ОП _____

Рабочая программа актуализирована, обсуждена и одобрена на заседании ПЦК _____
Протокол от __ _____ 2028 г. № __
Председатель ПЦК _____

==========

Актуализация с учетом развития науки, техники, культуры, экономики, техники, технологий и социальной сферы

Руководитель ОП _____

Рабочая программа актуализирована, обсуждена и одобрена на заседании ПЦК _____
Протокол от __ _____ 2029 г. № __
Председатель ПЦК _____

==========

ПРИЛОЖЕНИЕ А

ЛАБОРАТОРНЫЕ РАБОТЫ

РАЗДЕЛ 1. ОСНОВНЫЕ ПРИНЦИПЫ АЛГОРИТМИЗАЦИИ И ПРОГРАММИРОВАНИЯ

Лабораторная работа №1. *Представление алгоритмов в графическом и табличном виде*
Цель: Закрепление навыков и овладение приемами составления алгоритмов в виде блок-схем.
Задание:

Все задания лабораторной работы выполняются по вариантам *(номер варианта определяется по номеру в списке учебной группы)*.

Задание №1. Построить блок-схему в соответствии со следующей задачей:

Вариант	Условие задачи
1	Дана площадь S круга. Найти его диаметр D и длину L окружности, ограничивающей этот круг, учитывая, что $L = \pi \cdot D$, $S = \pi \cdot D^2/4$.
2	Дан объем цилиндра V и его высота h. Вычислить радиус основания цилиндра, учитывая, что $V = \pi \cdot r^2 \cdot h$
3	Даны два круга с общим центром и радиусами R_1 и R_2 ($R_1 > R_2$). Найти площади этих кругов S_1 и S_2, а также площадь S_3 кольца, внешний радиус которого равен R_1, а внутренний радиус равен R_2: $S_1 = \pi \cdot (R_1)^2$, $S_2 = \pi \cdot (R_2)^2$, $S_3 = S_1 - S_2$.
4	Даны длины ребер a, b, c прямоугольного параллелепипеда. Найти его объем $V = a \cdot b \cdot c$ и площадь поверхности $S = 2 \cdot (a \cdot b + b \cdot c + a \cdot c)$.
5	Даны переменные A, B, C. Изменить их значения, переместив содержимое A в B, B – в C, C – в A, и вывести новые значения переменных A, B, C.
6	Найти длину окружности L и площадь круга S заданного радиуса R: $L = 2 \cdot \pi \cdot R$, $S = \pi \cdot R^2$.
7	Даны стороны прямоугольника a и b. Найти его площадь $S = a \cdot b$ и периметр P.
8	Даны два неотрицательных числа a и b. Найти их среднее геометрическое и среднее арифметическое.
9	Даны катеты прямоугольного треугольника a и b. Найти его гипотенузу c и периметр P: $c = \sqrt{a^2 + b^2}$, $P = a + b + c$.
10	Даны три стороны треугольника. Найти его периметр и площадь. Для нахождения площади треугольника со сторонами a, b, c использовать формулу Герона: $S = \sqrt{p \cdot (p - a) \cdot (p - b) \cdot (p - c)}$, где $p = (a + b + c)/2$ – полупериметр.

Задание №2. Построить блок-схему для вычисления значения выражения Y от переменной X. Формулы для вычисления значения Y приведены в таблице:

Вариант	Формула
1	$y = \begin{cases} \sin 5x, & -1 \le x < 0 \\ tg\,(1+x), & 0 \le x < 1 \\ \dfrac{1}{1+x}, & 1 \le x \le 2 \end{cases}$
2	$y = \begin{cases} \dfrac{tg\,x}{2}, & 0 \le x < 0{,}8 \\ \dfrac{\sin x}{x}, & 0{,}8 \le x < 1{,}3 \\ x^3, & 1{,}3 \le x \le 2 \end{cases}$

3	$y=\begin{cases} arctg\ x, & 0{,}1 \le x < 0{,}8 \\ \cos(x)\cdot\sin(x), & 0{,}8 \le x < 1{,}3 \\ \dfrac{e^{x}+e^{-x}}{2}, & 1{,}3 \le x \le 1{,}8 \end{cases}$
4	$y=\begin{cases} \dfrac{10}{x+1{,}5}, & 1 \le x < 3 \\ 2^{x}-2, & 3 \le x < 5 \\ \sqrt{\pi\cdot x}, & 5 \le x \le 7 \end{cases}$
5	$y=\begin{cases} \dfrac{e^{x}}{e^{-x}+\pi-1}, & -1 \le x < 0 \\ \dfrac{20\sin x}{(2+x)^{3}}, & 0 \le x \le 1 \end{cases}$
6	$y=\begin{cases} \sqrt[3]{2+x}, & 1 \le x < 2 \\ \lg\left(\dfrac{x-1}{x+1}\right), & 2 \le x \le 3 \end{cases}$
7	$y=\begin{cases} \sqrt{\lvert 0{,}5x^{2}-4x+2\rvert}, & -0{,}1 \le x < 0{,}1 \\ \dfrac{x^{3}+0{,}5x^{2}-4x+2}{\sin x}, & 0{,}1 \le x \le 0{,}3 \end{cases}$
8	$y=\begin{cases} \dfrac{tg\ x}{x-3}, & -2 \le x < 0 \\ \lg x\cdot\sin x, & 0 \le x \le 2 \end{cases}$
9	$y=\begin{cases} \dfrac{\sin 0{,}1x}{x-2}+\sqrt{x}, & 10 \le x < 20 \\ \cos(x)-x^{3{,}5}, & 20 \le x \le 30 \end{cases}$
10	$y=\begin{cases} \dfrac{tg\ x}{1+tg\ 2x}, & -2 \le x \le 1 \\ (\lg x)^{2{,}5}, & 1 < x \le 4 \end{cases}$

Задание №3. Составить алгоритм любого процесса, смысл которого хорошо понятен студенту или который студент может выполнить самостоятельно, т.е. успешно достичь конечного результата. Разработанный алгоритм должен обладать всеми свойствами алгоритма, которые студент должен определить при выполнении данного задания.

Алгоритм представить в следующих видах:

– табличный вид,

– граф,
– блок-схема.

Табличное представление алгоритма должен содержать следующую таблицу:

№ п/п	Наименование действия	Зависимость от			Графическое обозначение
		Выполнения предыдущего	Ресурсы	Исполнители	

Пример табличного представления алгоритма:

Алгоритм подготовки и выполнения лабораторной работы по физике

№ п/п	Наименование действия	Зависимость от			Графическое обозначение
		Выполнения предыдущего	Ресурсы	Исполнители	
1.	Получение задания к лабораторной работе	—	методические указания	преподаватель, студент	1
2.	Изучение теоретического материала по данным вопросам	1	библиотеки, учебные пособия, монографии, конспект лекций	работники библиотек, студент, консультант	2
3.	Получение допуска к лабораторной работе	1,2	теоретическая подготовка студента	преподаватель, студент	3
	. . . . .				

Процесс, подлежащий алгоритмизации должен быть достаточно сложным, т.е. алгоритм должен содержать не менее 10-12 действий. После табличного представления алгоритм должен быть представлен в виде графа, отражающего взаимосвязь его действий и порядок их следования.



РАЗДЕЛ 2. ЯЗЫКИ ПРОГРАММИРОВАНИЯ

Лабораторная работа №2. *Разработка консольных приложений на ввод и вывод данных. Класс Random и Math*

Цель: Закрепление навыков и овладение приемами разработки программ на ввод и вывод данных и на использование методов встроенных классов Random и Math.

Задание:

Все задания лабораторной работы выполняются по вариантам *(номер варианта определяется по номеру в списке учебной группы)*. При выполнении заданий использовать методы класса Math и Random.

Задание №1. Разработать алгоритм в виде блок-схемы и программу в соответствии со следующей задачей:

Вариант	Условие задачи
1	1. Сгенерировать два случайных целых числа (катеты) в диапазоне от 1 до 20. 2. Вывести их на экран. 3. Вычислить длину гипотенузы по формуле: $c = \sqrt{(a^2 + b^2)}$. 4. Вывести результат с округлением до 2 знаков (использовать Round).

2	1. Запросить у пользователя количество случайных углов (целое число от 3 до 10). 2. Сгенерировать указанное количество случайных углов в градусах (от 0 до 360). 3. Для каждого угла: перевести в радианы (PI * deg / 180,0), вычислить Sin(radians) 4. Вывести угол, значение синуса с точностью до 4 знаков.
3	1. Сгенерировать 7 случайных вещественных чисел от -50,0 до 50,0. 2. Найти среди них максимум (Max – использовать в цикле сравнения). 3. Найти минимум (Min). 4. Вывести все числа, затем max и min.
4	1. Сгенерировать случайное вещественное число от 0 до 100 (с двумя знаками после запятой). 2. Использовать Ceiling – округлить вверх до целого. 3. ИспользоватььFloor – округлить вниз до целого. 4. Вывести исходное число и оба результата округления.
5	1. Сгенерировать случайный радиус (вещественное число от 0.5 до 15.0). 2. Вычислить площадь круга: $S = \pi * R^2$ (использовать PI и Pow). 3. Пользователь должен ввести коэффициент увеличения площади (целое число 1-5). 4. Увеличить площадь умножением на коэффициент, вывести результат.
6	1. Сгенерировать 4 случайных положительных целых числа (от 1 до 30). 2. Вывести их. 3. Вычислить сумму. 4. Если сумма ≥ 0 – вычислить квадратный корень (Sqrt) и округлить до 3 знаков. 5. Если сумма < 0 – вывести сообщение (но по условию сумма всегда ≥ 4).
7	1. Сгенерировать случайное вещественное число от -20 до 20. 2. Вывести его. 3. Вычислить абсолютное значение (Abs). 4. Если абсолютное значение > 0, вычислить натуральный логарифм (Log). 5. Если результат логарифма отрицательный – вывести модуль от него.
8	1. Пользователь вводит степень (целое число от 1 до 5). 2. Сгенерировать два случайных целых числа от 1 до 10. 3. Каждое возвести в степень пользователя (Pow). 4. Определить, какое из возведённых чисел больше (используя Max). 5. Вывести оба результата и максимальное.
9	1. Сгенерировать случайный угол в градусах (от 0 до 180, исключая 90). 2. Перевести в радианы. 3. Вычислить тангенс (Tan). 4. Вычислить котангенс как 1 / Tan(radians) (использовать проверку на близость к нулю). 5. Вывести угол, тангенс, котангенс (с точностью 3 знака).
10	1. Сгенерировать N случайных целых чисел (N задаёт пользователь от 5 до 15) в диапазоне от 10 до 100. 2. Вывести все числа. 3. Вычислить среднее арифметическое. 4. Для каждого числа вывести его отклонение от среднего: Abs(number - average). 5. Найти максимальное отклонение.

Задание №2. Разработать алгоритм в виде блок-схемы и программу в соответствии со следующей задачей:

Вариант	Условие задачи
1	Вы работаете в геодезической компании. Вам необходимо написать программу, которая генерирует два случайных целых числа (катеты прямоугольного треугольника) в диапазоне от 1 до 30, выводит их на экран,

	вычисляет длину гипотенузы по формуле $c = \sqrt{(a^2 + b^2)}$ с использованием Sqrt и Pow, после чего округляет результат до двух знаков с помощью Round и выводит его пользователю.
2	В учебном центре по физике требуется программа для демонстрации преобразования углов. Программа должна запросить у пользователя количество случайных углов (от 3 до 12), сгенерировать указанное количество целых углов в градусах (от 0 до 360), каждый угол перевести в радианы по формуле радианы = градусы × π / 180, вычислить его синус с помощью Sin, а затем вывести для каждого угла исходное значение и соответствующий синус с точностью до 4 знаков.
3	Вы анализируете данные метеостанции. Напишите программу, которая генерирует 10 случайных вещественных температур в диапазоне от -30 до +30, выводит их на экран, находит максимальное значение с помощью Max (применяя его последовательно в цикле), минимальное значение с помощью Min, после чего сообщает пользователю, какая температура была самой высокой и какой самой низкой.
4	Сотруднику склада нужно округлять веса товаров. Программа генерирует случайное вещественное число (вес в килограммах) от 0 до 100, затем с помощью Ceiling округляет его вверх до целого числа (например, для расчета количества упаковок), с помощью Floor округляет вниз до целого (для расчета полных килограммов), после чего выводит исходный вес и оба полученных значения.
5	Вы создаете калькулятор для дизайнера интерьеров, который рассчитывает площадь круглого стола. Программа генерирует случайный радиус стола (вещественное число от 0,3 до 2,5 метров), вычисляет площадь по формуле $S = \pi \times R^2$ (используя PI и Pow), затем запрашивает у пользователя коэффициент увеличения площади (целое число от 1 до 4, означающий, во сколько раз нужно увеличить площадь для большого стола), умножает площадь на этот коэффициент и выводит итоговое значение с точностью до двух знаков.
6	В банке требуется программа для проверки вкладов. Программа генерирует 5 случайных целых положительных чисел (суммы вкладов в тысячах рублей) в диапазоне от 10 до 100, выводит их, вычисляет общую сумму всех вкладов, затем извлекает квадратный корень из этой суммы с помощью Sqrt (для расчета некоторого коэффициента надежности) и округляет результат до трех знаков. Если сумма оказалась бы отрицательной (чего по условию быть не может), программа должна вывести предупреждение.
7	Вы разрабатываете обучающую программу по математике. Программа генерирует случайное вещественное число в диапазоне от -50,0 до 50,0, выводит его, затем с помощью Abs находит его абсолютное значение, после чего (если абсолютное значение больше нуля) вычисляет натуральный логарифм абсолютного значения через Log. Если полученный логарифм оказался отрицательным, программа дополнительно выводит его модуль.
8	На заводе необходимо сравнить прочность двух материалов после обработки. Пользователь вводит целую степень обработки (от 1 до 5). Программа генерирует два случайных целых числа (начальные показатели прочности) от 5 до 20, каждое возводит в введенную степень с помощью Pow, затем с помощью Max определяет, какой из двух материалов стал прочнее, и выводит оба полученных значения вместе с указанием максимального.
9	Инженер-строитель рассчитывает уклон крыши. Программа генерирует случайный угол наклона в градусах (целое число от 1 до 89, исключая 0 и 90), переводит его в радианы, вычисляет тангенс через Tan (который показывает отношение высоты подъема к горизонтальному пролету), затем вычисляет котангенс как 1 / Tan(радианы), предусмотрев проверку, чтобы тангенс не был нулем. После этого программа выводит угол, значение

	тангенса и котангенса с точностью до трех знаков.
10	Вы работаете в статистическом отделе. Программа запрашивает у пользователя целое число N (от 5 до 15), затем генерирует N случайных целых чисел в диапазоне от 10 до 100, выводит все сгенерированные числа, вычисляет их среднее арифметическое, после чего для каждого числа вычисляет абсолютное отклонение от среднего с помощью Abs, находит максимальное отклонение среди всех чисел и выводит его.

Лабораторная работа №3. *Разработка консольных приложений с применением условного оператора и оператора выбора*

Цель: Закрепление навыков и овладение приемами разработки программ с применением условного оператора и оператора выбора.

Задание:

Все задания лабораторной работы выполняются по вариантам *(номер варианта определяется по номеру в списке учебной группы)*.

Задание №1. Разработать алгоритм в виде блок-схемы и программу в соответствии со следующей задачей:

Вариант	Условие задачи
1	Ввести значения четырех чисел x_1, x_2, x_3, x_4. Вычислить значения переменных y_1, y_2, y_3, z (если это возможно) при условии, что $y_1 = \sqrt{x_1 - 3x_2}$; $y_2 = \ln(x_2 - x_3 x_4)$; $y_3 = 2\cos(x_4 - 10)$; $z = 2y_1 + 3y_2 + 4y_3$. Значения y_1, y_2, y_3, z вывести на экран.
2	Ввести значения четырех чисел x_1, x_2, x_3, x_4. Вычислить значения переменных y_1, y_2, y_3, z (если это возможно) при условии, что $y_1 = \frac{1}{\sqrt{x_1}}$; $y_2 = \ln(5x_3)$; $y_3 = \cos x_3 - 5\sin x_4$; $z = \frac{y_1 + y_2}{y_3}$. Значения y_1, y_2, y_3, z вывести на экран.
3	Ввести значения четырех чисел x_1, x_2, x_3, x_4. Вычислить значения переменных y_1, y_2, y_3, z (если это возможно) при условии, что $y_1 = \sqrt{5x_1 - 2x_2}$; $y_2 = tg(2x_1 - 10x_2)$; $y_3 = \ln(x_3 - x_4)$; $z = (y_1 + y_2)\sqrt{y_3}$. Значения y_1, y_2, y_3, z вывести на экран.
4	Ввести значения четырех чисел x_1, x_2, x_3, x_4. Вычислить значения переменных y_1, y_2, y_3, z (если это возможно) при условии, что $y_1 = \sqrt{2x_1 - x_2}$; $y_2 = tg(5x_1 - 4x_2)$; $y_3 = \ln(5x_3 - 10)$; $z = \sqrt{y_1 + y_2 + y_3}$. Значения y_1, y_2, y_3, z вывести на экран.
5	Ввести значения четырех чисел x_1, x_2, x_3, x_4. Вычислить значения переменных y_1, y_2, y_3, z (если это возможно) при условии, что $y_1 = \sqrt{x_1^2 - x_2^2}$; $y_2 = tg(2x_1 - 10x_2)$; $y_3 = \ln(5x_2 - x_3^2)$; $z = \sin(y_1 + y_2 + y_3)$. Значения y_1, y_2, y_3, z вывести на экран.
6	Ввести значения четырех чисел x_1, x_2, x_3, x_4. Вычислить значения переменных y_1, y_2, y_3, z (если это возможно) при условии, что $y_1 = \sqrt{x_1 - 3x_2}$; $y_2 = \ln(x_2 - x_3 x_4)$; $y_3 = 2\cos(x_4 - 10)$; $z = \frac{y_1 + y_2}{y_3}$. Значения y_1, y_2, y_3, z вывести на экран.
7	Ввести значения четырех чисел x_1, x_2, x_3, x_4. Вычислить значения переменных y_1, y_2, y_3, z (если это возможно) при условии, что $y_1 = \sqrt{x_2^2 - 8x_1}$; $y_2 = \frac{1}{5x_2 - 2x_3}$; $y_3 = tg(x_3 - x_4)$; $z = tg(y_1 + y_2 + y_3)$. Значения y_1, y_2, y_3, z вывести на экран.
8	Ввести значения четырех чисел x_1, x_2, x_3, x_4. Вычислить значения

	переменных y_1, y_2, y_3, z (если это возможно) при условии, что $y_1 = \frac{1}{\sqrt{8x_2 - x_1}}$; $y_2 = \frac{1}{3x_1 - 5x_4}$; $y_3 = \ln(5 - 10x_3)$; $z = tg(y_1 y_2 + y_3)$. Значения y_1, y_2, y_3, z вывести на экран.
9	Ввести значения четырех чисел x_1, x_2, x_3, x_4. Вычислить значения переменных y_1, y_2, y_3, z (если это возможно) при условии, что $y_1 = \sqrt{5x_1 - 2x_2}$; $y_2 = tg(2x_4 - 10x_2)$; $y_3 = \ln(5x_3 - 10)$; $z = \sin(y_1 + y_2 + y_3)$. Значения y_1, y_2, y_3, z вывести на экран.
10	Ввести значения четырех чисел x_1, x_2, x_3, x_4. Вычислить значения переменных y_1, y_2, y_3, z (если это возможно) при условии, что $y_1 = \frac{1}{\sqrt{x_1}}$; $y_2 = \ln(x_2 - x_3 x_4)$; $y_3 = tg(x_3 - x_4)$; $z = \sqrt{y_1 + y_2 + y_3}$. Значения y_1, y_2, y_3, z вывести на экран.

Задание №2. Разработать алгоритм в виде блок-схемы и программу в соответствии со следующей задачей:

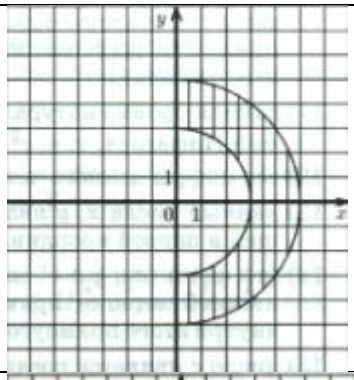
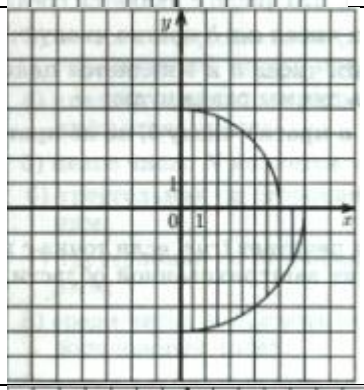
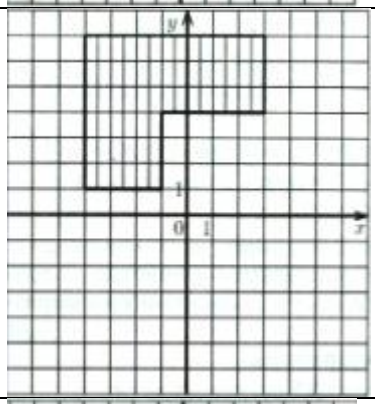
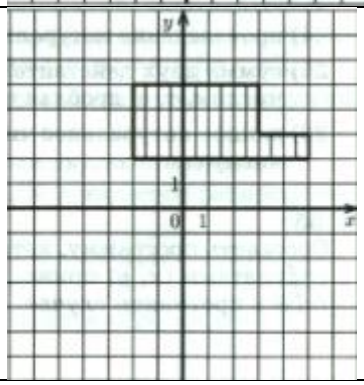
Вариант	Условие задачи
1	По введенному целому числу вывести на экран строку – словесное описание соответствующей оценки: – если пользователь введет цифру 1, то на экран выводится сообщение «Плохо», – если пользователь введет цифру 2, то на экран выводится сообщение «Неудовлетворительно», – если пользователь введет цифру 3, то на экран выводится сообщение «Удовлетворительно», – если пользователь введет цифру 4, то на экран выводится сообщение «Хорошо», – если пользователь введет цифру 5, то на экран выводится сообщение «Отлично», – если пользователь введет другие цифры или числа, то на экран должно выводится сообщение «Нет такой оценки».
2	Единицы измерения пронумерованы в следующем порядке: – 1 – километр, – 2 – метр, – 3 – миллиметр, – 4 – сантиметр. Пользователь вводит номер единицы измерения и соответствующее значение в данной единице измерения. Вывести на экран значения остальных единиц измерения (в том же порядке). Если пользователь введет другую цифру или число в качестве номера единицы измерения, то вывести сообщение «Нет такой единицы измерения».
3	Элементы окружности пронумерованы следующим образом: – 1 – радиус (R), – 2 – диаметр (D), – 3 – длина (L), – 4 – площадь круга (S). Пользователь вводит номер элемента окружности и соответствующее значение. Вывести на экран значения остальных элементов данной окружности (в том же порядке). Например, если пользователь вводит длину окружности, то на экран выводится значения радиуса, диаметра, длины и площади круга (вычисляются по значению радиуса). Если пользователь введет другое число, то вывести соответствующее сообщение.
4	По введенному целому числу выполняется соответствующая арифметическая операция:

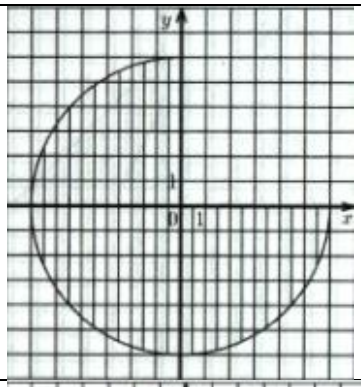
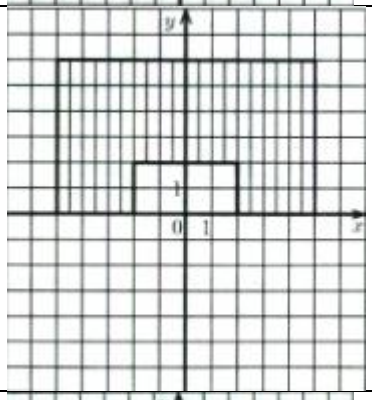
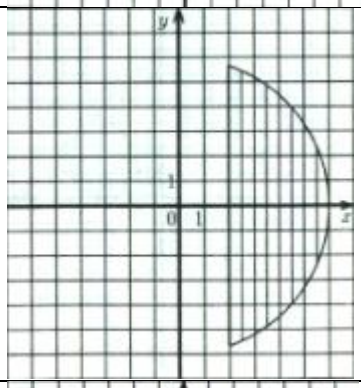
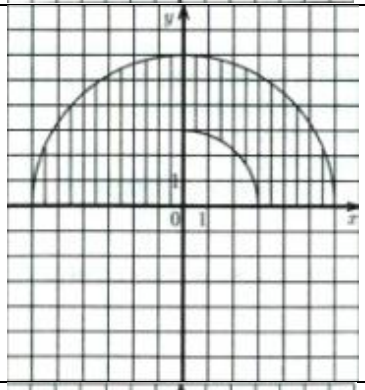
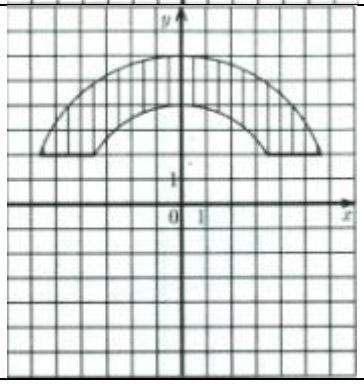
	– если, пользователь ввел цифру 1, то происходит сложение двух чисел, которые вводит пользователь, и результат выводится на экран, – если, пользователь ввел цифру 2, то происходит вычитание двух чисел, которые вводит пользователь, и результат выводится на экран, – если, пользователь ввел цифру 3, то происходит умножение двух чисел, которые вводит пользователь, и результат выводится на экран, – если, пользователь ввел цифру 4, то происходит деление двух чисел, которые вводит пользователь, и результат выводится на экран (предусмотреть деление на 0), – если пользователь вводит другие цифры (числа), то должно выводится соответствующее сообщение, что нет такой арифметической операции.
5	Вывести на экран название соответствующего времени года («зима», «весна» и т.д.) по номеру месяца. Номер месяца вводится с клавиатуры (1 – январь, 2 – февраль, ...). Если пользователь введет другое число, которое не соответствует номеру месяца, то вывести соответствующее сообщение.
6	По номеру семестра вывести на экран курс (с указанием осеннего или весеннего семестра), к которому относится введенный семестр (1 и 2 семестр – 1 курс, 3 и 4 семестр – 2 курс и т. д.). Например, если пользователь ввел цифру 4, то на экран должно выводится сообщение «2 курс, весенний семестр». Если пользователь введет другое число, не соответствующее номеру семестра, то вывести соответствующее сообщение.
7	По введенному символу выполняется соответствующая арифметическая операция: – если, пользователь ввел символ «+», то происходит сложение двух чисел, которые вводит пользователь, и результат выводится на экран, – если, пользователь ввел символ «-», то происходит вычитание двух чисел, которые вводит пользователь, и результат выводится на экран, – если, пользователь ввел символ «*», то происходит умножение двух чисел, которые вводит пользователь, и результат выводится на экран, – если, пользователь ввел символ «/», то происходит деление двух чисел, которые вводит пользователь, и результат выводится на экран (предусмотреть деление на 0), – если пользователь вводит другие символы, то должно выводится соответствующее сообщение, что нет такой арифметической операции.
8	Элементы равнобедренного прямоугольного треугольника пронумерованы следующим образом: – 1 – катет, – 2 – гипотенуза, – 3 – высота, опущенная на гипотенузу, – 4 – площадь. Дан номер одного из этих элементов и его значение. Вывести на экран значения остальных элементов данного треугольника (в том же порядке). Если пользователь ввел другое число (не 1, 2, 3 или 4), то вывести соответствующее сообщение.
9	Вывести на экран площадь геометрической фигуры, если задан тип геометрической фигуры с помощью цифры: – 1 – окружность, – 2 – равнобедренный прямоугольный треугольник, – 3 – квадрат. Второе целое число, вводимое после типа фигуры, определяет соответствующий элемент для вычисления площади (для окружности это радиус, для треугольника – длина катета, для квадрата – длина стороны). Если пользователь ввел другую цифру или число в качестве типа геометрической фигуры, то вывести соответствующее сообщение.
10	По номеру месяца (1 – январь, 2 – февраль, ...) вывести на экран число дней в этом месяце для невисокосного года (т.е. в феврале 28 дней). Если

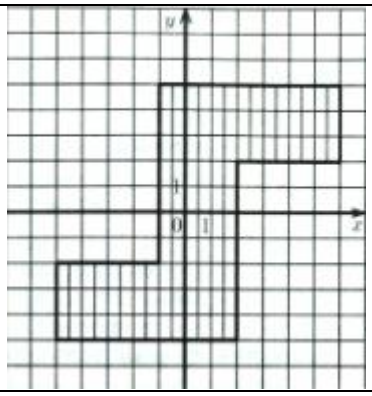
	пользователь ввел другое число, то вывести соответствующее сообщение.

Задание №3. Разработать алгоритм в виде блок-схемы и программу в соответствии со следующей задачей:

Дана область и координаты точки (вводятся с клавиатуры). Определить, принадлежит ли точка с заданными координатами, исходной области, т.е. вывести сообщение *«Точка с координатами (x, y) принадлежит заданной области»*, а в случае, если точка находится за пределами заданной области, то вывести сообщение *«Точка с координатами (x, y) НЕ принадлежит заданной области»*.

Вариант	Область данных
1	
2	
3	
4	

5	
6	
7	
8	
9	

10	

Лабораторная работа №4. *Разработка консольных приложений с применением операторов цикла*

Цель: Закрепление навыков и овладение приемами разработки программ по работе с операторами циклов.

Задание:

Все задания лабораторной работы выполняются по вариантам *(номер варианта определяется по номеру в списке учебной группы)*.

Задание №1. Разработать алгоритм в виде блок-схемы и программу в соответствии со следующей задачей:

Общая формулировка задания:

Найти сумму N слагаемых (значение N вводится с клавиатуры) и вывода на экран полученного результата. Значение X вводится с клавиатуры. При нахождении суммы N слагаемых применить цикл с параметром.

Для вычисления степени можно использовать встроенную функцию, а для вычисления факториала – переменную (не использовать какие-либо встроенные функции).

Вариант	Условие задачи
1	$\frac{x}{1!}+\frac{x^2}{2!}+\frac{x^3}{3!}+.....$
2	$-x+\frac{x^2}{2!}-\frac{x^3}{3!}+.....$
3	$\frac{x}{x+1!}+\frac{x^2}{x+2!}+\frac{x^3}{x+3!}+.....$
4	$(x-1)+\frac{x^2-2}{2!}+\frac{x^3-3}{3!}+.....$
5	$\frac{x-1}{1!}-\frac{x^2-2}{2!}+\frac{x^3-3}{3!}-\frac{x^4-4}{4!}+.....$
6	$-\frac{x}{1!}+2\frac{x^2}{2!}-3\frac{x^3}{3!}+.....$
7	$-x+\frac{x^2+5}{2!-1}-\frac{x^3+6}{3!-2}+.....$
8	$\frac{x}{1!}-\frac{x^4}{2!}+2\frac{x^5}{3!}-3\frac{x^6}{4!}+.....$
9	$-x^2+\frac{x^3}{2!}-\frac{x^4}{3!}+.....$
10	$x+\frac{x^4+1}{2!}+\frac{x^5+2}{3!}+.....$

Задание №2. Разработать алгоритм в виде блок-схемы и программу в соответствии со следующей задачей:

Переделать задание №1:

– если номер варианта является чётным, то цикл с параметром переделать на цикл с предусловием,

– если номер варианта является нечётным, то цикл с параметром переделать на цикл с постусловием,

Задание №3. Разработать алгоритм в виде блок-схемы и программу в соответствии со следующей задачей:

Вариант №1. Напишите программу, которая в последовательности натуральных чисел определяет максимальное число, кратное 5. Программа получает на вход количество чисел в последовательности, а затем сами числа. В последовательности всегда имеется число, кратное 5. Количество чисел не превышает 1000. Введённые числа не превышают 30 000. Программа должна вывести одно число – максимальное число, кратное 5.

Пример работы программы:

Входные данные	Выходные данные
3 10 25 12	25

Вариант №2. Напишите программу, которая в последовательности натуральных чисел определяет сумму чисел, кратных 6. Программа получает на вход количество чисел в последовательности, а затем сами числа. В последовательности всегда имеется число, кратное 6. Количество чисел не превышает 100. Введённые числа не превышают 300. Программа должна вывести одно число – сумму чисел, кратных 6.

Пример работы программы:

Входные данные	Выходные данные
3 12 25 6	18

Вариант №3. Напишите программу, которая в последовательности натуральных чисел определяет количество чисел, кратных 4. Программа получает на вход количество чисел в последовательности, а затем сами числа. В последовательности всегда имеется число, кратное 4. Количество чисел не превышает 1000. Введённые числа не превышают 30 000. Программа должна вывести одно число – количество чисел, кратных 4.

Пример работы программы:

Входные данные	Выходные данные
3 16 26 24	2

Вариант №4. Напишите программу, которая в последовательности натуральных чисел определяет количество чисел, кратных 4, но не кратных 7. Программа получает на вход количество чисел в последовательности, а затем сами числа. В последовательности всегда имеется число, кратное 4 и не кратное 7. Количество чисел не превышает 1000. Введённые числа не превышают 30 000. Программа должна вывести одно число: количество чисел, кратных 4, но не кратных 7.

Пример работы программы:

Входные данные	Выходные данные
4 16 28 26 24	2

Вариант №5. Напишите программу, которая в последовательности натуральных чисел определяет количество чисел, кратных 3 и оканчивающихся на 2. Программа получает на вход количество чисел в последовательности, а затем сами числа. Количество чисел не превышает 1000. Введённые числа по модулю не превышают 30 000. Программа должна вывести одно число: количество чисел, кратных 3 и оканчивающихся на 2.

Пример работы программы:

Входные данные	Выходные данные
4 12 25 12 9	2

Вариант №6. Напишите программу, которая в последовательности натуральных чисел определяет сумму чисел, оканчивающихся на 8. Программа получает на вход количество чисел в последовательности, а затем сами числа. В последовательности всегда имеется число, оканчивающееся на 8. Количество чисел не превышает 100. Введённые числа по модулю не превышают 300. Программа должна вывести одно число: сумму чисел, оканчивающихся на 8.

Пример работы программы:

Входные данные	Выходные данные
3 18 28 24	46

Вариант №7. Напишите программу, которая в последовательности натуральных чисел определяет минимальное число, кратное 3. Программа получает на вход количество чисел в последовательности, а затем сами числа. В последовательности всегда имеется число, кратное 3. Количество чисел не превышает 1000. Введённые числа не превышают 30 000. Программа должна вывести одно число – минимальное число, кратное 3.

Пример работы программы:

Входные данные	Выходные данные
3 21 12 31	12

Вариант №8. Напишите программу для решения следующей задачи. Камера наблюдения регистрирует в автоматическом режиме скорость проезжающих мимо неё автомобилей, округляя значения скорости до целых чисел. Необходимо определить среднюю зарегистрированную скорость всех автомобилей. Если скорость хотя бы одного автомобиля была не меньше 60 км/ч, выведите «YES», иначе выведите «NO».

Программа получает на вход число проехавших автомобилей N ($1 \le N \le 30$), затем указываются их скорости. Значение скорости не может быть меньше 1 и больше 300. Программа должна сначала вывести среднюю скорость с точностью до одного знака после запятой, затем «YES» или «NO».

Пример работы программы:

Входные данные	Выходные данные
4 74 69 63 96	75.5 YES

Вариант №9. Напишите программу, которая в последовательности натуральных чисел определяет сумму чисел, кратных 3. Программа получает на вход количество чисел в последовательности, а затем сами числа. В последовательности всегда имеется число, кратное 3. Количество чисел не превышает 100. Введённые числа не превышают 300. Программа должна вывести одно число – сумму чисел, кратных 3.

Пример работы программы:

Входные данные	Выходные данные
3 12 25 9	21

Вариант №10. Напишите программу, которая в последовательности натуральных чисел определяет минимальное число, оканчивающееся на 6. Программа получает на вход количество чисел в

последовательности, а затем сами числа. В последовательности всегда имеется число, оканчивающееся на 6. Количество чисел не превышает 1000. Введённые числа не превышают 30 000. Программа должна вывести одно число – минимальное число, оканчивающееся на 6.

Пример работы программы:

Входные данные	Выходные данные
3 26 16 36	16

Задание №4. Разработать алгоритм в виде блок-схемы и программу в соответствии со следующей задачей:

Вариант №1. Напишите программу, которая в последовательности целых чисел определяет количество нечётных чисел, кратных 3. Программа получает на вход целые числа, количество введённых чисел неизвестно, последовательность чисел заканчивается числом 0 (0 – признак окончания ввода, не входит в последовательность). Количество чисел не превышает 1000. Введённые числа по модулю не превышают 30 000. Программа должна вывести два числа: длину последовательности (завершающий 0 не учитывается) и количество нечётных чисел, кратных 3.

Пример работы программы:

Входные данные	Выходные данные
4 6 15 180 0	4 1

Вариант №2. Напишите программу, которая в последовательности целых чисел определяет их сумму и количество чётных чисел, кратных 5. Программа получает на вход целые числа, количество введённых чисел неизвестно, последовательность чисел заканчивается числом 0 (0 – признак окончания ввода, не входит в последовательность). Количество чисел не превышает 1000. Введённые числа по модулю не превышают 30 000. Программа должна вывести два числа: сумму последовательности и количество чётных чисел, кратных 5.

Пример работы программы:

Входные данные	Выходные данные
4 60 15 0	79 1

Вариант №3. Напишите программу, которая в последовательности натуральных чисел определяет сумму чисел последовательности, кратных трём и не кратных пяти. Программа получает на вход натуральные числа, количество введённых чисел неизвестно, последовательность чисел заканчивается числом 0 (0 – признак окончания ввода, не входит в последовательность). В последовательности всегда имеется число, кратное трём и не кратное пяти. Количество чисел не превышает 1000. Введённые числа не превышают 30000.

Программа должна вывести одно число – сумму чисел последовательности, кратных трём и не кратных пяти.

Пример работы программы:

Входные данные	Выходные данные
15 12 7 27 0	39
3 18 36	57

Вариант №4. Напишите программу, которая в последовательности натуральных чисел находит среднее арифметическое чисел, кратных 8, или сообщает, что таких чисел нет (выводит «NO»).

Программа получает на вход натуральные числа, количество введённых чисел неизвестно, последовательность чисел заканчивается числом 0 (0 – признак окончания ввода, не входит в последовательность). Количество чисел не превышает 100. Введённые числа не превышают 300. Программа должна вывести среднее арифметическое чисел, кратных 8, или вывести «NO», если таких чисел нет. Значение выводить с точностью до десятых.

Пример работы программы:

Входные данные	Выходные данные
8 122 64 16 0	29.3
111 1 0	NO

Вариант №5. Напишите программу, которая в последовательности натуральных чисел находит среднее арифметическое двузначных чисел или сообщает, что таких чисел нет (выводит «NO»). Программа получает на вход натуральные числа, количество введённых чисел неизвестно, последовательность чисел заканчивается числом 0 (0 – признак окончания ввода, не входит в последовательность). Количество чисел не превышает 100. Введённые числа не превышают 300. Программа должна вывести среднее арифметическое двузначных чисел или вывести «NO», если таких чисел нет. Значение выводить с точностью до десятых

Пример работы программы:

Входные данные	Выходные данные
10 120 49 0	29.5
111 1 0	NO

Вариант №6. Напишите программу, которая в последовательности натуральных чисел определяет количество таких чисел, которые кратны 4 и оканчиваются на 2. Программа получает на вход натуральные числа, количество введённых чисел неизвестно, последовательность чисел заканчивается числом 0 (0 – признак окончания ввода, не входит в последовательность). Количество чисел не превышает 1000. Введённые числа не превышают 30 000. Программа должна вывести одно число: количество чисел, кратных 4 и оканчивающихся на 2.

Пример работы программы:

Входные данные	Выходные данные
12 140 22 0	1

Вариант №7. Напишите программу, которая в последовательности натуральных чисел определяет сумму всех чисел, кратных 6 и оканчивающихся на 4. Программа получает на вход натуральные числа, количество введённых чисел неизвестно, последовательность чисел заканчивается числом 0 (0 – признак окончания ввода, не входит в последовательность). Количество чисел не превышает 1000. Введённые числа не превышают 30 000. Программа должна вывести одно число: сумму всех чисел, кратных 6 и оканчивающихся на 4.

Пример работы программы:

Входные данные	Выходные данные
14 24 144 22 12	168

0	

Вариант №8. Напишите программу, которая в последовательности натуральных чисел определяет количество чётных трёхзначных чисел. Программа получает на вход натуральные числа, количество введённых чисел неизвестно, последовательность чисел заканчивается числом 0 (0 – признак окончания ввода, не входит в последовательность). Количество чисел не превышает 1000. Введённые числа не превышают 30000.

Программа должна вывести одно число – количество чётных трёхзначных чисел.

Пример работы программы:

Входные данные	Выходные данные
124 18 376 241 0	2
101 999 8 1002 0	0

Вариант №9. Напишите программу, которая в последовательности натуральных чисел определяет максимальное число, меньшее 1000. Программа получает на вход натуральные числа, количество введённых чисел неизвестно, последовательность чисел заканчивается числом 0 (0 – признак окончания ввода, не входит в последовательность). Количество чисел не превышает 1000. Введённые числа не превышают 30000.

Программа должна вывести одно число – максимальное число последовательности, меньшее 1000.

Пример работы программы:

Входные данные	Выходные данные
265 3248 789 35 0	789
1000 999 28 0	999

Вариант №10. Напишите программу, которая в последовательности натуральных чисел определяет наименьшее число, кратное семи и не оканчивающееся на 4. Программа получает на вход натуральные числа, количество введённых чисел неизвестно, последовательность чисел заканчивается числом 0 (0 – признак окончания ввода, не входит в последовательность). В последовательности всегда имеется число, кратное семи и не оканчивающееся на 4. Количество чисел не превышает 1000. Введённые числа не превышают 30000.

Программа должна вывести одно число – наименьшее число, кратное семи и не оканчивающееся на 4.

Пример работы программы:

Входные данные	Выходные данные
14 21 28 12 0	21
42 14 7 0	7

Лабораторная работа №5. *Разработка консольных приложений по обработке массивов*

Цель: Закрепление навыков и овладение приемами разработки программ по обработке одномерных и двумерных массивов.

Задание:

Все задания лабораторной работы выполняются по вариантам *(номер варианта определяется по номеру в списке учебной группы)*.

Задание №1. Разработать алгоритм в виде блок-схемы и программу в соответствии со следующей задачей:

Общая формулировка задачи:

Заполнить массив из 10 элементов случайными целыми числами (положительными и отрицательными). В данном массиве найти (выполняется по вариантам). При разработке программы на языке C# необходимо установить заголовок окна консоли, а также изменить цвет символов для исходных данных и для вывода результата (можно изменить цвет фона).

Вариант	Условие задачи
1	Количество нулевых элементов.
2	Сумму отрицательных элементов, взятых по модулю.
3	Среднее арифметическое положительных элементов.
4	Сумму чётных элементов, которые больше 10.
5	Количество элементов больше заданного числа M (вводит пользователь с клавиатуры).
6	Сумму элементов равных заданному числу M (вводит пользователь с клавиатуры).
7	Количество чётных элементов, которые больше заданного числа M (вводит пользователь с клавиатуры).
8	Сумму элементов не равных заданному числу M (вводит пользователь с клавиатуры).
9	Количество элементов, которые равны первому элементу массива.
10	Сумму элементов, которые равны последнему элементу массива.

Задание №2. Разработать алгоритм в виде блок-схемы и программу в соответствии со следующей задачей:

Общая формулировка задачи:

Размерность массива задана константой N. Заполнить массив из N элементов случайными целыми числами (положительными и отрицательными). В данном массиве (выполняется по вариантам). При разработке программы на языке C# необходимо установить заголовок окна консоли, а также изменить цвет символов для исходных данных и для вывода результата (можно изменить цвет фона).

Вариант	Условие задачи
1	Определить является ли сумма элементов массива чётным числом. Выполнить сортировку по возрастанию (использовать метод пузырька). В текстовый файл вывести исходный и отсортированный массив (в разных строках, каждый элемент массива указывается через пробел).
2	Определить является ли сумма элементов массива положительным числом. Выполнить сортировку по убыванию (использовать метод минимального элемента). В текстовый файл вывести исходный и отсортированный массив (в разных строках, каждый элемент массива указывается через пробел).
3	Определить является ли произведение элементов массива имеющих нечётные порядковые номера чётным числом. Выполнить сортировку по возрастанию (использовать метод пузырька). В текстовый файл вывести исходный и отсортированный массив (в разных строках, каждый элемент массива указывается через пробел).
4	Определить среднее арифметическое нечётных элементов массива. Выполнить сортировку по убыванию (использовать метод пузырька). В текстовый файл вывести исходный и отсортированный массив (в разных строках, каждый элемент массива указывается через пробел).
5	Определить количество чётных элементов массива, которые меньше среднего арифметического его элементов. Выполнить сортировку по возрастанию (использовать метод минимального элемента). В текстовый

	файл вывести исходный и отсортированный массив (в разных строках, каждый элемент массива указывается через пробел).
6	Определить количество пар элементов массива, в которых сумма элементов делится на 3, но не делится на 9 (под парой подразумеваются два соседних элемента массива). Выполнить сортировку по убыванию (использовать метод минимального элемента). В текстовый файл вывести исходный и отсортированный массив (в разных строках, каждый элемент массива указывается через пробел).
7	Определить является ли сумма элементов массива имеющих чётные порядковые номера нечётным числом. Выполнить сортировку по возрастанию (использовать метод пузырька). В текстовый файл вывести исходный и отсортированный массив (в разных строках, каждый элемент массива указывается через пробел).
8	Определить количество пар элементов массива, в которых сумма элементов чётное число (под парой подразумеваются два соседних элемента массива). Выполнить сортировку по убыванию (использовать метод пузырька). В текстовый файл вывести исходный и отсортированный массив (в разных строках, каждый элемент массива указывается через пробел).
9	Определить количество чётных элементов массива, которые меньше среднего арифметического его элементов. Выполнить сортировку по возрастанию (использовать метод минимального элемента). В текстовый файл вывести исходный и отсортированный массив (в разных строках, каждый элемент массива указывается через пробел).
10	Определить количество пар элементов массива, в которых произведение элементов нечётное число (под парой подразумеваются два соседних элемента массива). Выполнить сортировку по убыванию (использовать метод пузырька). В текстовый файл вывести исходный и отсортированный массив (в разных строках, каждый элемент массива указывается через пробел).

Задание №3. Разработать алгоритм в виде блок-схемы и программу в соответствии со следующей задачей:

При разработке программы на языке C# необходимо установить заголовок окна консоли, а также изменить цвет символов для исходных данных и для вывода результата (можно изменить цвет фона).

При решении задач не использовать встроенные функции для нахождения максимального (минимального) значения элементов массива, сортировки массива и т.п.

Вариант	**Условие задачи**
1	Дан целочисленный массив из N элементов (число N вводит пользователь с клавиатуры). Элементы массива – случайные числа в диапазоне от -100 до 100 включительно. Найти и вывести на экран количество пар элементов массива, произведение которых положительно, а сумма по модулю кратна 7. Под парой подразумевается два подряд идущих элемента массива. Если таких пар нет, то вывести на экран соответствующее сообщение. Если такие пары есть, то их записать в текстовый файл (каждую пару на новой строке, а в паре числа через пробел).
2	Дан массив, содержащий N неотрицательных целых чисел (число N вводит пользователь с клавиатуры). Элементы массива – случайные числа в диапазоне от 10 до 100. Ямой называется не крайний элемент массива, который меньше обоих своих соседей. Необходимо найти в массиве самую глубокую яму, то есть яму, значение которой минимально. Если в массиве нет ни одной ямы, то вывести на экран соответствующее сообщение. Если ямы есть, то вывести в файл все найденные ямы (в виде тройки чисел через пробел, каждая тройка на отдельной строке), а на экран вывести только самую глубокую яму. Например, в массиве из шести элементов, равных соответственно 4, 9, 2, 17, 3, 8, есть две ямы – 2 и 3, самая глубокая яма – 2. В файл будет записаны следующие тройки (каждая тройка на отдельной строке): 9 2 17 и 17 3 8

3	Дан целочисленный массив из N элементов (число N вводит пользователь с клавиатуры). Элементы массива – случайные числа в диапазоне от 0 до 1000 включительно. Найти и вывести на экран количество пар элементов массива, в которых шестнадцатеричная запись обеих чисел оканчивается на B или F. Под парами понимайте два смежных, соседних элемента массива. Все найденные пары записать в текстовый файл (каждую пару на новой строке, а в паре числа через пробел). Если таких пар нет, то вывести соответствующее сообщение.
4	Дан целочисленный массив из N элементов (число N вводит пользователь с клавиатуры). Элементы массива – случайные числа в диапазоне от 0 до 100 включительно. Найти и вывести на экран сумму элементов самой длинной возрастающей последовательности подряд идущих элементов массива. Если таких последовательностей несколько, то можно найти сумму для любой последовательности. В текстовый файл записать эту возрастающую последовательность.
5	Дан целочисленный массив из N элементов (число N формируется случайным образом в диапазоне от 5 до 15). Массив заполняется пользователем с клавиатуры. Выполнить сортировку массива по возрастанию последней цифры числа. Определить и вывести на экран количество замен при сортировке. Записать отсортированный массив в текстовый файл (каждое число массива на отдельной строке).
6	Дан массив, содержащий N целых чисел (число N и элементы массива вводятся пользователем с клавиатуры). Числа могут быть положительными и отрицательными. Найти и вывести на экран максимальный по модулю элемент массива, который в троичной системе счисления оканчивается на 2. В текстовый файл вывести все числа, которые по модулю в троичной системе счисления оканчиваются на 2 (каждое число на новой строке). Если в массиве нет таких элементов, то вывести соответствующее сообщение.
7	Дан массив, содержащий N положительных и отрицательных целых чисел (число N вводит пользователь с клавиатуры). Элементы массива – случайные числа в диапазоне от -100 до 100. Если сумма всех элементов массива, взятых по модулю, чётная, то нужно вывести на экран количество чётных (по значению) элементов массива, а сами элементы записать в файл, а если сумма нечётная, то на экран вывести количество нечётных элементов и сами элементы записать в текстовый файл.
8	Дан целочисленный массив из N элементов (число N вводит пользователь с клавиатуры). Элементы массива – случайные числа в диапазоне от 0 до 1000 включительно. Определить сумму чисел среди тех элементов массива, у которых в десятичной записи вторая с конца цифра (число десятков) – чётная. Все такие числа, в десятичной записи которых вторая с конца цифра – чётная, записать в текстовый файл (каждое число на новой строке). Если таких чисел нет, то вывести сообщение.
9	Дан целочисленный массив из N элементов (число N вводит пользователь с клавиатуры). Элементы массива – случайные числа в диапазоне от 100 до 1000 включительно. Найти и вывести на экран сумму элементов массива, у которых последние две цифры числа одинаковые. Если таких элементов нет, то вывести на экран соответствующее сообщение. Если такие числа есть, то их записать в текстовый файл (каждое число на новой строке).
10	Дан целочисленный массив из N элементов (число N вводит пользователь с клавиатуры). Элементы массива – случайные числа в диапазоне от 0 до 1000 включительно. Выполнить сортировку массива по убыванию последних двух цифр числа. Определить количество замен при сортировке. Вывести отсортированный массив в текстовый файл.

Задание №4. Разработать алгоритм в виде блок-схемы и программу в соответствии со следующей задачей:

При разработке программы на языке C# необходимо установить заголовок окна консоли, а также изменить цвет символов для исходных данных и для вывода результата (можно изменить цвет фона).

При решении задач не использовать встроенные функции для нахождения максимального, минимального значения и т.п.

Использовать двойной вложенный цикл с параметром для решения задачи.

Вариант 1. Заполнить матрицу 10×10 следующим образом:

$$A = \begin{pmatrix} 1...2...3...4...5...6...7...8...9...10 \\ 2...0...0...0...0...0...0...0...0...9 \\ 3...0...0...0...0...0...0...0...0...8 \\ 4...0...0...0...0...0...0...0...0...7 \\ 5...0...0...0...0...0...0...0...0...6 \\ 6...0...0...0...0...0...0...0...0...5 \\ 7...0...0...0...0...0...0...0...0...4 \\ 8...0...0...0...0...0...0...0...0...3 \\ 9...0...0...0...0...0...0...0...0...2 \\ 10...0...0...0...0...0...0...0...0...1 \end{pmatrix}$$

Вывести заполненную матрицу на экран компьютера. Подсчитать произведение ненулевых элементов, а также среднее арифметическое ненулевых чисел. Вывести полученный результат.

Вариант 2. Заполнить матрицу 10×10 следующим образом:

$$A = \begin{pmatrix} 1...2...3...4...5...6...7...8...9...10 \\ 0...2...0...0...0...0...0...0...9...0 \\ 0...0...3...0...0...0...0...8...0...0 \\ 0...0...0...4...0...0...7...0...0...0 \\ 0...0...0...0...5...6...0...0...0...0 \\ 0...0...0...0...5...6...0...0...0...0 \\ 0...0...0...4...0...0...7...0...0...0 \\ 0...0...3...0...0...0...0...8...0...0 \\ 0...2...0...0...0...0...0...0...9...0 \\ 1...0...0...0...0...0...0...0...0...10 \end{pmatrix}$$

Вывести заполненную матрицу на экран компьютера. Подсчитать сумму квадратов элементов и среднее арифметическое элементов главной диагонали. Вывести полученный результат.

Вариант 3. Заполнить матрицу 10×10 следующим образом:

$$A = \begin{pmatrix} 0...0...0...0...0...6...7...8...9...10 \\ 0...0...0...0...0...7...8...9...10.11 \\ 0...0...0...0...0...8...9...10.11.12 \\ 0...0...0...0...0...9...10.11.12.13 \\ 0...0...0...0...0...10.11.12.13.14 \\ 1...2...3...4...5...0...0...0...0...0 \\ 1...2...3...4...5...0...0...0...0...0 \\ 1...2...3...4...5...0...0...0...0...0 \\ 1...2...3...4...5...0...0...0...0...0 \\ 1...2...3...4...5...0...0...0...0...0 \end{pmatrix}$$

Вывести заполненную матрицу на экран компьютера. Подсчитать количество ненулевых элементов матрицы, у которых сумма индексов – чётное число. Среди таких элементов найти минимальное значение. Вывести полученный результат.

Вариант 4. Заполнить матрицу 10×10 следующим образом:

$$A = \begin{pmatrix} 1...2...3...4...5...6...7...8...9...10 \\ 11.12.13.14.15.16.17.18.19.20 \\ 21.22.23.24.25.26.27.28.29.30 \\ 31.32.33.34.35.36.37.38.39.40 \\ 41.42.43.44.45.46.47.48.49.50 \\ 51.52.53.54.55.56.57.58.59.60 \\ 61.62.63.64.65.66.67.68.69.70 \\ 71.72.73.74.75.76.77.78.79.80 \\ 81.82.83.84.85.86.87.88.89.90 \\ 91.92.93.94.95.96.97.98.99.100 \end{pmatrix}$$

Вывести заполненную матрицу на экран компьютера. Подсчитать сумму элементов матрицы, у которых сумма индексов – нечётное число. Найти чётный максимальный элемент, который находится ниже главной диагонали. Вывести полученный результат.

Вариант 5. Заполнить матрицу 10×10 следующим образом:

$$A = \begin{pmatrix} 1...1...1...1...1...1...1...1...1...1 \\ 0...2...2...2...2...2...2...2...2...0 \\ 0...0...3...3...3...3...3...3...0...0 \\ 0...0...0...4...4...4...4...0...0...0 \\ 0...0...0...0...5...5...0...0...0...0 \\ 0...0...0...0...6...6...0...0...0...0 \\ 0...0...0...7...0...0...7...0...0...0 \\ 0...0...8...0...0...0...0...8...0...0 \\ 0...9...0...0...0...0...0...0...9...0 \\ 10.10.10.10.10.10.10.10.10.10 \end{pmatrix}$$

Вывести заполненную матрицу на экран компьютера. Подсчитать среднее арифметическое ненулевых элементов, больших 4 и меньших 9, вывести полученный результат.

Вариант 6. Заполнить матрицу 10×10 следующим образом:

$$A = \begin{pmatrix} 1...0...0...0...0...0...0...0...0...0 \\ 2...2...0...0...0...0...0...0...0...0 \\ 3...3...3...0..0...0...0...0...0...0 \\ 4...4...4...4...0...0...0...0...0...0 \\ 5...5...5...5...5...0...0...0...0...0 \\ 6...6...6...6...6...6...0...0...0...0 \\ 7...7...7...7...7...7...7...0...0...0 \\ 8...8...8...8...8...8...8...8...0...0 \\ 9...9...9...9...9...9...9...9...9...0 \\ 10.10.10.10.10.10.10.10.10.10 \end{pmatrix}$$

Вывести заполненную матрицу на экран компьютера. Подсчитать произведение и количество элементов матрицы, больших 2 и меньших 5, вывести полученный результат.

Вариант 7. Заполнить матрицу 10×10 следующим образом:

$$A = \begin{pmatrix} 1...1...1...1...1...0...0...0...0...0 \\ 0...2...2...2...2...0...0...0...0...0 \\ 0...0...3...3...3...0...0...0...0...0 \\ 0...0...0...4...4...0...0...0...0...0 \\ 0...0...0...0...5.....0...0...0...0 \\ 0...0...0...0...0...6...0...0...0...0 \\ 0...0...0...0...0...7...7...0...0...0 \\ 0...0...0...0...0...8...8...8...0...0 \\ 0...0...0...0...0...9...9...9...9...0 \\ 0...0...0...0...0..10.10.10.10.10 \end{pmatrix}$$

Вывести заполненную матрицу на экран компьютера. Подсчитать количество ненулевых элементов. Найти произведение всех элементов больших 4 и меньших 8, вывести полученный результат.

Вариант 8. Заполнить матрицу 10×10 следующим образом:

$$A = \begin{pmatrix} 1...2...3...4...5...0...0...0...0...0 \\ 2...3...4...5...6...0...0...0...0...0 \\ 3...4...5...6...7...0...0...0...0...0 \\ 4...5...6...7...8...0...0...0...0...0 \\ 5...6...7...8...9...0...0...0...0...0 \\ 0...0...0...0...0...6...7...8...9..10 \\ 0...0...0...0...0...7...8...9..10..11 \\ 0...0...0...0...0...8...9..10..11.12 \\ 0...0...0...0...0...9..10..11.12.13 \\ 0...0...0...0...0..10.11.12.13..14 \end{pmatrix}$$

Вывести заполненную матрицу на экран компьютера. Подсчитать сумму элементов, больших 10, вывести полученный результат.

Вариант 9. Заполнить матрицу 10×10 следующим образом:

$$A = \begin{pmatrix} 1...2...3...4...5...6...7...8...9..10 \\ 0...2...3...4...5...6...7...8...9..10 \\ 0...0...3...4...5...6...7...8...9..10 \\ 0...0...0...4...5...6...7...8...9..10 \\ 0...0...0...0...5...6...7...8...9..10 \\ 0...0...0...0...0...6...7...8...9..10 \\ 0...0...0...0...0...0...7...8...9..10 \\ 0...0...0...0...0...0...0...8...9..10 \\ 0...0...0...0...0...0...0...0...9..10 \\ 0...0...0...0...0...0...0...0...0..10 \end{pmatrix}$$

Вывести заполненную матрицу на экран компьютера. Подсчитать количество ненулевых элементов, у которых номер столбца является нечётным числом, вывести полученный результат.

Вариант 10. Заполнить матрицу 10×10 следующим образом:

$$A = \begin{pmatrix} 1...1...1...1...1...1...1...1...1...1 \\ 1...2...2...2...2...2...2...2...2...2 \\ 1...2...3...3...3...3...3...3...3...3 \\ 1...2...3...4...4...4...4...4...4...4 \\ 1...2...3...4...5...5...5...5...5...5 \\ 1...2...3...4...5...6...6...6...6...6 \\ 1...2...3...4...5...6...7...7...7...7 \\ 1...2...3...4...5...6...7...8...8...8 \\ 1...2...3...4...5...6...7...8...9...9 \\ 1...2...3...4...5...6...7...8...9..10 \end{pmatrix}$$

Вывести заполненную матрицу на экран компьютера. Найти сумму нечетных элементов матрицы, вывести полученный результат.

Задание №5. Разработать алгоритм в виде блок-схемы и программу в соответствии со следующей задачей:

При выполнении данного задания можно использовать любой вид цикла. Заполнять матрицы можно с клавиатуры или случайными числами или числа брать из файла.

Необходимо вывести сформированную исходную матрицу (матрицу выводить в формате прямоугольной таблицы, а не так чтобы все элементы матрицы были на новой строке или все элементы на одной строке).

Если с матрицей происходят какие-либо изменения (сортировка, замена строк или столбцов местами и т.п.), то необходимо вывести на экран преобразованную матрицу.

Для сортировки запрещается использовать встроенные методы.

Вариант	Задача
1	Дана действительная квадратная матрица 7×7. Получить вектор b = (b_1, b_2, b_3, b_4, b_5, b_6, b_7), где b_i – это количество чётных элементов в i-й строке. Вывести на экран матрицу A и вектор b. Упорядочить строки матрицы A по убыванию элементов первого столбца.
2	С клавиатуры вводятся элементы матрицы 5×5. Определить, является ли заданная целая квадратная матрица симметричной относительно главной диагонали. Если не является, то необходимо исходную матрицу транспонировать. Вывести транспонированную матрицу на экран или сообщение о том, что исходная матрица симметричная. Определить, есть ли столбцы матрицы, элементы которых образуют убывающую последовательность элементов.
3	Матрицу 4×4 вводим с клавиатуры. Элемент матрицы назовем седловой точкой, если он является наименьшим в своей строке и одновременно наибольшим в своем столбце. Для введенной матрицы найти и вывести на печать все ее седловые точки и их индексы, если таких точек нет вывести соответствующее сообщение. При отладке программы один из тестовых примеров должен содержать хотя бы 1 седловую точку.
4	Дана вещественная матрица размером 4×6. Упорядочить ее строки по возрастанию их наибольших элементов. Вывести вектор наибольших элементов строк, первоначальную и полученную матрицу на экран. Для данной матрицы получить $x_1x_6 + x_2x_5 + \ldots + x_6x_1$, где x_i – наибольшее значение i-го столбца данной матрицы.
5	Дана вещественная матрица размером 4×6. Упорядочить ее строки по не убыванию их первых элементов. Вывести первоначальную и полученную матрицу на экран. Определить, есть ли столбцы матрицы с возрастающими последовательностями элементов. Вывести номера таких столбцов.
6	Дана целочисленная квадратная матрица 7×7. Получить вектор b = (b_1, b_2, b_3, b_4, b_5, b_6, b_7), где b_i – это наименьшее из значений элементов в i-ой строке. Вывести на экран матрицу A и вектор b. Упорядочить строки матрицы по убыванию элементов первого столбца.

7	Дана действительная квадратная матрица 7×7. Получить вектор b = (b_1, b_2, b_3, b_4, b_5, b_6, b_7), где b_i – это среднеарифметическое значение элементов в i-м столбце. Вывести на экран матрицу A и вектор b. Упорядочить строки матрицы A по возрастанию элементов последнего столбца.
8	Дана вещественная матрица размером 4×6. Упорядочить ее столбцы по убыванию суммы их элементов. Вывести вектор сумм элементов строк, первоначальную и полученную матрицу на экран. Вывести элементы матрицы, которые встречаются в матрице более трех раз (считается, что хотя бы один такой элемент имеется).
9	Дана матрица размером 5×5. Найти скалярное произведение строки, в которой находится наибольший элемент матрицы на столбец с наименьшим элементом. (Скалярное произведение – это сумма произведений координат строки и столбца). Упорядочить столбец с наименьшим элементом по возрастанию элементов, строку с наибольшим элементом – по убыванию элементов.
10	С помощью действительной матрицы A, размером 3×7: в пространстве задано 7 векторов, каждый вектор определяет координаты (a_{1j}, a_{2j}, a_{3j}). Найти вектор, имеющий максимальную норму, если норму вектора определяют по формуле: $\|C\| = \sqrt{C_1^2 + C_2^2 + C_3^2}$, где c_1, c_2, c_3 – координаты вектора c. Вывести на экран норму максимального из векторов, определяемых матрицей A и номер столбца матрицы, задающего этот вектор. Упорядочить строки матрицы по убыванию произведений их элементов.

Лабораторная работа №6. *Разработка консольных приложений по обработке строк*

Цель: Закрепление навыков и овладение приемами разработки программ по обработке символов и строк.

Задание:

Все задания лабораторной работы выполняются по вариантам *(номер варианта определяется по номеру в списке учебной группы)*.

Разработать алгоритм в виде блок-схемы и программу в соответствии со следующей задачей:

Вариант	Условие задачи
1	Вы разрабатываете модуль проверки качества вводимых данных. Программа должна запросить у пользователя ввод произвольной строки текста, после чего определить и вывести: общее количество символов в строке (включая пробелы), количество цифр в строке (используя char.IsDigit), количество букв латинского алфавита (используя char.IsLetter и проверку диапазонов 'A'-'Z', 'a'-'z'), а также количество символов, не являющихся буквами и цифрами. Результаты должны быть представлены в понятном пользователю виде.
2	Вам необходимо реализовать простейший шифр Цезаря для защиты сообщений. Программа запрашивает у пользователя строку для шифрования (на английском языке) и целое число — величину сдвига (от 1 до 25). Затем программа преобразует каждый символ строки: буквы латинского алфавита сдвигаются на заданное количество позиций вперед по алфавиту (с циклическим возвратом в начало), а все остальные символы (цифры, знаки препинания, пробелы) остаются без изменений. Регистр букв должен сохраняться. Программа выводит исходную и зашифрованную строки.
3	Вы пишете лингвистический анализатор, который проверяет, является ли введенная фраза палиндромом (читается одинаково слева направо и справа налево). Программа запрашивает у пользователя строку, затем удаляет из нее все пробелы и знаки препинания (точки, запятые, восклицательные и вопросительные знаки, двоеточия, точки с запятой), приводит все буквы к нижнему регистру и проверяет, совпадает ли полученная строка с её перевернутой версией (используя метод Reverse() или ручной перебор символов). Программа выводит исходную строку и сообщение: "Это

	палиндром" или "Это не палиндром".
4	В школе требуется программа для обучения фонетике. Программа запрашивает у пользователя строку текста на английском языке, затем подсчитывает количество гласных букв (a, e, i, o, u в обоих регистрах) и количество согласных букв (все остальные буквы латинского алфавита). Также программа должна определить, каких букв больше — гласных или согласных, и вывести соответствующее сообщение. Небуквенные символы игнорируются. Результаты выводятся в виде подробного отчета.
5	Вы создаете программу для подготовки текстов к публикации. Программа запрашивает у пользователя строку, в которой между словами может быть произвольное количество пробелов (один, два, три и более), а также могут быть пробелы в начале и в конце строки. Программа должна удалить все лишние пробелы: ведущие и завершающие пробелы удаляются полностью, а между словами оставляется ровно один пробел. Результат выводится на экран. Также программа выводит исходную строку и длину исходной строки для сравнения.
6	Вы разрабатываете генератор необычных паролей. Программа запрашивает у пользователя строку (например, слово или фразу) и преобразует её так, чтобы каждый четный символ (по индексу, начиная с 0) был преобразован в верхний регистр, а каждый нечетный — в нижний регистр. Для латинских букв используются методы char.ToUpper и char.ToLower, все небуквенные символы остаются без изменений. Программа выводит исходную и преобразованную строки, а также демонстрирует результат пользователю.
7	Вы создаете простой текстовый редактор с функцией "найти и заменить". Программа запрашивает у пользователя исходную строку, затем запрашивает подстроку для поиска и подстроку для замены. Программа находит все вхождения подстроки поиска в исходной строке (без учета регистра) и заменяет их на подстроку замены. Регистр при поиске игнорируется, но при замене регистр подстроки замены сохраняется в том виде, в котором ввел пользователь. Программа выводит исходную строку, количество выполненных замен и результирующую строку.
8	Вы пишете программу для лингвистического эксперимента, которая меняет порядок слов в предложении на противоположный. Программа запрашивает у пользователя предложение (слова разделены пробелами, знаки препинания прилегают к словам без пробелов). Программа разделяет строку на слова (используя Split с пробелом в качестве разделителя), затем переставляет слова в обратном порядке и собирает их обратно в строку с одиночными пробелами между словами. Знаки препинания остаются прикрепленными к своим словам. Программа выводит исходное и перевернутое предложения.
9	Вы разрабатываете модуль регистрации пользователей, который проверяет корректность введенного email-адреса. Программа запрашивает у пользователя строку, которая должна представлять собой email-адрес. Программа проверяет следующие условия: адрес содержит ровно один символ '@', до символа '@' есть хотя бы один символ, после '@' есть хотя бы один символ '.', последняя точка не может быть последним символом, и в адресе нет символов, запрещенных для email (пробелы, русские буквы). Если адрес корректен, программа выводит "Email принят", если нет — указывает, какое именно правило нарушено.
10	В корпоративной среде часто требуется создавать аббревиатуры для длинных названий отделов. Программа запрашивает у пользователя фразу из нескольких слов (на английском языке), затем выделяет первую букву каждого слова (если слово начинается с буквы), преобразует эти буквы в верхний регистр и объединяет их в аббревиатуру. Слова в фразе могут разделяться любым количеством пробелов, также могут быть знаки препинания, которые игнорируются (например, "Global. Trading & Co" должно дать аббревиатуру "GTC"). Программа выводит исходную фразу и

	сгенерированную аббревиатуру.

Лабораторная работа №7. *Разработка консольных приложений по созданию собственных методов*

Цель: Закрепление навыков и овладение приемами разработки программ по созданию собственных методов.

Задание:

Все задания лабораторной работы выполняются по вариантам *(номер варианта определяется по номеру в списке учебной группы)*.

Задание №1. Разработать алгоритм в виде блок-схемы и программу в соответствии со следующей задачей:

Вариант	Условие задачи
1	Найти все трехзначные простые числа. Разработать метод, который будет распознавать простые числа и возвращать значение логического типа.
2	Даны два натуральных числа. Выяснить, в каком из них сумма цифр больше. Разработать метод для расчёта суммы цифр натурального числа.
3	Найти периметр треугольника, заданного координатами своих вершин. Разработать метод для расчёта длины отрезка по координатам его вершин.
4	Дано натуральное число N. Определить количество заданной цифры K (вводит пользователь) в системе счисления с основанием M (вводит пользователь). Например, если N=8, K=1, M=2, то это означает, что необходимо найти количество единиц (K=1) в двоичной записи (M=2) числа 8 (N=8). Напишите функцию, которая находит количество цифр в записи числа N с основанием M. Результатом работы функции должно быть число K.
5	Найти площадь треугольника с помощью формулы Герона: $$S = \sqrt{p(p-a)(p-b)(p-c)}$$ где p – полупериметр. Стороны треугольника вводятся с клавиатуры. Реализовать вычисление площади в виде метода, на вход которого подаются три числа, на выходе возвращается площадь треугольника. Перед тем как вычислять площадь треугольника по трем сторонам, необходимо убедиться, что по данным сторонам можно построить треугольник. Если треугольник существует по данным сторонам, то необходимо вывести площадь, а если треугольник не существует, то вывести соответствующее сообщение.
6	Найти НОД (наибольший общий делитель) двух чисел с помощью модифицированного алгоритма Евклида. Для нахождения НОД разработать метод, в который необходимо передавать два числа. Данный метод должен возвращать одно число. Например, для чисел 3430 и 1365 НОД будет равен 35.
7	Найти максимальное число среди трех чисел, которые формируются случайным образом. Для нахождения максимального числа не использовать встроенную функцию max. Разработать собственный метод, который принимает на вход три числа и возвращает одно число – максимальное число из трех возможных. Вызвать метод для пяти троек чисел.
8	Найти «цифровой корень» натуральных чисел. Разработать метод для нахождения суммы цифр числа. Метод принимает на вход одно число. Применить этот метод для нахождения «цифрового корня» пяти натуральных случайных чисел. Цифровой корень числа – это цифра, полученная в результате итеративного процесса суммирования цифр, на каждой итерации которого для подсчета суммы цифр берут результат, полученный на предыдущей итерации. Например, для числа 127 цифровой корень будет равен 1+2+7=10=1+0=1.
9	Разработать метод, который возвращает значение True, если переданное в данный метод число помещается в 8 битную ячейку памяти и значение False, если не помещается. Вызвать эту функцию для 10 случайных натуральных

	чисел в диапазоне от 1 до 300.
10	Даны два натуральных числа. Выяснить, в каком из них больше цифр. Разработать функцию для расчёта количества цифр натурального числа.

Задание №2. Разработать алгоритм в виде блок-схемы и программу в соответствии со следующей задачей:

Вариант	**Условие задачи**
1	Написать метод PowA3(A, B), вычисляющий третью степень числа A и возвращающий её в переменной B (A – входной, B – выходной параметр). С помощью этого метода найти третью степень пяти данных чисел, которые формируются случайным образом.
2	Написать метод PowA234(A, B, C, D), вычисляющий вторую и четвертую степень числа A и возвращающий эти степени соответственно в переменных B, C и D (A – входной, B, C, D – выходные параметры). С помощью этого метода найти вторую, третью и четвертую степень пяти данных чисел, которые вводятся с клавиатуры.
3	Разработать метод Mean (X, Y, AMean, GMean), вычисляющего среднее арифметическое AMean и среднее геометрическое GMean двух положительных чисел X иY (X и Y – входные параметры, AMean и GMean – выходные параметры). С помощью этого метода найти среднее арифметическое и среднее геометрическое для пар (A, B), (A, C), (A, D), если даны A, B, C и D (вводятся с клавиатуры).
4	Разработать метод TrianglePS(a, P, S), вычисляющего по стороне a равностороннего треугольника его периметр P и площадь S (a – входной параметр, P и S – выходные параметры). С помощью этого метода найти периметр и площади трех равносторонних треугольников (стороны формируются случайным образом).
5	Разработать метод DigitCountSum(K, C, S), вычисляющего количество C цифр целого положительного числа K, а также их сумму S (K – входной параметр, C и S – выходные параметры). С помощью этого метода найти количество и сумму цифр для каждого из пяти данных чисел (формируются случайным образом).
6	Разработать метод InvertDigits(K), меняющий порядок следования цифр целого положительного числа K на обратный (K – входной и выходной параметр). С помощь. Этого метода поменять порядок следования цифр на обратный для каждого из пяти данных целых чисел (вводятся с клавиатуры).
7	Разработать метод RootsCount(A, B, C), определяющего количество корней квадратного уравнения $ax^2+bx+c=0$ (a, b и c – вещественные параметры, $a\neq0$). С помощью данного метода найти количество корней для каждого из трех квадратных уравнений с данными коэффициентами (коэффициенты формируются случайным образом).
8	Разработать функцию RingS(R_1, R_2), находящего площадь кольца, заключенного между двумя окружностями с общим центром и радиусами R_1 и R_2 (R_1, R_2 – вещественные числа, $R_1>R_2$). С помощью данного метода найти площади трех колец, для которых даны радиусы (формируются случайным образом).
9	Разработать метод DigitN(K, N), возвращающего N-ю цифру целого положительного числа K (все цифры числа нумеруются справа налево). Если количество цифр в числе K меньше N, то метод возвращает значение -1. Применить данный метод для пяти данных целых положительных чисел (формируются случайным образом) с заданным параметром N, который вводится с клавиатуры.
10	Разработать метод SumRange(A, B), находящего сумму всех целых чисел от A и до B включительно (A и B – целые числа). Если A>B, то метод возвращает число 0. С помощью метода найти суммы чисел от A до B и от B до C, если даны числа A, B, C (вводятся с клавиатуры).

Задание №3. Разработать алгоритм в виде блок-схемы и программу в соответствии со следующей задачей:

Вариант	Условие задачи
1	Алгоритм вычисления функции F(n) задан следующими соотношениями: F(n) = 1 при n = 1 F(n) = 2·F(n–1) + n + 3, если n > 1 Чему равно значение функции F(19)?
2	Алгоритм вычисления функции F(n) задан следующими соотношениями: F(n) = 3 при n = 1 F(n) = 2·F(n–1) – n + 1, если n > 1 Чему равно значение функции F(21)?
3	Алгоритм вычисления функции F(n) задан следующими соотношениями: F(n) = 2 при n = 1 F(n) = F(n–1) + $5n^2$, если n > 1 Чему равно значение функции F(39)?
4	Алгоритм вычисления функции F(n) задан следующими соотношениями: F(n) = 2 при n ≤ 1 F(n) = F(n–1) + F(n–2) + 2n + 4, если n > 1 Чему равно значение функции F(25)?
5	Алгоритм вычисления функции F(n) задан следующими соотношениями: F(n) = 3 при n ≤ 1 F(n) = F(n–1) + 2·F(n–2) – 5, если n > 1 Чему равно значение функции F(22)?
6	Алгоритм вычисления функции F(n) задан следующими соотношениями: F(n) = 2 при n ≤ 1 F(n) = F(n–1) + F(n–2) + 4n, если n > 1 Чему равно значение функции F(24)?
7	Алгоритм вычисления функции F(n) задан следующими соотношениями: F(n) = n при n > 15 F(n) = 2·F(n+1) + 5n + 2, если n ≤ 15 Чему равно значение функции F(2)?
8	Алгоритм вычисления функции F(n) задан следующими соотношениями: F(n) = n при n > 18 F(n) = 3·F(n+1) + n + 8, если n ≤ 18 Чему равно значение функции F(9)?
9	Алгоритм вычисления функции F(n) задан следующими соотношениями: F(n) = n – 3 при n > 16 F(n) = 2·F(n+1) + 2n + 3, если n ≤ 16 Чему равно значение функции F(2)?
10	Алгоритм вычисления функции F(n) задан следующими соотношениями: F(n) = 2n – 5 при n > 12 F(n) = 2·F(n+2) + n – 4, если n ≤ 12 Чему равно значение функции F(1)?

Задание №4. Разработать алгоритм в виде блок-схемы и программу в соответствии со следующей задачей:

Вариант	Условие задачи
1	Алгоритм вычисления функции F(n) задан следующими соотношениями: F(n) = 2·n при n < 3 F(n) = 3n + 5 + F(n–2), если n ≥ 3 и чётно, F(n) = n + 2·F(n–6), если n ≥3 и нечётно. Чему равно значение функции F(61)?
2	Алгоритм вычисления функции F(n) задан следующими соотношениями: F(n) = –n при n < 0 F(n) = 2n + 1 + F(n–3), если n ≥ 0 и чётно, F(n) = 4n + 2·F(n–4), если n ≥ 0 и нечётно. Чему равно значение функции F(33)?

3	Алгоритм вычисления функции F(n) задан следующими соотношениями: F(n) = 5–n при n < 5 F(n) = 4· (n – 5)·F(n–5), если n ≥ 5 и делится на 3, F(n) = 3n + 2·F(n–1) + F(n–2), если n ≥ 5 и не делится на 3. Чему равно значение функции F(20)?
4	Алгоритм вычисления функции F(n) задан следующими соотношениями: F(n) = 1+2n при n < 5 F(n) = 2·(n + 1)·F(n–2), если n ≥ 5 и делится на 3, F(n) = 2·n + 1 + F(n–1) + 2·F(n–2), если n ≥ 5 и не делится на 3. Чему равно значение функции F(15)?
5	Алгоритм вычисления функции F(n) задан следующими соотношениями: F(n) = n + 3 при n < 3 F(n) = (n + 2)·F(n–4), если n ≥ 3 и делится на 3, F(n) = n + F(n–1) + 2·F(n–2), если n ≥ 3 и не делится на 3. Чему равно значение функции F(20)?
6	Алгоритм вычисления функции F(n) задан следующими соотношениями: F(n) = 1 при n ≤ 1; F(n) = n · F(n – 1) при чётных n > 1; F(n) = n + F(n – 2) при нечётных n > 1; Определите значение F(84).
7	Алгоритм вычисления функции F(n) задан следующими соотношениями: F(n) = 1 при n ≤ 1; F(n) = n + F(n – 1) при чётных n > 1; F(n) = n · n + F(n – 2) при нечётных n > 1; Определите значение F(80).
8	Алгоритм вычисления функции F(n), где n – целое число, задан следующими соотношениями: F(n) = n – 1 при n < 4, F(n) = n + 2·F(n – 1), когда n ≥ 4 и кратно 3, F(n) = F(n – 2) + F(n – 3), когда n ≥ 4 и не кратно 3. Чему равна сумма цифр значения F(25)?
9	Алгоритм вычисления функции F(n), где n – целое число, задан следующими соотношениями: F(n) = 1 при n = 0, F(n) = 2·F(1 – n) + 3·F(n – 1) + 2, когда n > 0, F(n) = –F(–n), когда n < 0. Чему равна сумма цифр значения F(50)?
10	Алгоритм вычисления функции F(n), где n – целое число, задан следующими соотношениями: F(n) = 5 при n = 0, F(n) = 3·F(n – 4) , когда n > 0, F(n) = F(n + 3), когда n < 0. Чему равно значение F(43)?

Лабораторная работа №8. *Разработка консольных приложений по работе с файлами*

Цель: Закрепление навыков и овладение приемами разработки программ по работе с файлами.

Задание:

Все задания лабораторной работы выполняются по вариантам *(номер варианта определяется по номеру в списке учебной группы)*.

Задание №1. Условие задания совпадает с **заданием №3 лабораторной работы №4.** Изменить код программы так, чтобы числа, которые вводит пользователь были записаны в файл (каждое число на новой строке). После всех чисел на новую строку вывести результат вычисления в соответствии с задачей.

Задание №2. Разработать алгоритм в виде блок-схемы и программу в соответствии со следующей задачей:

Вариант	**Условие задачи**

1	Вы разрабатываете программу для статистической обработки результатов экспериментов. Пользователь должен самостоятельно создать текстовый файл numbers.txt, в котором в каждой строке записано одно целое число (количество строк произвольное). Программа читает все числа из файла, вычисляет их сумму, количество, среднее арифметическое (с помощью Math.Round до двух знаков), минимальное и максимальное значение (используя Math.Min и Math.Max в цикле). Результаты программа записывает в файл analysis.txt в виде отчета. Если файл numbers.txt отсутствует или пуст, программа выводит соответствующее сообщение об ошибке.
2	Вы создаете утилиту для очистки лог-файлов. Пользователь должен подготовить файл input.txt с произвольными текстовыми строками (не менее 10 строк разной длины). Программа запрашивает у пользователя целое число L — минимальную длину строки, которую нужно оставить. Затем программа читает файл, отбирает все строки, длина которых больше или равна L (используя свойство Length), и записывает их в файл filtered.txt в том же порядке. Также программа выводит на экран количество удаленных строк. Если исходный файл не найден, программа сообщает об ошибке.
3	Вы пишете простой редактор конфигурационных файлов. Пользователь создает текстовый файл config.txt с произвольным содержимым. Программа запрашивает у пользователя слово для поиска и слово для замены, затем читает весь файл, заменяет все вхождения искомого слова (с учетом регистра) на слово замены и сохраняет результат в новый файл updated_config.txt. Также программа выводит на экран количество произведенных замен. Если исходный файл отсутствует, программа выводит сообщение об ошибке и завершает работу.
4	В учебном заведении ведется ведомость успеваемости. Пользователь создает файл grades.txt, где каждая строка имеет формат: "Фамилия Имя: оценка1, оценка2, оценка3" (например, "Иванов Иван: 4, 5, 3"). Программа читает файл, для каждого студента вычисляет средний балл (как сумму всех оценок, деленную на их количество) с помощью Math.Round до одного знака, определяет, сдал ли студент экзамен (средний балл >= 4), и записывает в файл report.txt результат в формате: "Фамилия Имя: средний балл X.XX — Сдал/Не сдал". В конце отчета программа добавляет общий процент сдавших студентов.
5	Вы создаете программу для объединения данных с разных датчиков. Пользователь самостоятельно создает два текстовых файла: data1.txt и data2.txt, каждый из которых содержит целые числа (по одному на строку, количество строк может различаться). Программа читает оба файла, объединяет все числа из обоих файлов в один список, сортирует их по возрастанию (используя Array.Sort или List.Sort), удаляет дубликаты (каждое число встречается только один раз) и записывает результат в файл merged_sorted.txt (каждое число на новой строке). Если один из файлов отсутствует, программа выводит сообщение об ошибке.
6	Вы разрабатываете инструмент для криптоанализа. Пользователь создает текстовый файл text.txt с произвольным содержанием (например, абзац книги на английском языке). Программа читает файл, подсчитывает частоту каждого символа (буквы в нижнем регистре, цифры, знаки препинания — отдельно), игнорируя разницу между прописными и строчными буквами (приводя все буквы к нижнему регистру с помощью char.ToLower). Результаты программа записывает в файл frequency.txt в формате: "Символ 'X' встретился Y раз (Z.ZZ%)", где процент вычисляется от общего количества символов (исключая символы перевода строки). Символы сортируются по убыванию частоты.
7	Вы работаете с табличными данными в научном проекте. Пользователь создает файл matrix.txt, в котором первая строка содержит два целых числа:

	количество строк M и количество столбцов N (через пробел), а следующие M строк содержат N целых чисел каждая, разделенных пробелами. Программа читает матрицу, транспонирует её (строки становятся столбцами) и записывает результат в файл transposed.txt в том же формате (сначала строка с размерами N и M, затем N строк по M чисел). Если формат файла нарушен или файл не найден, программа выводит сообщение об ошибке.
8	Вы создаете бизнес-аналитику для небольшого магазина. Пользователь создает CSV-файл sales.csv (разделитель — точка с запятой) со следующими колонками: "Товар; Цена; Количество". Пример строки: "Мыло; 45.50; 3". Программа читает файл, для каждой строки вычисляет сумму продажи (Цена × Количество) с помощью Math.Round до двух знаков, находит товар с максимальной выручкой (используя Math.Max в цикле), общую выручку по всем товарам и записывает в файл sales_report.txt подробный отчет: по каждому товару — выручка, а в конце — максимальный товар и общая сумма.
9	Вы пишете утилиту для дедупликации и сортировки данных. Пользователь создает текстовый файл source.txt с произвольным количеством строк (не менее 6). Программа читает файл, записывает все строки с четными номерами (2, 4, 6, ...) в файл even_lines.txt, а строки с нечетными номерами (1, 3, 5, ...) — в файл odd_lines.txt. В каждый из выходных файлов строки записываются в обратном порядке (последняя прочитанная становится первой). Также программа выводит на экран количество строк в исходном файле. Если исходный файл отсутствует, программа сообщает об ошибке.
10	Вы разрабатываете систему проверки данных в архиве. Пользователь создает файл dates.txt, в котором каждая строка содержит дату в формате ДД.ММ.ГГГГ (например, 15.07.2024). Некоторые даты могут быть некорректными (например, 32.13.2024 или 29.02.2023). Программа читает файл, проверяет каждую дату с помощью DateTime.TryParseExact (формат "dd.MM.yyyy"), корректные даты записывает в файл valid_dates.txt в том же формате, а некорректные строки — в файл invalid_dates.txt с указанием причины ошибки (неверный день, месяц или формат). В конце программа выводит на экран количество корректных и некорректных записей.

РАЗДЕЛ 3. ОСНОВЫ ОБЪЕКТНО-ОРИЕНТИРОВАННОГО ПРОГРАММИРОВАНИЯ

Лабораторная работа №9. Разработка приложений по работе с классами

Цель: Закрепление навыков и овладение приемами разработки программ по работе с классами.

Задание:

Все задания лабораторной работы выполняются по вариантам *(номер варианта определяется по номеру в списке учебной группы)*.

Разработать программу в соответствии со следующей задачей:

Вариант	Условие задачи
1	Создайте класс Calculator. Реализуйте в нём три перегруженных метода Add: первый складывает два целых числа, второй — три целых числа, третий — два вещественных числа. Также реализуйте перегруженные методы Multiply (два целых, три целых, два вещественных). Добавьте метод GetInfo, который выводит информацию о том, сколько перегрузок имеет каждый метод. В конструкторе класса выводите сообщение «Калькулятор создан». В основной программе создайте объект Calculator и продемонстрируйте работу всех перегруженных методов, выводя результаты на экран.
2	Создайте базовый класс Rectangle с полями Width и Height (тип double). Реализуйте конструктор, принимающий ширину и высоту, методы GetArea (площадь) и GetPerimeter (периметр), а также виртуальный метод PrintInfo, выводящий размеры, площадь и периметр. Создайте производный класс Square, который принимает в конструкторе одну сторону и вызывает

	конструктор базового класса с двумя одинаковыми сторонами. Переопределите метод PrintInfo в классе Square, добавив слово «Квадрат» в вывод. В основной программе создайте объекты Rectangle и Square, вызовите для каждого метод PrintInfo.
3	Создайте базовый класс Vehicle с полями Brand (марка) и Speed (скорость, км/ч). Реализуйте конструктор с двумя параметрами и виртуальный метод Move, который выводит строку «Транспортное средство движется». Создайте производные классы Car (добавляет поле NumberOfDoors) и Motorcycle (добавляет поле HasSidecar). Переопределите метод Move в каждом классе: для Car — «Автомобиль [марка] едет со скоростью [скорость] км/ч», для Motorcycle — «Мотоцикл [марка] разгоняется до [скорость] км/ч». В основной программе создайте массив из объектов Vehicle, заполните его Car и Motorcycle, затем в цикле вызовите метод Move для каждого.
4	Создайте класс Employee с полями: Name (имя), Position (должность), Salary (зарплата). Реализуйте три конструктора: первый принимает только имя (должность по умолчанию «Стажёр», зарплата 20000), второй принимает имя и должность (зарплата по умолчанию 30000), третий принимает все три параметра. Добавьте метод PrintInfo, выводящий информацию о сотруднике. Создайте в основной программе три объекта Employee, используя разные конструкторы, и вызовите для каждого PrintInfo.
5	Создайте класс BankAccount с полями AccountNumber (уникальный номер, генерируется статическим счётчиком), OwnerName (имя владельца), Balance (баланс). Реализуйте конструктор, принимающий имя владельца (баланс по умолчанию 0). Добавьте перегруженные методы Deposit: первый принимает сумму и добавляет её к балансу, второй принимает сумму и валюту (строку «USD» или «EUR»), умножая сумму на курс (1 USD = 100, 1 EUR = 110) перед добавлением. Добавьте метод PrintBalance, выводящий баланс. В основной программе создайте счёт, выполните пополнение через оба варианта метода Deposit и выведите итоговый баланс.
6	Создайте абстрактный класс Shape с виртуальным методом CalculateArea, который возвращает 0. Создайте производные классы: Circle (с полем Radius, площадь = πR^2 с использованием Math.PI), Rectangle (поля Width и Height, площадь = ширина × высота), Triangle (поля Base и Height, площадь = ½ × основание × высота). В каждом производном классе переопределите метод CalculateArea. Добавьте конструкторы для инициализации полей и метод PrintArea, выводящий название фигуры и её площадь. В основной программе создайте список из разных фигур, вычислите и выведите площадь каждой.
7	Создайте класс TextString, который содержит поле Value (строка). Реализуйте конструктор, принимающий строку. Добавьте перегруженные методы Replace: первый принимает два символа (старый и новый) и заменяет все вхождения одного символа на другой, второй принимает две строки (подстроку для поиска и подстроку для замены) и выполняет замену. Оба метода возвращают новую строку, не изменяя исходную. Также добавьте метод GetLength, возвращающий длину строки, и метод Print. В основной программе создайте объект TextString, продемонстрируйте оба варианта замены и выведите результаты.
8	Создайте класс Timer с полями Hours, Minutes, Seconds. Реализуйте конструктор по умолчанию (устанавливает 00:00:00) и конструктор, принимающий часы, минуты, секунды. Добавьте перегруженные методы SetTime: первый принимает только часы (минуты и секунды обнуляются), второй принимает часы и минуты (секунды обнуляются), третий принимает часы, минуты и секунды. Добавьте метод Tick, который увеличивает время на 1 секунду (с учётом переполнения), и метод PrintTime. В основной программе создайте объект Timer, установите время разными способами,

	выполните несколько вызовов Tick и выведите результаты.
9	Создайте базовый класс Product с полями Name (название), Price (цена), Quantity (количество). Реализуйте конструктор с параметрами и виртуальный метод GetTotalPrice, возвращающий Price × Quantity. Добавьте метод PrintProduct, выводящий информацию о товаре. Создайте производный класс FragileProduct, который добавляет поле InsurancePercent (процент страховки от общей стоимости). Переопределите метод GetTotalPrice, чтобы он возвращал (Price × Quantity) × (1 + InsurancePercent / 100). В основной программе создайте массив из Product и FragileProduct, выведите общую стоимость каждого товара.
10	Создайте класс MathFunctions без полей. Реализуйте в нём статические перегруженные методы Max: первый находит максимум из двух целых чисел, второй — из трёх целых, третий — из двух вещественных. Также реализуйте статические перегруженные методы Min по аналогии. Добавьте нестатический метод SetPrecision, который устанавливает количество знаков после запятой для вывода (поле точности), и метод PrintResult, выводящий результат с заданной точностью. В конструкторе класса устанавливайте точность по умолчанию — 2 знака. В основной программе продемонстрируйте работу методов Max и Min, а затем создайте объект, измените точность и выведите несколько результатов.

РАЗДЕЛ 4. МОДУЛЬНОЕ ПРОГРАММИРОВАНИЕ

Лабораторная работа №10. Модульное программирование. Исключения

Цель: Закрепление навыков и овладение приемами разработки программ по обработке исключительных ситуаций.

Задание:

Все задания лабораторной работы выполняются по вариантам *(номер варианта определяется по номеру в списке учебной группы)*.

Разработать программу в соответствии со следующей задачей:

Вариант	Условие задачи
1	Вы разрабатываете консольный калькулятор. Программа запрашивает у пользователя два числа (целых или вещественных) и операцию (+, -, *, /). При вводе некорректных данных (например, букв вместо чисел) должно возникать исключение FormatException, которое программа перехватывает и предлагает пользователю ввести данные заново. При попытке деления на ноль возникает исключение DivideByZeroException, после чего программа выводит сообщение «Деление на ноль невозможно» и завершает операцию, но не завершает работу программы (пользователь может ввести новые числа). После успешного вычисления результат выводится на экран.
2	Вы создаете программу для работы с массивом целых чисел. Пользователь вводит размер массива (от 1 до 10). Программа создает массив указанного размера и заполняет его случайными числами от 0 до 100. Затем программа запрашивает у пользователя индекс элемента (от 0 до размера-1) и выводит значение элемента по этому индексу. Если пользователь вводит индекс, выходящий за границы массива, возникает исключение IndexOutOfRangeException. Программа должна перехватить это исключение, вывести сообщение «Индекс находится вне границ массива. Допустимые значения от 0 до X», где X — последний допустимый индекс, и запросить индекс повторно. Программа завершается только после успешного получения корректного индекса и вывода значения элемента.
3	Вы пишете программу для парсинга пользовательского ввода. Программа запрашивает у пользователя строку и пытается преобразовать её в целое число. Если преобразование не удается, возникает исключение FormatException или OverflowException. Программа должна перехватывать оба исключения: при FormatException выводить сообщение «Введена

	нечисловая строка», при OverflowException — «Число слишком большое или слишком маленькое для типа int». После каждого некорректного ввода программа повторно запрашивает строку. Цикл продолжается до тех пор, пока не будет введено корректное целое число. После успешного преобразования программа выводит число и его квадрат (с использованием Math.Pow).
4	Вы разрабатываете программу для чтения текстового файла. Программа запрашивает у пользователя имя файла (например, «data.txt») и пытается прочитать все строки из этого файла. Если файл не существует, возникает исключение FileNotFoundException. Программа перехватывает это исключение, выводит сообщение «Файл не найден. Проверьте имя файла», и запрашивает имя файла повторно. Также возможны исключения DirectoryNotFoundException (если указана несуществующая директория) и IOException (другие ошибки ввода-вывода). Программа должна обрабатывать все эти исключения с соответствующими сообщениями. После успешного открытия файла программа выводит количество строк в файле и их содержимое на экран.
5	Вы создаете банковскую программу. Создайте класс BankAccount с полями Balance (баланс) и OwnerName. Реализуйте методы Deposit (пополнение) и Withdraw (снятие средств). При попытке снять сумму, превышающую баланс, метод Withdraw должен генерировать собственное исключение InsufficientFundsException (создайте этот класс, унаследовав его от Exception). Исключение должно содержать сообщение о недостающей сумме. В основной программе запросите у пользователя сумму для снятия, обработайте возможное исключение и выведите информативное сообщение. При успешном снятии выведите новый баланс. Также обработайте возможный FormatException при вводе суммы.
6	Вы разрабатываете систему управления списком задач. Создайте класс TaskItem с полями Title (название) и IsCompleted (выполнена ли задача). В основной программе создайте массив из 5 элементов типа TaskItem, но инициализируйте только первые три элемента (остальные оставьте null). Напишите метод PrintTasks, который проходит по массиву и выводит информацию о каждой задаче. При обращении к null-объекту возникает исключение NullReferenceException. Перехватите это исключение внутри метода, выведите сообщение «Задача не инициализирована (null)» и продолжите обработку остальных элементов. После обработки всех задач программа должна вывести количество успешно выведенных задач и количество пропущенных.
7	Вы создаете иерархию классов: базовый класс Animal с виртуальным методом MakeSound, и производные классы Dog и Cat, переопределяющие этот метод. В основной программе создайте массив типа Animal[], заполните его объектами Dog, Cat и несколькими объектами базового класса Animal. Напишите метод MakeAllSounds, который в цикле приводит каждый элемент к типу Dog (с помощью явного приведения) и вызывает метод Bark (специфичный для Dog). Если элемент не является собакой, возникает исключение InvalidCastException. Перехватите это исключение и выведите сообщение «Это не собака, звук пропущен». Также обработайте ситуацию, когда элемент равен null. В конце выведите количество собак, у которых удалось вызвать метод Bark.
8	Вы создаете класс Circle с полем Radius. В конструкторе класса проверьте, что радиус больше нуля. Если радиус меньше или равен нулю, сгенерируйте исключение ArgumentException с сообщением «Радиус должен быть положительным числом». Также при вычислении площади круга (Math.PI * Radius * Radius) используйте Math.Round для округления до двух знаков. В основной программе запросите у пользователя радиус в цикле. Обработайте возможные исключения: FormatException (если введено не число),

	OverflowException (если число слишком большое), ArgumentException (если радиус не положительный). После успешного создания объекта выведите площадь круга. Программа должна продолжать запрашивать радиус до тех пор, пока не будет введено корректное положительное число.
9	Вы создаете программу для регистрации пользователей. Создайте класс User с полями Name (имя) и Age (возраст). В конструкторе класса проверьте, что возраст находится в диапазоне от 0 до 120 лет. Если возраст выходит за эти пределы, сгенерируйте собственное исключение InvalidAgeException (унаследованное от Exception). Имя пользователя не может быть пустым или состоять только из пробелов — в этом случае также генерируйте ArgumentException (стандартное). В основной программе запросите данные пользователя, обработайте оба типа исключений с выводом соответствующих сообщений. Если данные корректны, создайте объект User и выведите сообщение «Пользователь успешно зарегистрирован». Программа не должна завершаться при ошибке — предлагайте ввести данные заново.
10	Вы изучаете опасности бесконечной рекурсии. Напишите рекурсивный метод Countdown, который принимает целое число n и выводит числа от n до 1, а затем вызывает сам себя с параметром n-1. Если в метод передано отрицательное число, рекурсия должна корректно завершаться (базовый случай). Создайте также ошибочный рекурсивный метод InfiniteRecursion, который вызывает сам себя без условия выхода. Вызовите этот метод в блоке try-catch. Обработайте исключение StackOverflowException (учтите, что в .NET начиная с версии 2.0 это исключение нельзя перехватить обычным способом; задание предполагает демонстрацию понимания — либо используйте RuntimeHelpers.EnsureSufficientExecutionStack, либо объясните в комментариях). Альтернативно: создайте глубокую рекурсию (например, 10000 вызовов), которая вызовет StackOverflowException, и обработайте её. Программа должна вывести сообщение «Обнаружена бесконечная или слишком глубокая рекурсия».

Лабораторная работа №11. Разработка оконных приложений

Цель: Закрепление навыков и овладение приемами разработки программ по обработке массивов.

Задание:

Все задания лабораторной работы выполняются по вариантам *(номер варианта определяется по номеру в списке учебной группы)*.

Задание №1. Разработать программу в соответствии со следующей задачей:

Общая формулировка задачи:

Дана область *(см. вариант)* и координаты точки (вводятся с клавиатуры через компоненты). Определить, принадлежит ли точка с заданными координатами, исходной области, т.е. вывести сообщение *«Точка с координатами (x, y) принадлежит заданной области»*, а в случае, если точка находится за пределами заданной области, то вывести сообщение *«Точка с координатами (x, y) НЕ принадлежит заданной области»*.

На форме разместить изображение заданной области *(в соответствии с вариантом)*. Для размещения использовать компонент PictureBox. Настроить его свойства: Image, SizeMode.

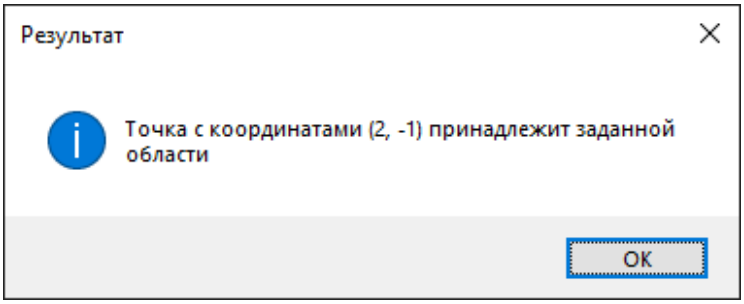
При решении задачи разработать свой метод.

При вводе исходных данных необходимо предусмотреть нажатие клавиши *Enter*. После того, как пользователь ввел первое число, он нажимает клавишу *Enter* и курсор переходит в следующее поле для ввода второго числа. Когда второе число ввели, пользователь может опять нажать клавишу *Enter* и в фокусе должна оказаться кнопка, при нажатии на которую появляется результат вычислений.

При разработке приложения необходимо учитывать:

1. Пользователь может ввести различные исходные числовые данные.
2. При вводе данных пользователь должен вводить только числа (целые и дробные, положительные и отрицательные).
3. Компоненты на форме должны размещаться компактно.
4. Форма должна иметь заголовок.
5. Размер текста для элементов на форме сделать не менее 12.

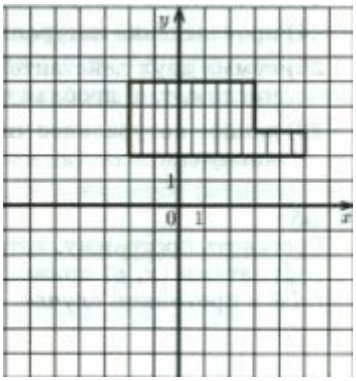
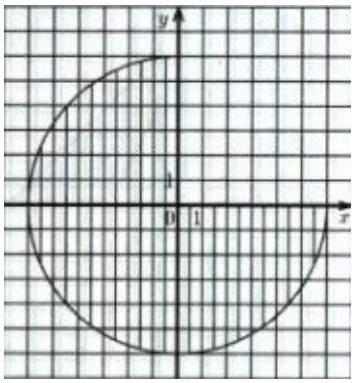
6. Пользователь не должен менять размеры окна, но при этом он может свернуть окно.
7. При запуске приложения форма должна открываться в центре экрана.
8. Результат выводить в виде диалогового окна с указанием сообщения, заголовка окна, иконки и одной кнопки. *Пример,*

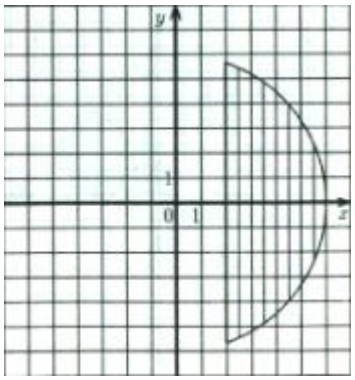
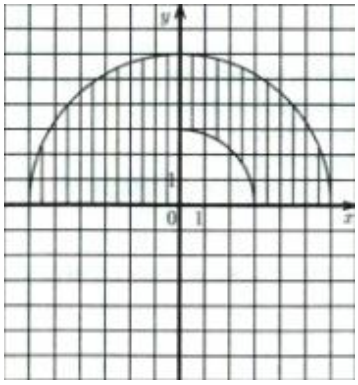
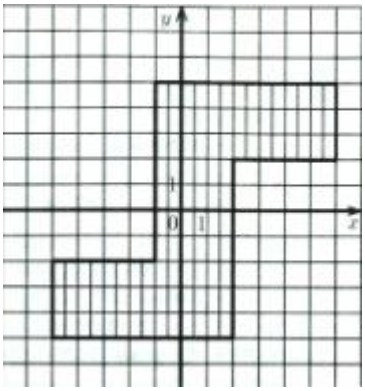


В процессе разработки приложения заполнить таблицу *(для формы и компонентов, которые на форме будут расположены)*:

Компонент	Наименование свойства	Значение свойства
Форма	Name	
	Text	
	…	
TextBox	Name	
	Text	
	…	
Button	Name	
	Text	
	…	
…	…	…

Варианты:

Вариант	Область данных
1	
2	

3	
4	
5	

Задание №2. Разработать приложение Windows Forms для вычисления значения переменных в соответствии с заданием №1 из лабораторной работы №3 (по вариантам). Настроить форму как в первом задании.

При вводе исходных данных необходимо предусмотреть нажатие клавиши Enter. После того, как пользователь ввел первое число, он нажимает клавишу Enter и курсор переходит в следующее поле для ввода второго числа и т.д. Когда ввели последнее число, то пользователь может опять нажать клавишу Enter и в фокусе должна оказаться кнопка, при нажатии на которую появляется результат вычислений.

Задание №3. Разработать приложение Windows Forms в соответствии со следующей задачей:

Вариант	Условие задачи
1, 6	Рассматривается множество целых чисел, принадлежащих отрезку [1012; 9638], которые делятся на 3 и не делятся на 11, 13, 17 и 19. Все найденные числа записать в список (ListBox) и в файл. Имя файла указывает пользователь через текстовое поле. Найдите количество таких чисел и максимальное из них и выведите их в текстовый компонент, который должен быть многострочным. Количество и максимальное число выводятся на разных строках этого компонента с пометкой, что это за число в формате «Количество: значение» или «Максимум: значение».
2, 7	Рассматривается множество целых чисел, принадлежащих отрезку [1200;11200], которые делятся на 5 и не делятся на 7, 13, 17 и 19. Все найденные числа записать в список (ListBox) и в файл. Имя файла указывает пользователь через текстовое поле. Найдите количество таких чисел и минимальное из них и выведите их в текстовый компонент, который должен

	быть многострочным. В ответе запишите два числа через пробел: сначала количество, затем минимальное число. Количество и минимальное число выводятся на разных строках этого компонента с пометкой, что это за число в формате «Количество: значение» или «Минимум: значение».
3, 8	Рассматривается множество целых чисел, принадлежащих отрезку [1512;13202], которые делятся на 7 и не делятся на 11, 13, 17 и 23. Все найденные числа записать в список (ListBox) и в файл. Имя файла указывает пользователь через текстовое поле. Найдите количество таких чисел и максимальное из них и выведите их в текстовый компонент, который должен быть многострочным. В ответе запишите два числа через пробел: сначала количество, затем максимальное число. Количество и максимальное число выводятся на разных строках этого компонента с пометкой, что это за число в формате «Количество: значение» или «Максимум: значение».
4, 9	Рассматривается множество целых чисел, принадлежащих отрезку [1098; 13765], которые делятся на 2 и не делятся на 7, 11, 13 и 23. Все найденные числа записать в список (ListBox) и в файл. Имя файла указывает пользователь через текстовое поле. Найдите количество таких чисел и минимальное из них и выведите их в текстовый компонент, который должен быть многострочным. В ответе запишите два числа через пробел: сначала количество, затем минимальное число. Количество и минимальное число выводятся на разных строках этого компонента с пометкой, что это за число в формате «Количество: значение» или «Минимум: значение».
5, 10	Рассматривается множество целых чисел, принадлежащих отрезку [1156; 12209], которые делятся на 2 или на 5 и не делятся на 7, 13, 17 и 23. Все найденные числа записать в список (ListBox) и в файл. Имя файла указывает пользователь через текстовое поле. Найдите количество таких чисел и максимальное из них и выведите их в текстовый компонент, который должен быть многострочным. В ответе запишите два числа через пробел: сначала количество, затем максимальное число. Количество и максимальное число выводятся на разных строках этого компонента с пометкой, что это за число в формате «Количество: значение» или «Максимум: значение».

Задание №4. Разработать приложение Windows Forms в соответствии со следующей задачей:

Вариант	Условие задачи
1, 6	Рассматривается множество целых чисел, принадлежащих числовому отрезку [697; 3458], которые удовлетворяют следующим условиям: а) Число в шестнадцатеричной записи оканчивается цифрой «E»; б) Число в семеричной записи и в восьмеричной записи оканчивается на одну цифру. Все найденные числа вывести в многострочный текстовый компонент. Найдите сумму таких чисел и их количество и выведите их в разные текстовые компоненты.
2, 7	Рассматривается множество целых чисел, принадлежащих числовому отрезку [99; 999], которые удовлетворяют следующим условиям: а) Число в шестнадцатеричной записи оканчивается цифрой «9»; б) Число в девятеричной записи оканчивается на цифру «8». Все найденные числа вывести в многострочный текстовый компонент Найдите сумму таких чисел и их количество и выведите их в разные текстовые компоненты.
3, 8	Рассматривается множество четных целых чисел, принадлежащих числовому отрезку [100; 1000], которые удовлетворяют следующим условиям: а) Число в шестнадцатеричной записи оканчивается цифрой «0»; б) Число не делится на 3. Все найденные числа вывести в многострочный текстовый компонент Найдите сумму таких чисел и их количество и выведите их в разные текстовые компоненты.

4, 9	Рассматривается множество целых чисел, принадлежащих числовому отрезку [5; 10000] с шагом 5, которые удовлетворяют следующим условиям: а) Число в шестнадцатеричной записи оканчивается цифрой «A»; б) Число не делится на 7, но делится на 5. Все найденные числа вывести в многострочный текстовый компонент Найдите сумму таких чисел и их количество и выведите их в разные текстовые компоненты.
5, 10	Рассматривается множество целых чисел, принадлежащих числовому отрезку [-5000; 5000], которые удовлетворяют следующим условиям: а) Число в шестнадцатеричной записи оканчивается цифрой «B»; б) Число не делится на 6, но делится на 5 и на 7. Все найденные числа вывести в многострочный текстовый компонент Найдите количество таких чисел и максимальное их них (по модулю) и выведите их в разные текстовые компоненты.

ПРИЛОЖЕНИЕ Б

Вопросы к зачёту с оценкой

Индикатор ЗНАТЬ:

ОК 02. Использовать современные средства поиска, анализа и интерпретации информации и информационные технологии для выполнения задач профессиональной деятельности

1. Установите правильную последовательность этапов решения задачи с помощью компьютера
А) Разработка алгоритма
Б) Анализ и постановка задачи
В) Написание программы на языке программирования
Г) Отладка и тестирование программы
2. Установите правильную последовательность действий при отладке программы
А) Исправление найденной ошибки
Б) Воспроизведение ситуации, при которой возникает ошибка
В) Повторное тестирование после исправления
Г) Локализация места возникновения ошибки
3. Установите правильную последовательность этапов работы с файлом в языке C#
А) Закрытие файла
Б) Открытие файла (создание потока)
В) Чтение или запись данных
Г) Проверка успешности открытия файла
4. Установите правильную последовательность передачи управления при вызове функции в структурном программировании
А) Выполнение тела функции
Б) Передача параметров (аргументов) функции
В) Возврат значения из функции в вызывающую программу
Г) Передача управления вызываемой функции
5. Установите правильную последовательность преобразования данных при работе с массивами в языке C#
А) Обращение к элементу массива по индексу
Б) Объявление массива с указанием типа и размера
В) Инициализация массива значениями
Г) Выделение памяти под массив (с помощью оператора new)
6. Установите правильную последовательность этапов модульного программирования при создании проекта
А) Компиляция каждого модуля отдельно
Б) Подключение модулей друг к другу (директивы using или import)
В) Разбиение программы на логически законченные модули
Г) Линковка (сборка) модулей в единое приложение
7. Установите правильную последовательность событий при наследовании классов в объектно-ориентированном программировании
А) Вызов конструктора производного класса
Б) Вызов конструктора базового класса
В) Инициализация полей производного класса
Г) Создание объекта производного класса (вызов оператора new)
8. Установите соответствие между понятием алгоритмизации и его определением

Понятие алгоритмизации	Определение
А) Алгоритм	1. Процесс поиска и исправления ошибок в программе
Б) Блок-схема	2. Конечная последовательность команд для решения задачи
В) Отладка	3. Совокупность входных данных для проверки программы
Г) Тестовый набор	4. Пошаговое выполнение программы с остановками
	5. Графическое представление алгоритма

9. Установите соответствие между парадигмой программирования и её характерной чертой

Парадигма программирования	Характерная черта
А) Структурное программирование	1. Программа состоит из классов и объектов с наследованием и полиморфизмом
Б) Объектно-ориентированное программирование	2. Запрет на использование оператора безусловного перехода
В) Функциональное программирование	3. Программа описывается в терминах фактов и правил
Г) Логическое программирование	4. Функции являются объектами первого класса, отсутствие побочных эффектов

10. Установите соответствие между простым типом данных в C# и его описанием

Типы данных	Описание
А) int	1. Хранит один символ в кодировке Unicode
Б) float	2. Хранит целое число со знаком
В) bool	3. Хранит логическое значение (истина или ложь)
Г) char	4. Хранит число с плавающей точкой одинарной точности
	5. Хранит текстовую строку

11. Установите соответствие между конструкцией структурного программирования и её кратким описанием

Конструкция	Описание
А) Ветвление (if-else)	1. Повторение блока, пока условие истинно, проверка в начале
Б) Цикл с предусловием (while)	2. Выбор одного из нескольких вариантов по значению выражения
В) Цикл с постусловием (do-while)	3. Выполнение блока, после чего проверка условия продолжения
Г) Множественный выбор (switch)	4. Выполнение одного блока, если условие истинно, и другого – если ложно

12. Установите соответствие между принципом объектно-ориентированного программирования и его сущностью

Принцип ООП	Описание
А) Инкапсуляция	1. Сокрытие внутренних деталей реализации и объединение данных с методами
Б) Наследование	2. Возможность существования одного имени метода с разными реализациями
В) Полиморфизм	3. Выделение существенных характеристик объекта, игнорируя несущественные
Г) Абстракция	4. Создание нового класса на основе существующего
	5. Возможность создания нескольких объектов одного класса

13. Установите соответствие между этапом обработки файла и выполняемым действием (на примере C#)

Этап обработки файла	Действия
А) Открытие файла для чтения	1. Создание объекта StreamReader с указанием пути к файлу
Б) Чтение строки из файла	2. Вызов метода Close() или использование using
В) Проверка конца файла	3. Вызов метода ReadLine() или ReadToEnd()
Г) Закрытие файла	4. Проверка свойства EndOfStream или сравнение с null

14. Установите соответствие между понятием модульного программирования и его описанием

Понятия	Описание
А) Модуль	1. Результирующий исполняемый или библиотечный файл (.exe или .dll)
Б) Пространство имён	2. Логически сгруппированный код в отдельном файле или разделе
В) Директива using	3. Способ организации имён типов во избежание конфликтов
Г) Сборка	4. Указание на использование типов из другого пространства имён
	5. Переменная, доступная из любого модуля

15. Установите соответствие между событием в оконном приложении и его типичным обработчиком (Windows Forms / WPF)

Событие в оконном приложении	Обработчик
А) Нажатие кнопки мыши на элементе	1. Событие Load (в Window или Form)
Б) Изменение текста в текстовом поле	2. Событие KeyDown или KeyPress
В) Загрузка главного окна	3. Событие Click (у кнопки) или MouseDown
Г) Нажатие клавиши на клавиатуре	4. Событие TextChanged
	5. Событие Closed (при закрытии окна)

ПК 2.1. Осуществлять установку и настройку отдельных программных, программно-аппаратных средств защиты информации

1. Установите правильную последовательность обработки исходного кода на C# в среде .NET

А) Выполнение промежуточного кода (CIL) с помощью JIT-компилятора

Б) Написание исходного кода на C#

В) Преобразование CIL в машинный код

Г) Компиляция исходного кода в промежуточный код (CIL)

2. Установите правильную последовательность создания и использования объекта в объектно-ориентированном программировании

А) Определение класса

Б) Вызов методов созданного объекта

В) Создание объекта (экземпляра класса) с помощью конструктора

Г) Уничтожение объекта (освобождение ресурсов)

3. Установите правильную последовательность разработки оконного приложения с графическим интерфейсом

А) Размещение элементов управления (кнопки, поля ввода) на форме

Б) Написание обработчиков событий для элементов управления

В) Создание проекта оконного приложения и главного окна (формы)

Г) Компиляция и запуск приложения для тестирования интерфейса

4. Установите правильную последовательность выполнения основных конструкций структурного программирования в линейном участке кода

А) Выполнение тела цикла

Б) Выполнение последовательных операторов (следование)

В) Проверка условия в ветвлении (if-else)

Г) Переход к следующему оператору после окончания цикла

5. Установите правильную последовательность шагов при тестовом контроле программы методом «белого ящика»

А) Разработка тестовых наборов данных

Б) Анализ исходного кода (путей выполнения, ветвлений, циклов)

В) Запуск тестов и сравнение фактических результатов с ожидаемыми

Г) Определение критериев полноты тестирования (например, покрытие ветвей)

6. Установите правильную последовательность обработки исключительных ситуаций (исключений) в C#

А) Выполнение блока catch (обработка ошибки)

Б) Возникновение исключительной ситуации в блоке try

В) Выполнение блока finally (освобождение ресурсов)

Г) Передача управления блоку catch, соответствующему типу исключения

7. Установите правильную последовательность шагов при полиморфизме подтипов (виртуальные методы) в C#

А) Вызов метода через ссылку на базовый класс

Б) Определение базового класса с виртуальным методом

В) Переопределение метода в производном классе

Г) Определение во время выполнения, какой именно метод (базовый или переопределённый) следует вызвать

8. Установите соответствие между этапом компиляции и выполнения программы на C# и его описанием

Этап компиляции	Описание
А) Исходный код (C#)	1. Преобразование промежуточного кода в код целевого процессора во время выполнения
Б) Промежуточный язык (CIL/MSIL)	2. Текст программы, написанный человеком на языке высокого уровня
В) JIT-компиляция	3. Код, который выполняется непосредственно процессором
Г) Машинный код	4. Переносимый код, генерируемый компилятором C#

9. Установите соответствие между видом тестирования и его характеристикой

Вид тестирования	Характеристика
А) Модульное тестирование (Unit-тесты)	1. Проверка взаимодействия между несколькими модулями или классами
Б) Интеграционное тестирование	2. Проверка программы в целом на соответствие требованиям заказчика
В) Системное тестирование	3. Тестирование отдельных функций, методов или классов изолированно
Г) Приёмочное тестирование	4. Проверка производительности и нагрузки на систему
	5. Тестирование полностью собранной системы в условиях, приближенных к реальным

10. Установите соответствие между способом передачи параметров в функцию и его поведением (на примере C#)

Способ передачи параметра	Описание
А) value	1. Позволяет передать переменное количество однотипных аргументов
Б) ref	2. Функция получает копию аргумента, оригинал не изменяется
В) out	3. Параметр должен быть обязательно инициализирован внутри функции до возврата
Г) params	4. Функция работает непосредственно с переменной-аргументом
	5. Параметр передаётся только для чтения, изменение запрещено

11. Установите соответствие между видом цикла в C# и его особенностью

Вид цикла	Особенность
А) for	1. Тело цикла выполняется как минимум один раз
Б) foreach	2. Идеален для перебора коллекций или массивов
В) while	3. Количество итераций обычно известно заранее (счётный цикл)
Г) do-while	4. Условие проверяется перед каждым выполнением тела цикла

12. Установите соответствие между видом исключения в C# и типичной ситуацией возникновения

Вид исключения	Ситуация
А) NullReferenceException	1. Попытка деления целого числа на ноль
Б) IndexOutOfRangeException	2. Обращение к файлу, которого нет на диске
В) DivideByZeroException	3. Вызов метода у переменной, которая не ссылается ни на какой объект
Г) FileNotFoundException	4. Выход за границы массива или списка
	5. Попытка преобразования строки в число с некорректным форматом

13. Установите соответствие между модификатором доступа в C# и уровнем видимости

Модификатор доступа	Уровень видимости
А) public	1. Доступен только внутри текущего класса
Б) private	2. Доступен внутри текущей сборки (проекта)
В) protected	3. Доступен из любого места (другие классы, сборки)
Г) internal	4. Доступен внутри текущего класса и в производных классах (даже из другой сборки)
	5. Доступен только внутри текущего метода

14. Установите соответствие между понятием указателей (или ссылок) в программировании и его описанием

Понятие указателей (ссылок)	Описание
А) Ссылка (reference)	1. Значение, которое не указывает ни на какой объект
Б) Указатель (pointer)	2. Переменная, хранящая адрес другой переменной в памяти
В) Разыменование	3. Переменная, которая косвенно обращается к объекту (без арифметики адресов)
Г) Нулевая ссылка (null)	4. Операция получения объекта по адресу или ссылке
	5. Автоматическое увеличение адреса на размер типа данных

15. Установите соответствие между компонентом оконного приложения (WPF или Windows Forms) и его основным назначением

Компоненты оконного приложения	Назначение
А) Button	1. Отображение статического текста, который пользователь не может редактировать
Б) TextBox	2. Запуск действия при щелчке пользователя
В) ListBox / ComboBox	3. Выбор одного или нескольких элементов из выпадающего или прокручиваемого списка
Г) Label	4. Ввод или редактирование однострочного или многострочного текста

Индикатор УМЕТЬ:

ОК 02. Использовать современные средства поиска, анализа и интерпретации информации и информационные технологии для выполнения задач профессиональной деятельности

1. Выберете правильный ответ из предложенных вариантов и обоснуйте свой выбор. Что понимается под конечностью алгоритма?

А) Алгоритм должен быть записан на понятном человеку языке

Б) Алгоритм обязательно содержит цикл

В) Алгоритм должен завершаться за конечное число шагов

Г) Алгоритм должен работать только с целыми числами

2. Выберете правильный ответ из предложенных вариантов и обоснуйте свой выбор. Какая парадигма программирования в первую очередь опирается на классы, объекты, наследование и полиморфизм?

А) Функциональное программирование

Б) Логическое программирование

В) Объектно-ориентированное программирование

Г) Структурное программирование

3. Выберете правильный ответ из предложенных вариантов и обоснуйте свой выбор. В языке C# какой тип данных предназначен для хранения логического значения (истина/ложь)?

А) int

Б) float

В) bool

Г) char

4. Выберете правильный ответ из предложенных вариантов и обоснуйте свой выбор. Что из перечисленного является корректным способом передачи параметра по ссылке в метод на C# (чтобы изменения внутри метода отразились на переменной вызывающего кода)?

А) Использование ключевого слова out

Б) Использование ключевого слова ref

В) Передача параметра без ключевых слов

Г) Использование ключевого слова params

5. Выберете правильный ответ из предложенных вариантов и обоснуйте свой выбор. Какой принцип объектно-ориентированного программирования напрямую обеспечивает сокрытие внутренних данных класса и доступ к ним только через методы?

А) Наследование

Б) Полиморфизм

В) Абстракция

Г) Инкапсуляция

6. Выберете правильный ответ из предложенных вариантов и обоснуйте свой выбор. В модульном программировании что обычно понимается под модулем?

А) Исполняемый файл

Б) Логически законченная часть программы, часто выделенная в отдельный файл

В) Главная функция программы

Г) База данных для хранения настроек

7. Выберете правильный ответ из предложенных вариантов и обоснуйте свой выбор. При отладке программы точка останова используется для:

А) Автоматического исправления синтаксических ошибок

Б) Остановки выполнения программы в заданном месте для анализа состояния

В) Оптимизации кода по скорости

Г) Преобразования программы в машинный код

8. Прочитайте текст и запишите развернутый ответ. Разработчик написал фрагмент кода на C#, который должен вычислять среднее арифметическое элементов массива целых чисел. При тестировании на массиве [5, 10, 15] программа выводит 0, а ожидается 10. В коде используется целочисленное деление суммы на количество элементов без явного преобразования типов. Объясните причину ошибки и предложите способ её исправления.

9. Прочитайте текст и запишите развернутый ответ. В объектно-ориентированной программе есть базовый класс Транспорт с методом Двигаться(). От него наследуются классы Автомобиль и Велосипед. В методе Двигаться() базового класса выводится строка «Транспорт движется». Разработчик хочет, чтобы при вызове метода Двигаться() у объекта Автомобиль выводилось «Автомобиль едет по дороге», а у объекта Велосипед – «Велосипед едет по велодорожке». При этом нужно сохранить возможность вызывать реализацию базового класса. Опишите, какой механизм ООП следует применить и какие ключевые слова нужно использовать в C# при объявлении метода в базовом классе.

10. Прочитайте текст и запишите развернутый ответ. В оконном приложении (Windows Forms) есть кнопка buttonCalculate и текстовое поле textBoxResult. При нажатии на кнопку программа должна вычислять площадь прямоугольника по двум числам, введённым пользователем в поля textBoxWidth и textBoxHeight. Однако при запуске, если пользователь вводит буквы или оставляет поле пустым, программа аварийно завершается с исключением FormatException. Предложите способ обработки ошибочного ввода, чтобы программа не падала, а выводила в textBoxResult сообщение «Ошибка: введите числа». Укажите, какую конструкцию C# следует использовать и какие методы преобразования строк в числа являются безопасными в данном контексте.

11. Прочитайте текст и запишите развернутый ответ. В оконном приложении на WPF есть список ListBox с именами студентов и кнопка «Удалить выбранного». При нажатии на кнопку программа должна удалять выбранный элемент из списка. Однако разработчик заметил, что если элемент не выбран (пользователь просто нажал кнопку), программа выбрасывает исключение. Какое исключение возникает и как его правильно обработать?

12. Прочитайте текст и запишите развернутый ответ. Константы в языках программирования позволяют определять неизменяемые значения, которые используются в коде для повышения читаемости и удобства поддержки. Они помогают избежать «магических чисел» и дублирования данных. Как объявить целочисленную константу в C#?

13. Рассчитайте и запишите ответ. Представлен фрагмент кода на языке C#:

```
for (int i=1; i<=5; i++)
        {
            if (i==4)
                break;
            Console.Write(i);
        }
```

Что выведет программа в результате своей работы? В ответе укажите число.

14. Рассчитайте и запишите ответ. Представлен фрагмент программы на языке C#:

```
int s = 0;
        for (int i=1; i<=5; i++)
        {
            if (i==3)
                continue;
            s = s + i;
```

```
            }
Console.Write(s);
```

Какое значение s выведет программа в результате своей работы? В ответе укажите число.

15. Рассчитайте и запишите ответ. В программе используется одномерный целочисленный массив A (размерность – 25 элементов). Ниже представлен фрагмент программы на языке C#:

```
int[] a=new int[25];
a[0]= 2;
for(int i=1; i<=24; i++)
{
    a[i] = 2*a[i-1]%10;
     Console.WriteLine("a[{0}]={1}",i,a[i]);
 }
```

Чему будет равно значение A[25] после выполнения фрагмента программы? В ответе укажите число.

ПК 2.1. Осуществлять установку и настройку отдельных программных, программно-аппаратных средств защиты информации

1. Выберете правильный ответ из предложенных вариантов и обоснуйте свой выбор. Что из перечисленного не относится к базовым конструкциям структурного программирования?

А) Следование (последовательное выполнение)

Б) Рекурсия

В) Ветвление (условный оператор)

Г) Цикл

2. Выберете правильный ответ из предложенных вариантов и обоснуйте свой выбор. В языке C# какой модификатор доступа обеспечивает видимость члена класса только внутри текущего класса?

А) public

Б) protected

В) internal

Г) private

3. Выберете правильный ответ из предложенных вариантов и обоснуйте свой выбор. Что происходит при вызове деструктора (финализатора) в классе на C#?

А) Объект немедленно удаляется из памяти

Б) Выполняется освобождение неуправляемых ресурсов перед сборкой мусора

В) Вызывается конструктор копирования

Г) Создаётся новый экземпляр класса

4. Выберете правильный ответ из предложенных вариантов и обоснуйте свой выбор. Какой режим открытия файла в C# позволяет добавлять данные в конец существующего файла без удаления его содержимого?

А) FileMode.Create

Б) FileMode.Open

В) FileMode.Append

Г) FileMode.Truncate

5. Выберете правильный ответ из предложенных вариантов и обоснуйте свой выбор. Какой принцип объектно-ориентированного программирования позволяет одному и тому же имени метода выполнять разные действия в зависимости от типа объекта?

А) Инкапсуляция

Б) Наследование

В) Полиморфизм

Г) Агрегация

6. Выберете правильный ответ из предложенных вариантов и обоснуйте свой выбор. В оконном приложении (Windows Forms / WPF) событие Load возникает, когда:

А) Пользователь нажимает кнопку мыши

Б) Форма или окно только что созданы и загружаются в первый раз

В) Программа завершает работу

Г) Изменяется текст в текстовом поле

7. Выберете правильный ответ из предложенных вариантов и обоснуйте свой выбор. Какая из

перечисленных сред выполнения управляет памятью, безопасностью и исключениями для программ на C#?

А) JVM (Java Virtual Machine)
Б) CLR (Common Language Runtime)
В) MSIL компилятор
Г) Операционная система Windows

Индикатор ВЛАДЕТЬ:

ПК 2.1. Осуществлять установку и настройку отдельных программных, программно-аппаратных средств защиты информации

1. Рассчитайте и запишите ответ. Представлен фрагмент программы на языке C#:

```
int i = 1;
        while (i<=5)
        {
            if (i == 3)
                break;
            Console.Write(i);
            i++;
        }
```

Какое значение выведет программа в результате своей работы? В ответе укажите число.

2. Прочитайте текст и запишите развернутый обоснованный ответ. В C# функция может возвращать значение любого типа. Какое значение вернет функция IsEven при вызове IsEven(7)?

```
csharp
bool IsEven(int number)
{
    return number % 2 == 0;
}
```

3. Рассчитайте и запишите ответ. Дан фрагмент программы на языке C#:

```
int k=1, s=0;
while (s < 66)
 {
     k=k+3;
     s=s+k;
  }
```

Определите, чему будет равно значение переменной k. В ответе укажите число.

4. Прочитайте текст и запишите развернутый обоснованный ответ. В C# существуют процедуры, которые не возвращают значение. Что выведет процедура Transform при вызове Transform(4)?

```
csharp
void Transform(int x)
{
    x = x * 2;
    Console.Write(x);
    x += 3;
    Console.Write(x);
}
```

5. Рассчитайте и запишите ответ. Представлен фрагмент программы на языке C#:

```
int s = 0;
        for (int i=5; i>1; i--)
        {
            if (i==3)
                break;
            s = s + i;
        }
Console.Write(s);
```

Какое значение s выведет программа в результате своей работы? В ответе укажите число.

6. Прочитайте текст и запишите развернутый обоснованный ответ. В C# существуют статические и нестатические члены класса. Объясните разницу между статическими и нестатическими членами класса в C#.

7. Рассчитайте и запишите ответ. Представлен фрагмент программы на языке C#:

```
int i = 2;
        while (i<=5)
        {
            Console.Write(i);
            i+=2;
        }
```

Какое значение выведет программа в результате своей работы? В ответе укажите число.

8. Рассчитайте и запишите ответ. Дан текст программы на языке C#:

```
class M1
    {
        public int f(int a, int b)
        {
            b = 10 * a;
            int c = a+b;
            return c;
        }
    }
    class Program
    {
        static void Main(string[] args)
        {
            int a = 10;
            int b = 20;
            …
        }
    }
```

Чему равно значение переменной b после вызова метода f (с параметрами a=10 и b=20) класса M1. В ответе укажите число.

Вопросы к экзамену

Индикатор ЗНАТЬ:

ОК 02. Использовать современные средства поиска, анализа и интерпретации информации и информационные технологии для выполнения задач профессиональной деятельности

1. Установите правильную последовательность действий при создании консольного приложения на C# в среде разработки

А) Написание кода в файле Program.cs

Б) Запуск приложения (без отладки или с отладкой)

В) Создание нового проекта типа Console Application

Г) Компиляция проекта (построение решения)

2. Установите правильную последовательность этапов жизненного цикла программы при интерпретации (в отличие от компиляции)

А) Выполнение программы интерпретатором строка за строкой

Б) Получение исходного кода программы

В) Остановка интерпретатора при возникновении ошибки

Г) Анализ текущей строки исходного кода интерпретатором

3. Установите правильную последовательность операций при передаче параметров в функцию по значению в C#

А) Выполнение тела функции с использованием локальной копии параметра

Б) Вычисление значения аргумента в вызывающей программе

В) Копирование вычисленного значения в параметр функции

Г) Возврат из функции, локальная копия уничтожается

4. Установите правильную последовательность этапов при разработке приложения с использованием рекурсивной функции

А) Проверка условия окончания рекурсии (базовый случай)

Б) Глубокое вложение рекурсивных вызовов

В) Рекурсивный вызов функции с изменёнными аргументами

Г) Возврат из рекурсивных вызовов и вычисление результата

5. Установите правильную последовательность действий при работе с текстовым файлом в потоковом режиме (построчное чтение)

А) Открытие файла для чтения (создание StreamReader)

Б) Закрытие файла и освобождение ресурсов

В) Чтение очередной строки из файла

Г) Проверка достижения конца файла

6. Установите правильную последовательность этапов при создании перегрузки методов в одном классе

А) Вызов метода с определённым набором параметров

Б) Объявление нескольких методов с одинаковым именем, но разными параметрами

В) Компилятор выбирает подходящую перегрузку по количеству и типам аргументов

Г) Сравнение типов переданных аргументов с типами параметров каждой перегрузки

7. Установите правильную последовательность наследования и доступа к членам класса при использовании модификатора protected

А) Отсутствие доступа к protected-полю из несвязанного класса (не наследника)

Б) Наследование производного класса от базового

В) Доступ к protected-полю из метода производного класса

Г) Объявление protected-поля в базовом классе

8. Установите соответствие между видом алгоритмической структуры и её графическим представлением в блок-схеме (описанием формы)

Вид структуры	Описание формы
А) Начало или конец алгоритма	1. Прямоугольник
Б) Условный блок	2. Ромб
В) Ввод или вывод данных	3. Овал
Г) Вычислительный блок	4. Параллелограмм

9. Установите соответствие между понятием из области памяти программы и его описанием

Понятие	Описание
А) Стек	1. Область для локальных переменных и адресов возврата функций
Б) Куча	2. Область для объектов, создаваемых динамически
В) Статическая память	3. Сверхбыстрая память внутри процессора для хранения промежуточных данных
Г) Регистры процессора	4. Область для глобальных и статических переменных, существующих всё время работы программы

10. Установите соответствие между оператором или конструкцией в C# и её основным действием

Оператор (конструкция)	Действие
А) Оператор is	1. Тернарная операция (условное выражение)
Б) Оператор as	2. Проверка совместимости типа с возвращением true/false
В) Условный оператор ?:	3. Контроль переполнения при арифметических операциях
Г) Оператор checked	4. Безопасное приведение типа с возвратом null при ошибке
	5. Создание нового объекта без вызова конструктора

11. Установите соответствие между термином объектно-ориентированного анализа и проектирования и его определением

Термин ООП	Определение
А) Агрегация	1. Отношение «часть-целое», где части могут существовать отдельно от целого
Б) Композиция	2. Отношение «часть-целое», где части не могут существовать без целого
В) Ассоциация	3. Общая связь между классами (один класс использует

	другой)
Г) Зависимость	4. Временная связь (один класс зависит от изменений в другом)
	5. Копирование объекта с созданием нового экземпляра

12. Установите соответствие между типом файла в программировании и способом доступа к данным

Тип файла	Способ доступа
А) Текстовый файл	1. Данные хранятся в виде последовательности байтов без внутренней структуры
Б) Бинарный (двоичный) файл	2. Данные можно прочитать только с начала до конца, нельзя перепрыгнуть
В) Файл с произвольным доступом	3. Данные хранятся как последовательность символов, разделённых строками
Г) Последовательный файл (логическая лента)	4. Можно перемещаться к любой позиции в файле

13. Установите соответствие между видом полиморфизма и его проявлением в языках программирования

Вид полиморфизма	Проявление
А) Полиморфизм подтипов (включения)	1. Один и тот же класс может работать с разными типами данных (List<T>)
Б) Параметрический полиморфизм (обобщения / generics)	2. Один метод может иметь несколько реализаций с разными параметрами (одинаковое имя)
В) Ad-hoc полиморфизм (перегрузка)	3. Объект производного класса может использоваться как объект базового класса
Г) Полиморфизм приведения	4. Автоматическое неявное преобразование типов (int в double)

14. Установите соответствие между методом отладки или тестирования и его описанием

Метод отладки (тестирования)	Описание
А) Точка останова	1. Повторный запуск ранее успешных тестов после внесения изменений в код
Б) Трассировка	2. Принудительная остановка выполнения программы для просмотра состояния
В) Сниффинг переменных	3. Вывод значений переменных в консоль или лог-файл в процессе выполнения
Г) Регрессионное тестирование	4. Просмотр текущих значений переменных в окне отладчика

15. Установите соответствие между жизненным циклом объекта в управляемой среде (.NET) и событием, которое происходит

Жизненный цикл объекта	Событие
А) Вызов конструктора	1. Объект становится недостижимым и помечается на удаление
Б) Назначение последней ссылки на объект	2. Инициализация полей объекта и выделение начальной памяти
В) Вызов финализатора (деструктора)	3. Окончательное освобождение физической памяти, занятой объектом
Г) Освобождение памяти сборщиком мусора	4. Выполнение очистки неуправляемых ресурсов (при наличии финализатора)
	5. Копирование объекта в другой участок памяти

ПК 2.1. Осуществлять установку и настройку отдельных программных, программно-аппаратных средств защиты информации

1. Установите правильную последовательность событий при обработке события в оконном приложении Windows Forms

А) Пользователь нажимает кнопку мыши на элементе управления

Б) Операционная система формирует сообщение о событии

В) Выполняется код обработчика события, привязанного к этому элементу

Г) Среда выполнения направляет сообщение соответствующему элементу управления

2. Установите правильную последовательность этапов при использовании статических членов класса (статических полей и методов)

А) Обращение к статическому методу через имя класса

Б) Выделение памяти под статическое поле при первом обращении к классу

В) Инициализация статического поля (статическим конструктором)

Г) Выполнение статического метода без создания экземпляра класса

3. Установите правильную последовательность шагов при создании многофайлового проекта (модульного программирования) на C#

А) Размещение различных классов в отдельных файлах .cs

Б) Компиляция всех файлов проекта в единую сборку (exe или dll)

В) Указание пространств имён (namespace) для организации модулей

Г) Использование директив using для подключения пространств имён в других файлах

4. Установите правильную последовательность при приведении типов в иерархии классов (upcasting и downcasting)

А) Проверка совместимости типов с помощью оператора is или as

Б) Неявное приведение к типу базового класса (upcasting)

В) Попытка явного приведения обратно к типу производного класса (downcasting)

Г) Создание объекта производного класса

5. Установите правильную последовательность логических этапов при реализации интерфейса в классе на C#

А) Объявление интерфейса с набором методов и свойств

Б) Вызов методов интерфейса через ссылку на интерфейс

В) Создание объекта класса и приведение к типу интерфейса

Г) Реализация всех членов интерфейса в классе

6. Установите правильную последовательность действий при работе с динамической памятью (управляемой) в среде .NET

А) Создание объекта с помощью оператора new

Б) Сборщик мусора (Garbage Collector) определяет объекты без ссылок

В) Использование объекта в программе

Г) Освобождение памяти, занимаемой объектом, сборщиком мусора в фоновом режиме

7. Установите правильную последовательность этапов при создании собственного исключения в C#

А) Создание класса исключения, унаследованного от Exception

Б) Генерация исключения с помощью throw в коде программы

В) Перехват исключения в блоке catch с указанием типа собственного исключения

Г) Добавление пользовательских полей и конструкторов в класс исключения

8. Установите соответствие между видом оператора цикла в C# и местом проверки условия продолжения

Вид оператора	Место проверки условия продолжения
А) Цикл for	1. Проверка условия в начале каждой итерации (кроме первой, если не считать инициализацию)
Б) Цикл foreach	2. Проверка условия выполняется неявно через перечисление коллекции
В) Цикл while	3. Проверка условия в конце тела цикла
Г) Цикл do-while	4. Проверка условия выполняется только один раз перед входом в цикл

9. Установите соответствие между компонентом архитектуры .NET и его функциональным назначением

Компонент	Назначение
А) CLR	1. Набор стандартных типов, единых для всех языков .NET
Б) CTS	2. Среда выполнения, управляющая памятью, безопасностью и исключениями
В) CLS	3. Набор стандартных классов (работа с файлами, строками, коллекциями)

Г) BCL	4. Правила, которым должен следовать язык, чтобы взаимодействовать с другими .NET-языками

10. Установите соответствие между способом создания многомерного массива в C# и его характеристикой

Способ создания массива	Характеристика
А) Прямоугольный массив	1. Массив, каждый элемент которого сам является массивом (возможна разная длина вложенных массивов)
Б) Ступенчатый массив	2. Массив, выделяемый в стеке, а не в куче (для unsafe-кода)
В) Одномерный массив объектов	3. Массив с одинаковым количеством элементов в каждом измерении (таблица)
Г) Массив фиксированной длины	4. Массив может хранить элементы любых типов (но требуется упаковка значимых типов)

11. Установите соответствие между видом функции (метода) в C# и её особенностью вызова или объявления

Вид функции (метода)	Особенность
А) Статический метод	1. Метод без реализации в классе, реализуется в другом файле (второй части класса)
Б) Виртуальный метод	2. Метод, который можно переопределить в производном классе
В) Абстрактный метод	3. Метод, принадлежащий самому типу, а не его экземпляру
Г) Частичный метод	4. Метод, который обязательно должен быть переопределён в производном классе
	5. Метод, который нельзя переопределять (по умолчанию)

12. Установите соответствие между термином из области событийно-ориентированного программирования и его описанием

Термин	Описание
А) Событие	1. Метод, который вызывается при наступлении события
Б) Подписчик	2. Тип, описывающий сигнатуру метода, который может обрабатывать событие
В) Обработчик события	3. Объект или класс, который ожидает наступления события
Г) Делегат	4. Механизм, позволяющий объекту уведомлять других об изменении своего состояния

13. Установите соответствие между принципом написания «чистого кода» и его кратким содержанием

Принцип написания	Содержание
А) DRY	1. Не следует добавлять функциональность, пока она действительно не понадобится
Б) KISS	2. Один из принципов: класс должен иметь только одну причину для изменения
В) YAGNI	3. Избегайте дублирования кода; выносите повторяющееся в отдельные методы или классы
Г) SOLID	4. Решение должно быть максимально простым и понятным
	5. Используйте как можно больше глобальных переменных для упрощения доступа

14. Установите соответствие между видом отладочного вывода и его типичным применением

Вид вывода	Применение
А) Вывод в консоль	1. Вывод, который остаётся в релизной версии для администратора системы
Б) Запись в лог-файл	2. Вывод, который работает только в отладочной (Debug) конфигурации
В) Отладочное окно	3. Вывод, требующий обязательного подтверждения пользователя и блокирующий интерфейс
Г) Всплывающее сообщение	4. Быстрый вывод при консольном тестировании алгоритма

15. Установите соответствие между видом комментария в программировании и его

назначением

Вид комментария	Назначение
А) Однострочный комментарий	1. Используется для автоматической генерации документации
Б) Многострочный комментарий	2. Закрывает от компиляции блок кода любой длины
В) XML-комментарий	3. Указывает на незавершённый фрагмент кода или место для доработки
Г) Комментарий-заглушка	4. Закрывает от компиляции только часть одной строки после двух косых черт

Индикатор УМЕТЬ:

ОК 02. Использовать современные средства поиска, анализа и интерпретации информации и информационные технологии для выполнения задач профессиональной деятельности

1. Выберите правильный ответ из предложенных вариантов и обоснуйте свой выбор. Что из перечисленного является правильным способом вызова функции в C#?

А) call MyFunction();

Б) MyFunction;

В) MyFunction();

Г) invoke MyFunction;

2. Выберите правильный ответ из предложенных вариантов и обоснуйте свой выбор. Какой из вариантов является некорректным объявлением переменной в C#?

А) int x = 5;

Б) double y = 3.14;

В) string s = "Hello";

Г) bool b = "true";

3. Выберите правильный ответ из предложенных вариантов и обоснуйте свой выбор. Какой оператор выполняет целочисленное деление на языке C#?

А) /

Б) %

В) //

Г) \

4. Выберите правильный ответ из предложенных вариантов и обоснуйте свой выбор. Какое максимальное количество классов может быть указано в списке наследования на языке C#?

А) 1

Б) 2

В) Неограниченно

Г) Зависит от версии .NET

5. Выберите правильный ответ из предложенных вариантов и обоснуйте свой выбор. Какое ключевое слово используется для определения функции в C#?

А) func

Б) method

В) void

Г) def

6. Выберите правильный ответ из предложенных вариантов и обоснуйте свой выбор. Какой символ используется для однострочного комментария в C#?

А) /*

Б) //

В) <!--

Г) #

7. Выберите правильный ответ из предложенных вариантов и обоснуйте свой выбор. Как объявить массив целых чисел в C#?

А) int array = new int[5];

Б) int[] array = new int[5];

В) array int = new int[5];

Г) int array[] = new int[5];

8. Прочитайте текст и запишите развернутый ответ. Вам необходимо разработать модуль для логирования сообщений в текстовый файл log.txt. Логи должны записываться в реальном времени, но файл не должен блокироваться другими процессами. При каждом запуске программы новые сообщения должны добавляться в конец файла, а старые – сохраняться. Какой режим открытия файла и какой класс из System.IO следует использовать в C# для дозаписи без потери существующих данных? Укажите конструкцию, которая автоматически закроет файл даже при возникновении исключения.

9. Прочитайте текст и запишите развернутый ответ. В оконном приложении (Windows Forms) есть главное окно и дочерняя форма, которая открывается по нажатию кнопки. Разработчик заметил, что если закрыть дочернюю форму, а затем попытаться открыть её снова, возникает ошибка, так как объект формы уже уничтожен. Почему нельзя повторно использовать один и тот же экземпляр формы после её закрытия?

10. Прочитайте текст и запишите развернутый ответ. Разработчик пишет метод на C#, который должен принимать массив целых чисел и возвращать новый массив, содержащий только положительные элементы из исходного. Он использует цикл for и внутри него условие if (numbers[i] > 0), после чего пытается добавить элемент в новый массив с помощью метода Add, но компилятор выдаёт ошибку. Почему возникает ошибка?

11. Прочитайте текст и запишите ответ. В массиве целых чисел из 10 элементов сумма всех элементов равна 100. После выполнения цикла, который увеличивает каждый чётный по индексу элемент на 10 (индексация с нуля), сумма элементов изменилась. Определите, на сколько увеличилась общая сумма массива. Разрешается использование калькулятора.

12. Прочитайте текст и запишите ответ. Программа считывает три числа: 15, 7 и 22. Алгоритм сначала находит их сумму, затем вычитает из суммы минимальное из этих трёх чисел, а затем делит результат на количество оставшихся чисел (после вычитания минимального). Чему равен итоговый результат? В ответе запишите только целую часть числа. Разрешается использование калькулятора.

13. Прочитайте текст и запишите ответ. В цикле выполняется 8 итераций. На первой итерации к переменной S прибавляется 1, на второй – 2, на третьей – 3, и так далее (каждая следующая итерация добавляет число, на единицу большее предыдущего). Начальное значение S равно нулю. Чему станет равно S после выполнения всех 8 итераций? Разрешается использование калькулятора.

14. Прочитайте текст и запишите ответ. В текстовом файле записаны целые числа: сначала 5 чисел, затем 8 чисел, затем 3 числа. Программа считывает все числа и находит их среднее арифметическое (сумма делённая на количество). Сумма всех чисел оказалась равна 112. Чему равно среднее арифметическое? Разрешается использование калькулятора.

15. Прочитайте текст и запишите ответ. В объектно-ориентированной программе создано 3 объекта класса Автомобиль и 2 объекта класса Мотоцикл. Каждый объект класса Автомобиль при создании увеличивает счётчик ВсегоТранспорта на 1, а каждый объект класса Мотоцикл — увеличивает тот же счётчик на 1. Изначально счётчик равен нулю. Затем 2 объекта Автомобиль удаляются (уничтожаются), но счётчик не уменьшается. Чему стал равен счётчик ВсегоТранспорта после создания всех объектов? Разрешается использование калькулятора.

ПК 2.1. Осуществлять установку и настройку отдельных программных, программно-аппаратных средств защиты информации

1. Выберите правильный ответ из предложенных вариантов и обоснуйте свой выбор. Какой из вариантов является корректным идентификатором на языке C#?

А) 123variable

Б) _myVar

В) float

Г) my-var

2. Выберите правильный ответ из предложенных вариантов и обоснуйте свой выбор. Какой тип данных в C# является ссылочным?

А) int

Б) double

В) string

Г) bool

3. Выберите правильный ответ из предложенных вариантов и обоснуйте свой выбор. Какой

метод класса Random в C# генерирует случайное целое число?

А) NextDouble()

Б) Next()

В) NextBytes()

Г) NextFloat()

4. Выберите правильный ответ из предложенных вариантов и обоснуйте свой выбор. Какая из конструкций в C# используется для вывода данных в консоль?

А) Console.Read()

Б) Console.ReadLine()

В) Console.Write()

Г) Console.Input()

5. Выберите правильный ответ из предложенных вариантов и обоснуйте свой выбор. Какой метод в C# считывает всю строку из консоли?

А) Console.Read()

Б) Console.ReadKey()

В) Console.ReadLine()

Г) Console.ReadString()

6. Выберите правильный ответ из предложенных вариантов и обоснуйте свой выбор. Что из перечисленного является примером полиморфизма?

А) Создание нескольких конструкторов класса

Б) Переопределение метода в производном классе

В) Объявление приватных полей

Г) Использование статических методов

7. Выберите правильный ответ из предложенных вариантов и обоснуйте свой выбор. Какое ключевое слово используется для вызова конструктора базового класса?

А) this

Б) base

В) super

Г) parent

Индикатор ВЛАДЕТЬ:

ПК 2.1. Осуществлять установку и настройку отдельных программных, программно-аппаратных средств защиты информации

1. Рассчитайте и запишите ответ. Значения элементов двухмерного массива A, задаются с помощью следующего фрагмента программы на языке C#:

```
int i, k;
for (i = 0; i < 5; i++)
{
    for (k = 0; k < 5; k++)
    {
        if (i > k)
            a[i, k] = 1;
        else
            a[i, k] = 0;
    }
 }
```

Чему равна сумма элементов двумерного массива после выполнения этого фрагмента программы? В ответе укажите число.

2. Прочитайте текст и запишите развернутый обоснованный ответ. В C# при создании методов (процедур) можно использовать операторы циклов. Что выведет процедура LoopDemo при вызове LoopDemo(3)?

csharp

```
void LoopDemo(int n)
{
    for (int i = 1; i <= n; i++)
```

```
    {
        Console.Write(i * 2 + " ");
    }
}
```

3. Рассчитайте и запишите ответ. Представлен фрагмент программы на языке C#:

```
for (int i=1; i<=5; i++)
            {
                Console.Write(i);
    if (i==3)
                    break;
            }
```

Что выведет программа в результате своей работы?

4. Рассчитайте и запишите ответ. Представлен фрагмент программы на языке C#:

```
for (int i=1; i<=6; i++)
            {
                if (i==3 || i==4)
                    continue;
                Console.Write(i);
            }
```

Что выведет программа в результате своей работы? В ответе укажите число.

5. Рассчитайте и запишите ответ. На языке C# дан массив строкового типа string[] s = { "1", "2", "3", "4", "5"}. Компонент ListBox заполняется значениями массива s. Если к строковому массиву применить метод RemoveAt(3), то какой вид будет иметь компонент ListBox? В ответе укажите числа без пробелов в одной строке в том порядке, как в массиве после применения метода. В ответе укажите число.

6. Рассчитайте и запишите ответ. Представлен фрагмент программы на языке C#:

```
int i = 2;
            do
            {
                Console.Write(i);
                i += 2;
            }
            while (i <= 6);
```

Какое значение выведет программа в результате своей работы? В ответе укажите число.

7. Рассчитайте и запишите ответ. Значения двумерного массива задаются с помощью вложенного оператора цикла в представленном фрагменте программы на языке C#:

```
for(i=0; i<n; i++)
    for (j = 0; j < n; j++)
        a[i, j] =i+j+3;
```

Чему будет равно значение B[1,3]? В ответе укажите только число.

8. Прочитайте текст и запишите развернутый обоснованный ответ. В C# все типы делятся на значимые и ссылочные. Приведите примеры значимых типов.

Итоговое тестирование

ОК 02. Использовать современные средства поиска, анализа и интерпретации информации и информационные технологии для выполнения задач профессиональной деятельности

1. Что понимается под алгоритмом в контексте алгоритмизации?

А) Любая последовательность чисел

Б) Конечная последовательность команд, однозначно определяющая процесс преобразования исходных данных в результат

В) Описание архитектуры компьютера

Г) Текст программы на языке высокого уровня

2. Какие из перечисленных языков относятся к языкам высокого уровня?

А) Ассемблер

Б) Python

В) Машинный код

Г) C#

3. Какие парадигмы программирования существуют?

А) Только объектно-ориентированная

Б) Структурное, объектно-ориентированное, функциональное, логическое программирование

В) Только процедурное программирование

Г) Только функциональное программирование

4. Какие из утверждений о процессе отладки верны?

А) Отладка направлена на обнаружение и исправление синтаксических и логических ошибок

Б) Отладка и тестовый контроль — это одно и то же

В) Тестовый контроль может выполняться автоматически с помощью Unit-тестов

Г) Отладка включает пошаговое выполнение программы и просмотр значений переменных

5. Какие характеристики относятся к языку C#?

А) Разработан компанией Microsoft

Б) Используется только для создания веб-сайтов

В) Поддерживает объектно-ориентированное программирование

Г) Не поддерживает сборку мусора

6. Какие типы данных в C# относятся к простым (встроенным) типам?

А) Классы

Б) Целочисленные типы (int, long)

В) Массивы

Г) Логический тип (bool)

7. Какие конструкции относятся к базовым конструкциям структурного программирования?

А) Следование (последовательное выполнение)

Б) Рекурсия

В) Ветвление (условный оператор)

Г) Цикл

ПК 2.1. Осуществлять установку и настройку отдельных программных, программно-аппаратных средств защиты информации

8. Какие из перечисленных действий корректны для работы с массивами в C#?

А) Объявление массива с указанием его размера

Б) Изменение длины массива после его создания без создания нового массива

В) Обращение к элементу массива по индексу

Г) Хранение в массиве элементов разных типов (без использования object)

9. Что верно для процедур и функций в программировании?

А) Процедура обязательно возвращает значение

Б) Функция обязательно возвращает значение

В) Функция не может быть вызвана без использования возвращаемого значения

Г) Процедура может иметь параметры

10. Что представляет собой класс в объектно-ориентированном программировании?

А) Конкретный объект, созданный в памяти

Б) Шаблон для создания объектов, объединяющий данные и методы

В) Отдельную переменную

Г) Функцию для ввода данных

11. Какие принципы объектно-ориентированного программирования реализуются через наследование и полиморфизм?

А) Инкапсуляция (только)

Б) Возможность создания нового класса на основе существующего (наследование)

В) Возможность объектов с одинаковым интерфейсом иметь разную реализацию (полиморфизм)

Г) Запрет на использование циклов

12. Что верно для конструктора и деструктора в классах?

А) Конструктор вызывается автоматически при создании объекта

Б) Деструктор служит для инициализации объекта

В) Деструктор вызывается автоматически при уничтожении объекта (освобождении ресурсов)

Г) Конструктор не может иметь параметров

13. Что характерно для модульного программирования?

А) Программа разбивается на логически законченные части (модули)
Б) Весь код пишется в одном файле
В) Модули могут разрабатываться и компилироваться отдельно
Г) Модули не могут повторно использоваться в других проектах

14. Какие из перечисленных свойств характерны для алгоритма?
А) Дискретность (разбиение на отдельные шаги)
Б) Бесконечность выполнения
В) Результативность (получение результата за конечное число шагов)
Г) Неоднозначность (возможность разного толкования команд)

15. Что из перечисленного относится к разработке оконных (графических) приложений?
А) Только консольный ввод-вывод
Б) Использование событийно-ориентированного подхода
В) Наличие визуальных элементов управления (кнопки, поля ввода)
Г) Отсутствие взаимодействия с пользователем