МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ
ФГБОУ ВО «Уральский государственный экономический университет»

Одобрена
Педагогическим советом колледжа

протокол № 4 от 18.11.2025 г.

Директор колледжа _____ А.Э.Чечулин
(подпись)

Утверждена
Советом по учебно-методическим вопросам и качеству образования

протокол № 4 от 16.12.2025 г.

Председатель _____ Д.А. Карх
(подпись)

РАБОЧАЯ ПРОГРАММА ДИСЦИПЛИНЫ

Наименование дисциплины	ОП.06 Основы алгоритмизации и программирования
Специальность	09.02.11 РАЗРАБОТКА И УПРАВЛЕНИЕ ПРОГРАММНЫМ ОБЕСПЕЧЕНИЕМ
Форма обучения	очная
Год набора	2026

Разработана:
Преподаватель
Д.О. Березин

Екатеринбург
2026 г.

СОДЕРЖАНИЕ

ВВЕДЕНИЕ

Рабочая программа дисциплины является частью основной образовательной программысреднего профессионального образования - программы подготовки специалистов среднего звена,разработанной в соответствии с ФГОС СПО

ФГОС СПО	Федеральный государственный образовательный стандарт среднегопрофессионального образования по специальности 09.02.11 РАЗРАБОТКА ИУПРАВЛЕНИЕ ПРОГРАММНЫМ ОБЕСПЕЧЕНИЕМ (приказ МинпросвещенияРоссии от 24.02.2025 г. № 138)
ПС	

1. ЦЕЛЬ ОСВОЕНИЯ ДИСЦИПЛИНЫ

Целью дисциплины "Основы алгоритмизации и программирования" является изучение иосвоение базовых понятий и приемов программирования, применяемых на всех основных этапахразработки программ; изучение методов программирования для овладения знаниями в областитехнологии программирования; подготовка к осознанному использованию как языковпрограммирования, так и методов программирования.

В результате освоения дисциплины обучающийся должен:

Уметь:

- разрабатывать алгоритмы для конкретных задач;
- использовать программы для графического отображения алгоритмов;
- определять сложность работы алгоритмов;
- работать в среде программирования;
- реализовывать построенные алгоритмы в виде программ на конкретном языкепрограммирования;
- оформлять код программы в соответствии со стандартом кодирования;
- выполнять проверку, отладку кода программы

Знать:

- понятие алгоритмизации, свойства алгоритмов, общие принципы построенияалгоритмов, основные алгоритмические конструкции;
- эволюцию языков программирования, их классификацию, понятие системыпрограммирования;
- основные элементы языка, структуру программы, операторы и операции, управляющиеструктуры, структуры данных, файлы, классы памяти;
- подпрограммы, составление библиотек подпрограмм
- объектно-ориентированную модель программирования, основные принципы объектно-ориентированного программирования на примере алгоритмического языка: понятие классов иобъектов, их свойств и методов, инкапсуляция и полиморфизма, наследования и переопределения.

Результатом освоения дисциплины, в соответствии с рабочей программой воспитания,является формирование у обучающихся следующих личностных результатов обучения:

ПТВ 1. Понимающий профессиональные идеалы и ценности, уважающий труд, результатытруда, трудовые достижения российского народа, трудовые и профессиональные достижения своихземляков, их вклад в развитие своего поселения, края, страны.

ПТВ 2. Участвующий в социально значимой трудовой и профессиональной деятельностиразного вида в семье, образовательной организации, на базах производственной практики, в своейместности.

ПТВ 3. Выражающий осознанную готовность к непрерывному образованию исамообразованию в выбранной сфере профессиональной деятельности.

ЦНП 1. Деятельно выражающий познавательные интересы в разных предметных областях сучётом своих интересов, способностей, достижений, выбранного направления профессиональногообразования и подготовки.

ЦНП 2. Обладающий представлением о современной научной картине мира, достиженияхнауки и техники, аргументированно выражающий понимание значения науки и технологий дляразвития российского общества и обеспечения его безопасности.

ЦНП 6. Развивающий и применяющий навыки наблюдения, накопления и систематизации

фактов, осмысления опыта в естественнонаучной и гуманитарной областях познания,исследовательской и профессиональной деятельности

ЦНП 7 В. Признающий ценность непрерывного образования, ориентирующийся визменяющемся рынке труда, избегающий безработицы; управляющий собственнымпрофессиональным развитием; рефлексивно оценивающий собственный жизненный опыт, критерииличной успешности

2. МЕСТО ДИСЦИПЛИНЫ В СТРУКТУРЕ ООП

Дисциплина относится к вариативной части учебного плана.

3. ОБЪЕМ ДИСЦИПЛИНЫ

Промежуточнаяаттестация	Часов				
	Всего засеместр	Контактнаяработа .(по уч.зан.)		Самостоятельная работа в том числеподготовкаконтрольных икурсовых	
		Всего	Лабораторные		
Семестр 3					
Зачет	50	48	48	2	0
Семестр 4					
Экзамен	102	94	92	2	0
	152	142	140	4	0

4.ПЛАНИРУЕМЫЕ РЕЗУЛЬТАТЫ ОСВОЕНИЯ ООП

В результате освоения ООП у выпускника должны быть сформированы компетенции,установленные в соответствии ФГОС СПО.

Общие компетенции (ОК)

Шифр и наименованиекомпетенции	Индикаторы достижения компетенций

ОК 01. Выбирать способырешения задачпрофессиональнойдеятельностиприменительно к различнымконтекстам	Уметь: - распознавать задачу и/или проблему в профессиональном и/илисоциальном контексте; - анализировать задачу и/или проблему и выделять ее составные части; - определять этапы решения задачи; - выявлять и эффективно искать информацию, необходимую длярешения задачи и/или проблемы; - составить план действий; - определять необходимые ресурсы; - владеть актуальными методами работы в профессиональной исмежных сферах; - реализовать составленный план; - оценивать результат и последствия своих действий (самостоятельноили с помощью наставника) Знать: - актуальный профессиональный и социальный контекст, в которомприходится работать и жить; - основные источники информации и ресурсы для решения задач ипроблем в профессиональном и/или социальном контексте; - алгоритмы выполнения работ в профессиональных и смежныхобластях; - методы работы в профессиональных и смежных сферах;
ОК 02. Использоватьсовременные средствапоиска, анализа иинтерпретацииинформации, иинформационныетехнологии для выполнениязадач профессиональнойдеятельности	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 Знать: - номенклатура информационных источников, применяемых впрофессиональной деятельности;
ОК 09. Пользоватьсяпрофессиональнойдокументацией нагосударственном иностранном языках	Уметь: - понимать общий смысл четко произнесенных высказываний наизвестные темы (профессиональные и бытовые), понимать тексты набазовые профессиональные темы. Знать: - правила построения простых и сложных предложений напрофессиональные темы

Профессиональные компетенции (ПК)

Шифр и наименованиекомпетенции	Индикаторы достижения компетенций
разработка и интеграция модулей программного обеспечения	
ПК 2.2. Разрабатыватьмодули программногообеспечения	Уметь: - разрабатывать алгоритмы для конкретных задач; - использовать программы для графического отображенияалгоритмов; - определять сложность работы алгоритмов; - работать в среде программирования; - реализовывать построенные алгоритмы в виде программ наконкретном языке программирования; - оформлять код программы в соответствии со стандартомкодирования; Знать: - понятие алгоритмизации, свойства алгоритмов, общиепринципы построения алгоритмов, основные алгоритмическиеконструкции; - эволюцию языков программирования, их классификацию,понятие системы программирования; - основные элементы языка, структуру программы, операторы иоперации, управляющие структуры, структуры данных, файлы, классыпамяти; - подпрограммы, составление библиотек подпрограмм - объектно-ориентированную модель программирования,основные принципы объектно-ориентированного программированияна примере алгоритмического языка: понятие классов и объектов, ихсвойств и методов, инкапсуляция и полиморфизма, наследования ипереопределения.
ПК 2.4. Выполнятьтестирование и отладкупрограммного обеспечения	Уметь: - работать в среде программирования; - реализовывать построенные алгоритмы в виде программ наконкретном языке программирования; - выполнять проверку, отладку кода программы Знать: - понятие алгоритмизации, свойства алгоритмов, общиепринципы построения алгоритмов, основные алгоритмическиеконструкции; - эволюцию языков программирования, их классификацию,понятие системы программирования; - основные элементы языка, структуру программы, операторы иоперации, управляющие структуры, структуры данных, файлы, классыпамяти; - подпрограммы, составление библиотек подпрограмм - объектно-ориентированную модель программирования,основные принципы объектно-ориентированного программированияна примере алгоритмического языка: понятие классов и объектов, ихсвойств и методов, инкапсуляция и полиморфизма, наследования ипереопределения.

5. ТЕМАТИЧЕСКИЙ ПЛАН

Тема	Часов						
	Наименование темы	Всего часов	Контактная работа .(по уч.зан.)			Самост.работа	Контрольсамостоятельной работы
			Лекции	Лабораторные	Практическиезанятия		
Семестр 3		48					
Тема 1.	Языки программирования (ОК 01, ОК 2,ОК 9, ПК 2.2, ПК 2.4, ПТВ 1, ПТВ 2,ПТВ	2		2			
Тема 2.	Типы данных (ОК 01, ОК 2, ОК 9, ПК2.2, ПК 2.4, ПТВ 1, ПТВ 2, ПТВ 3, ЦНП1,	2		2			
Тема 3.	Операторы языка программирования (ОК01, ОК 2, ОК 9, ПК 2.2, ПК 2.4, ПТВ 1,ПТВ 2, ПТВ 3, ЦНП	20		20			
Тема 4.	Процедуры и функции (ОК 01, ОК 2, ОК9, ПК 2.2, ПК 2.4, ПТВ 1, ПТВ 2, ПТВ	16		16			
Тема 5.	Структуризация в программировании(ОК 01, ОК 2, ОК 9, ПК 2.2, ПК 2.4, ПТВ1, ПТВ 2, ПТВ 3, ЦНП	8		6		2	
Семестр 4		98					
Тема 6.	Модульное программирование (ОК 01,ОК 2, ОК 9, ПК 2.2, ПК 2.4, ПТВ 1, ПТВ2, ПТВ 3,	18		18			
Тема 7.	Указатели (ОК 01, ОК 2, ОК 9, ПК 2.2,ПК 2.4, ПТВ 1, ПТВ 2, ПТВ 3, ЦНП	20		20			
Тема 8.	Основные принципы объектно-ориентированного программирования(ООП) (ОК 01, ОК 2, ОК 9, ПК 2.2,	2		2			
Тема 9.	Интегрированная среда разработчика(ОК 01, ОК 2, ОК 9, ПК 2.2, ПК 2.4, ПТВ1, ПТВ 2, ПТВ 3, ЦНП	26		24		2	
Тема 10.	Визуальное событийно-управляемоепрограммирование (ОК 01, ОК 2, ОК 9,ПК 2.2, ПК 2.4, ПТВ 1,	32		30			

6. ФОРМЫ ТЕКУЩЕГО КОНТРОЛЯ И ПРОМЕЖУТОЧНОЙ АТТЕСТАЦИИ ШКАЛЫ ОЦЕНИВАНИЯ

Раздел/Тема	Вид оценочного	Описание оценочного средства	Критериоценивания
Текущий контроль (Приложение 4)			

Тема 1	Реферат	Защита реферата по теме. Количество тем - 25.	Оценивается от 2до 5 баллов
Тема 2	Вопросы	Письменный опрос по вопросам. Количествовопросов 10. Количество вариантов - 1.	Оценивается от 2до 5 баллов
Тема 3-4	Практическаяработа	Работа состоит из 2 вариантов по 3 задания вкаждом варианте.	Оценивается от 2до 5 баллов
Тема 5	Вопросы	Письменный опрос по вопросам. Количествовопросов 15. Количество вариантов - 1.	Оценивается от 2до 5 баллов
Тема 6	Практическаяработа	Работа состоит из 2 вариантов по 3 задания вкаждом варианте.	Оценивается от 2до 5 баллов
Тема 7-8	Тест	Тест состоит из 15 вопросов. Закрытого типа.Количество вариантов - 2.	Оценивается от 2до 5 баллов
Тема 9-10	Практическаяработа	Работа состоит из 2 вариантов по 3 задания вкаждом варианте.	Оценивается от 2до 5 баллов
Промежуточная аттестация (Приложение 5)			
3 семестр (За)	Билет к зачету	Билет состоит из 2 теоретических вопросов и1 практического задания. Количество билетов- 25.	Зачет/незачет
4 семестр(Эк)	Экзаменационныйбилет	Билет состоит из 2 теоретических вопросов, 3тестовых заданий и 1 практическогозадания. Количество билетов - 25.	Оценивается от 2до 5 баллов

ОПИСАНИЕ ШКАЛ ОЦЕНИВАНИЯ

Показатель оценки освоения ООП формируется на основе объединения текущего контроля ипромежуточной аттестации обучающегося.

Показатель рейтинга по каждой дисциплине выражается в процентах, который показываетуровень подготовки студента.

Текущий контроль. Используется 5-балльная система оценивания. Оценка работы студента втечении семестра осуществляется преподавателем в соответствии с разработанной им системойоценки учебных достижений в процессе обучения по данной дисциплине.

В рабочих программах дисциплин (предметов) и практик закреплены виды текущегоконтроля, планируемые результаты контрольных мероприятий и критерии оценки учебныйдостижений.

В течение семестра преподавателем проводится не менее 3-х контрольных мероприятий, пооценке деятельности студента.

Промежуточная аттестация. Используется 5-балльная система оценивания. Оценка работыстудента по окончанию дисциплины (части дисциплины) осуществляется преподавателем всоответствии с разработанной им системой оценки достижений студента в процессе обучения поданной дисциплине. Промежуточная аттестация также проводится по окончанию формированиякомпетенций.

Показатель оценки	По 5-балльной системе	Характеристика показателя
100% - 85%	отлично	обладают теоретическими знаниями в полном объеме,понимают, самостоятельно умеют применять,исследовать, идентифицировать, анализировать,систематизировать, распределять по категориям,рассчитать показатели, классифицировать,разрабатывать модели, алгоритмизировать, управлять,организовать, планировать процессы исследования,осуществлять оценку результатов на высоком уровне
84% - 70%	хорошо	обладают теоретическими знаниями в полном объеме,понимают, самостоятельно умеют применять,исследовать, идентифицировать, анализировать,систематизировать, распределять по категориям,рассчитать показатели, классифицировать,разрабатывать модели, алгоритмизировать, управлять,организовать, планировать процессы исследования,осуществлять оценку результатов. Могут быть допущены недочеты, исправленныестудентом самостоятельно в процессе работы (ответаи т.д.)
69% - 50%	удовлетворительно	обладают общими теоретическими знаниями, умеютприменять, исследовать, идентифицировать,анализировать, систематизировать, распределять покатегориям, рассчитать показатели,классифицировать, разрабатывать модели,алгоритмизировать, управлять, организовать,планировать процессы исследования, осуществлятьоценку результатов на среднем уровне. Допускаются ошибки, которые студент затрудняетсяисправить самостоятельно.
49 % и менее	неудовлетворительно	обладают не полным объемом общих теоретическимизнаниями, не умеют самостоятельно применять,исследовать, идентифицировать, анализировать,систематизировать, распределять по категориям,рассчитать показатели, классифицировать,разрабатывать модели, алгоритмизировать, управлять,организовать, планировать процессы исследования,осуществлять оценку результатов. Не сформированы умения и навыки для
100% - 50%	зачтено	характеристика показателя соответствует «отлично»,«хорошо», «удовлетворительно»
49 % и менее	не зачтено	характеристика показателя соответствует«неудовлетворительно»

7. СОДЕРЖАНИЕ ДИСЦИПЛИНЫ

7.2 Содержание практических занятий и лабораторных работ

Тема 1. Языки программирования (ОК 01, ОК 2, ОК 9, ПК 2.2, ПК 2.4, ПТВ 1, ПТВ 2, ПТВ 3, ЦНП 1,ЦНП 2, ЦНП 6, ЦНП 7В)

Лабораторная работа №1 "Стандарты языков программирования. Среда проектирования.Компиляторы и интерпретаторы"
Выполнение практических заданий по теме

Тема 2. Типы данных (ОК 01, ОК 2, ОК 9, ПК 2.2, ПК 2.4, ПТВ 1, ПТВ 2, ПТВ 3, ЦНП 1, ЦНП 2,ЦНП 6, ЦНП 7В)

Лабораторная работа №2 "Составление словесных алгоритмов"
Выполнение практических заданий по теме

Тема 3. Операторы языка программирования (ОК 01, ОК 2, ОК 9, ПК 2.2, ПК 2.4, ПТВ 1, ПТВ 2,ПТВ 3, ЦНП 1, ЦНП 2, ЦНП 6, ЦНП 7В)

Лабораторная работа №3 "Интерфейс Visual Studio"
Выполнение практических заданий по теме
Лабораторная работа №4 "Типы данных"
Выполнение практических заданий по теме
Лабораторная работа №5 "Организация ввода-вывода данных"
Выполнение практических заданий по теме
Лабораторная работа №6 "Использование стандартных функций"
Выполнение практических заданий по теме
Лабораторная работа №7 "Использование стандартных функций для работы со строками"
Выполнение практических заданий по теме
Лабораторная работа №8 "Операторы языка C#"
Выполнение практических заданий по теме
Лабораторная работа №9 "Составление программ линейной структуры"
Выполнение практических заданий по теме
Лабораторная работа №10 "Составление программ разветвляющейся структуры"
Выполнение практических заданий по теме
Лабораторная работа №11 "Составление программ циклической структуры"
Выполнение практических заданий по теме
Лабораторная работа №12 "Обработка одномерных массивов"
Выполнение практических заданий по теме

Тема 4. Процедуры и функции (ОК 01, ОК 2, ОК 9, ПК 2.2, ПК 2.4, ПТВ 1, ПТВ 2, ПТВ 3, ЦНП 1,ЦНП 2, ЦНП 6, ЦНП 7В)

Лабораторная работа №13 "Обработка двухмерных массивов"
Выполнение практических заданий по теме
Лабораторная работа №14 "Работа со строковыми переменными"
Выполнение практических заданий по теме
Лабораторная работа №15 "Работа с данными типа множество"
Выполнение практических заданий по теме
Лабораторная работа №16 "Организация методов"
Выполнение практических заданий по теме
Лабораторная работа №17 "Использование функций"
Выполнение практических заданий по теме
Лабораторная работа №18 "Использование стандартных функций "
Выполнение практических заданий по теме
Лабораторная работа №19 "Использование стандартных функций для работы со строками"
Выполнение практических заданий по теме
Лабораторная работа №20 "Использование стандартных функций для работы с массивами"
Выполнение практических заданий по теме

Тема 5. Структуризация в программировании (ОК 01, ОК 2, ОК 9, ПК 2.2, ПК 2.4, ПТВ 1, ПТВ 2,ПТВ 3, ЦНП 1, ЦНП 2, ЦНП 6, ЦНП 7В)

Лабораторная работа №21 "Работа со стандартными подпрограммами"
Выполнение практических заданий по теме
Лабораторная работа №22 "Составление схемы вызова библиотек"
Выполнение практических заданий по теме
Лабораторная работа №23 "Решение задач. Функции"
Выполнение практических заданий по теме

Тема 6. Модульное программирование (ОК 01, ОК 2, ОК 9, ПК 2.2, ПК 2.4, ПТВ 1, ПТВ 2, ПТВ 3,ЦНП 1, ЦНП 2, ЦНП 6, ЦНП 7В)

Лабораторная работа №24 "Работа с файлами различных форматов" Выполнение практическихзаданий по теме
Лабораторная работа №25 "Работа с файлом последовательного доступа"
Выполнение практических заданий по теме
Лабораторная работа №26 "Работа с файлом произвольного доступа"
Выполнение практических заданий по теме
Лабораторная работа №27 "Использование стандартных методов"
Выполнение практических заданий по теме
Лабораторная работа №28 "Использование стандартных методов для работы с файлами"
Выполнение практических заданий по теме
Лабораторная работа №29 "Использование стандартных функций для работы с файлами"
Выполнение практических заданий по теме
Лабораторная работа №30 "Решение задач. Методы"
Выполнение практических заданий по теме
Лабораторная работа №31 "Программирование модуля"
Выполнение практических заданий по теме
Лабораторная работа №32 "Создание библиотеки подпрограмм"
Выполнение практических заданий по теме

Тема 7. Указатели (ОК 01, ОК 2, ОК 9, ПК 2.2, ПК 2.4, ПТВ 1, ПТВ 2, ПТВ 3, ЦНП 1, ЦНП 2, ЦНП 6,ЦНП 7В)

Лабораторная работа №33 "Использование библиотеки подпрограмм"
Выполнение практических заданий по теме
Лабораторная работа №34 "Указатели"
Выполнение практических заданий по теме
Лабораторная работа №35 "Указатели и ссылки"
Выполнение практических заданий по теме
Лабораторная работа №36 "Определение указателя и операции над ним"
Выполнение практических заданий по теме
Лабораторная работа №37 "Прямое обращение по указателю"
Выполнение практических заданий по теме
Лабораторная работа №38 "Косвенное обращение по указателю"
Выполнение практических заданий по теме
Лабораторная работа №39 "Указатели и массивы"
Выполнение практических заданий по теме
Лабораторная работа №40 "Указатели и многомерные массивы"
Выполнение практических заданий по теме
Лабораторная работа №41 "Типичные ошибки при работе с указателями"
Выполнение практических заданий по теме
Лабораторная работа №42 "Сравнение указателей на равенство"
Выполнение практических заданий по теме

Тема 8. Основные принципы объектно-ориентированного программирования (ООП) (ОК 01, ОК 2,ОК 9, ПК 2.2, ПК 2.4, ПТВ 1, ПТВ 2, ПТВ 3, ЦНП 1, ЦНП 2, ЦНП 6, ЦНП 7В)

Лабораторная работа №43 "Использование паттернов ООП" Выполнение практических заданий потеме

Тема 9. Интегрированная среда разработчика (ОК 01, ОК 2, ОК 9, ПК 2.2, ПК 2.4, ПТВ 1, ПТВ 2,ПТВ 3, ЦНП 1, ЦНП 2, ЦНП 6, ЦНП 7В)

Лабораторная работа №44 "Сравнение указателей на «больше-меньше»"
Выполнение практических заданий по теме
Лабораторная работа №45 "Разность значений указателей"
Выполнение практических заданий по теме
Лабораторная работа №46 "Преобразование типа указателя"
Выполнение практических заданий по теме
Лабораторная работа №47 "Строковая константа"
Выполнение практических заданий по теме
Лабораторная работа №48 "Индексация или перемещение указателя"
Выполнение практических заданий по теме
Лабораторная работа №49 "Поиск всех вхождений подстроки в строке"
Выполнение практических заданий по теме
Лабораторная работа №50 "Сортировка слов в строке (выбором)"
Выполнение практических заданий по теме
Лабораторная работа №51 "Рекурсия"
Выполнение практических заданий по теме
Лабораторная работа №52 "Граничный и рекурсивный случай"
Выполнение практических заданий по теме
Лабораторная работа №53 "Стек вызовов"
Выполнение практических заданий по теме
Лабораторная работа №54 "Требования рекурсии"
Выполнение практических заданий по теме
Лабораторная работа №55 "Факториал"
Выполнение практических заданий по теме

Тема 10. Визуальное событийно-управляемое программирование (ОК 01, ОК 2, ОК 9, ПК 2.2, ПК2.4, ПТВ 1, ПТВ 2, ПТВ 3, ЦНП 1, ЦНП 2, ЦНП 6, ЦНП 7В)

Лабораторная работа №56 "Числа Фибоначчи"
Выполнение практических заданий по теме

Лабораторная работа №57 "Задача о Ханойской башне"
Выполнение практических заданий по теме
Лабораторная работа №58 "Решение задач. Рекурсия "
Выполнение практических заданий по теме
Лабораторная работа №59 "Создание простого проекта"
Выполнение практических заданий по теме
Лабораторная работа №60 "Создание проекта с использованием кнопочных компонентов"
Выполнение практических заданий по теме
Лабораторная работа №61 "Создание проекта с использованием компонентов для работы с текстом"
Выполнение практических заданий по теме
Лабораторная работа №62 "Создание проекта с использованием компонентов"
Выполнение практических заданий по теме
Лабораторная работа №63 "Язык визуального программирования"
Выполнение практических заданий по теме
Лабораторная работа №64 "Основы визуального программирования"
Выполнение практических заданий по теме
Лабораторная работа №65 "Применение визуального программирования при построенииинтерфейса приложения"
Выполнение практических заданий по теме
Лабораторная работа №66 "Примеры визуального программирования в известных программныхсредах"
Выполнение практических заданий по теме
Лабораторная работа №67 "Достоинства и недостатки визуального программирования"
Выполнение практических заданий по теме
Лабораторная работа №68 "Этапы реализации объектно-ориентированного подхода"
Выполнение практических заданий по теме
Лабораторная работа №69 "Инкапсуляция"
Выполнение практических заданий по теме
Лабораторная работа №70 "Наследование "
Выполнение практических заданий по теме

7.3. Содержание самостоятельной работы

Тема 9. Интегрированная среда разработчика (ОК 01, ОК 2, ОК 9, ПК 2.2, ПК 2.4, ПТВ 1, ПТВ 2,ПТВ 3, ЦНП 1, ЦНП 2, ЦНП 6, ЦНП 7В)
Решение задач различной сложности с ограничением по времени работы программы и памяти

7.3.1. Примерные вопросы для самостоятельной подготовки к зачету/экзамену
Приложение 1.

7.3.2. Практические задания по дисциплине для самостоятельной подготовки кзачету/экзамену
Приложение 2.

7.3.3. Перечень курсовых работ
Не предусмотрено

7.4. Электронное портфолио обучающегося
Материалы не размещаются.

7.5. Методические рекомендации по выполнению контрольной работы
Не предусмотрено.

7.6 Методические рекомендации по выполнению курсовой работы
Не предусмотрено.

8. ОСОБЕННОСТИ ОРГАНИЗАЦИИ ОБРАЗОВАТЕЛЬНОГО ПРОЦЕССА ПО ДИСЦИПЛИНЕ ДЛЯ ЛИЦ С ОГРАНИЧЕННЫМИ ВОЗМОЖНОСТЯМИ ЗДОРОВЬЯ

По заявлению студента

В целях доступности освоения программы для лиц с ограниченными возможностями здоровья при необходимости кафедра обеспечивает следующие условия:

- особый порядок освоения дисциплины, с учетом состояния их здоровья;
- электронные образовательные ресурсы по дисциплине в формах, адаптированных к ограничениям их здоровья;
- изучение дисциплины по индивидуальному учебному плану (вне зависимости от формы обучения);
- электронное обучение и дистанционные образовательные технологии, которые предусматривают возможности приема-передачи информации в доступных для них формах.
- доступ (удаленный доступ), к современным профессиональным базам данных и информационным справочным системам, состав которых определен РПД.

9. ПЕРЕЧЕНЬ ОСНОВНОЙ И ДОПОЛНИТЕЛЬНОЙ УЧЕБНОЙ ЛИТЕРАТУРЫ, НЕОБХОДИМОЙ ДЛЯ ОСВОЕНИЯ ДИСЦИПЛИНЫ

Сайт библиотеки УрГЭУ
http://lib.usue.ru/

Основная литература:

2. Гуриков С.Р. Основы алгоритмизации и программирования на Python [Электронныйресурс]:Учебное пособие. - Москва: ООО "Научно-издательский центр ИНФРА-М", 2025. - 343 –Режим доступа: https://znanium.com/catalog/product/2186214

3. Трофимов В. В., Павловская Т. А. Основы алгоритмизации и программирования[Электронный ресурс]:учебник для спо. - Москва: Юрайт, 2025. - 108 – Режим доступа:https://urait.ru/bcode/563861

4. Кудрина Е. В., Огнева М. В. Основы алгоритмизации и программирования на языке C#[Электронный ресурс]:учебник для спо. - Москва: Юрайт, 2025. - 322 – Режим доступа:https://urait.ru/bcode/565504

5. Жуков Р.А. Язык программирования Python. Практикум [Электронный ресурс]:Учебноепособие. - Москва: ООО "Научно-издательский центр ИНФРА-М", 2026. - 216 – Режим доступа:https://znanium.com/catalog/product/2216925

6. Палий И. А. Линейное программирование [Электронный ресурс]:учебник для спо. - Москва: Юрайт, 2025. - 175 – Режим доступа: https://urait.ru/bcode/568837

Дополнительная литература:

2. Кудрявцева И., Швецкий М. В. Программирование: теория типов [Электронныйресурс]:учебное пособие для спо. - Москва: Юрайт, 2024. - 652 – Режим доступа:https://urait.ru/bcode/542169

3. Гниденко И. Г., Павлов Ф. Ф., Федоров Д. Ю. Технология разработки программногообеспечения [Электронный ресурс]:учебник для спо. - Москва: Юрайт, 2025. - 248 – Режим доступа:https://urait.ru/bcode/563151

4. Казанский А. А. Программирование на C# [Электронный ресурс]:учебное пособие для спо.- Москва: Юрайт, 2025. - 181 – Режим доступа: https://urait.ru/bcode/569863

5. Кудрявцева И., Швецкий М. В. Программирование: комбинаторная логика [Электронныйресурс]:учебник для спо. - Москва: Юрайт, 2025. - 524 – Режим доступа:

6. Маркин А. В. Программирование на SQL [Электронный ресурс]:учебник для спо. - Москва:Юрайт, 2025. - 435 – Режим доступа: https://urait.ru/bcode/566220

7. Гуриков С.Р. Основы алгоритмизации и программирования на языке Microsoft Visual Basic[Электронный ресурс]:Учебное пособие. - Москва: ООО "Научно-издательский центр ИНФРА-М",2025. - 594 – Режим доступа: https://znanium.com/catalog/product/2196851

10. ПЕРЕЧЕНЬ ИНФОРМАЦИОННЫХ ТЕХНОЛОГИЙ, ВКЛЮЧАЯ ПЕРЕЧЕНЬ ЛИЦЕНЗИОННОГО ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ И ИНФОРМАЦИОННЫХ СПРАВОЧНЫХ СИСТЕМ, ОНЛАЙН КУРСОВ, ИСПОЛЬЗУЕМЫХ ПРИ ОСУЩЕСТВЛЕНИИ ОБРАЗОВАТЕЛЬНОГО ПРОЦЕССА ПО ДИСЦИПЛИНЕ

Перечень лицензионного программного обеспечения:

Astra Linux Common Edition. Договор №0417-ПО/2019 от 08.05.2019, Акт №Sk000343 от 24.05.2019 и Контракт № 35-У/2018 от 13.06.2018, Акт № УТ213 от 17.12.2018. Срок действия лицензии - без ограничения срока.

МойОфис стандартный. Соглашение № СК-281 от 7 июня 2017. Дата заключения - 07.06.2017. Срок действия лицензии - без ограничения срока.

Libre Office. Лицензия GNU LGPL. Срок действия лицензии - без ограничения срока.

Inkscape. Лицензия GNU GENERAL PUBLIC LICENSE. Срок действия лицензии - без ограничения срока.

Microsoft Dynamics CRM. Соглашение от 23.08.2016.

Microsoft Visual Studio Community. Лицензия для образовательных учреждений. Срок действия лицензии - без ограничения срока.

Microsoft SQL Server Express. Лицензия для образовательных учреждений. Срок действия лицензии - без ограничения срока.

Vortex. Акт предоставления прав № Tr024234 от 24.04.2017.

Directum RX. Соглашение № 0045-25 от 28.08.2025. Срок действия лицензии 28.08.2026.

Язык программирования R.Лицензия GNU GPL 2.Срок действия лицензии - без ограничения срока.

R Studio (среда для языка программирования R).Лицензия GNU Affero General Public License v3.Срок действия лицензии - без ограничения срока.

Язык программирования Python.Python Software Foundation License (PSFL). Срок действия лицензии - без ограничения срока.

Эмулятор GNS 3.Лицензия GNU GPL. Срок действия лицензии - без ограничения срока.

Архиватор 7-Zip. Лицензия GNU LGPLv2.1 + with unRAR restriction / LZMA SDK in the public domain. Срок действия лицензии - без ограничения срока.

FAR Manager. Лицензия Revised BSD license. Срок действия лицензии - без ограничения срока.

Notepad++. Лицензия GNU General Public License. Срок действия лицензии - без ограничения срока.

Adobe Reader. Лицензия freeeware. Срок действия лицензии - без ограничения срока.

Язык программирования Java.

Intellij IDEA.

Система контроля версий Git. Лицензия GNU GPL v2 and GNU LGPL v2.1. Срок действия лицензии - без ограничения срока.

Перечень информационных справочных систем, ресурсов информационно-телекоммуникационной сети «Интернет»:

11. ОПИСАНИЕ МАТЕРИАЛЬНО-ТЕХНИЧЕСКОЙ БАЗЫ, НЕОБХОДИМОЙ ДЛЯ ОСУЩЕСТВЛЕНИЯ ОБРАЗОВАТЕЛЬНОГО ПРОЦЕССА ПО ДИСЦИПЛИНЕ

Реализация учебной дисциплины осуществляется с использованием материально-технической базы УрГЭУ, обеспечивающей проведение всех видов учебных занятий и научно-исследовательской и самостоятельной работы обучающихся:

Специальные помещения представляют собой учебные аудитории для проведения всех видов занятий, групповых и индивидуальных консультаций, текущего контроля и промежуточной аттестации.

Помещения для самостоятельной работы обучающихся оснащены компьютерной техникой с возможностью подключения к сети "Интернет" и обеспечением доступа в электронную информационно-образовательную среду УрГЭУ.

Все помещения укомплектованы специализированной мебелью и оснащены мультимедийным оборудованием спецоборудованием (информационно-телекоммуникационным, иным компьютерным), доступом к информационно-поисковым, справочно-правовым системам, электронным библиотечным системам, базам данных действующего законодательства, иным информационным ресурсам служащими для представления учебной информации большой аудитории.

Для проведения занятий лекционного типа презентации и другие учебно-наглядные пособия, обеспечивающие тематические иллюстрации.

Приложение 1
к рабочей программе

7.3.1. Примерные вопросы для самостоятельной подготовки к зачету

1. Что такое переменная? Какие типы данных существуют в C#?
2. Как объявить и инициализировать переменную в C#?
3. Что такое константа? Как объявить константу в C#?
4. Какие операторы ветвления существуют в C#? Приведите примеры использования if, else, switch.
5. Что такое циклы? Какие виды циклов существуют в C#? Приведите примеры использования for, while, do-while.
6. Как работает оператор break и continue в циклах?
7. Что такое массивы? Как объявить и инициализировать массив в C#?
8. Как работать с многомерными массивами в C#?
9. Что такое строки? Какие методы работы со строками вы знаете?
10. Что такое функция (метод) в C#? Как объявить и вызвать метод?
11. Какие типы возвращаемых значений могут быть у методов?
12. Что такое параметры метода? Какие виды параметров существуют (входные, выходные, ссылочные)?
13. Что такое перегрузка методов? Приведите пример.
14. Что такое рекурсия? Приведите пример рекурсивного метода.
15. Что такое класс и объект? Как создать класс и объект в C#?
16. Что такое конструктор? Какие виды конструкторов существуют?
17. Что такое наследование? Как реализовать наследование в C#?
18. Что такое полиморфизм? Приведите пример использования виртуальных методов и переопределения.
19. Что такое инкапсуляция? Как реализовать инкапсуляцию в C#?
20. Что такое интерфейсы? Как их использовать в C#?
21. Что такое абстрактные классы? Чем они отличаются от интерфейсов?
22. Что такое исключение? Как обрабатывать исключения в C#?
23. Какие блоки используются для обработки исключений (try, catch, finally)?
24. Как создать собственное исключение?
25. Что такое коллекции? Какие виды коллекций вы знаете (списки, словари, очереди, стеки)?
26. Как работать с коллекцией List<T>? Приведите пример.
27. Что такое LINQ? Как использовать LINQ для работы с коллекциями?
28. Приведите пример использования LINQ для фильтрации, сортировки и группировки данных.
29. Как читать и записывать данные в текстовый файл в C#?
30. Что такое потоки (Stream)? Как работать с потоками в C#?
31. Как работать с бинарными файлами?
32. Что такое делегаты и события? Как их использовать в C#?
33. Что такое лямбда-выражения? Приведите пример.
34. Что такое асинхронное программирование? Как использовать async и await?
35. Что такое обобщения (generics)? Как их использовать в C#?

7.3.1. Примерные вопросы для самостоятельной подготовки к экзамену

1. Что такое C#? Какие основные особенности языка?
2. Какова структура программы на C#? Что такое метод Main?
3. Какие типы данных существуют в C#? Приведите примеры.
4. Что такое переменная? Как объявить переменную в C#?
5. Что такое константа? Как объявить константу в C#?
6. Какие операторы присваивания существуют в C#?
7. Что такое условные операторы? Приведите примеры использования if, else, else if.
8. Как работает оператор switch? Приведите пример.
9. Что такое циклы? Какие виды циклов существуют в C#?
10. Как работают операторы break и continue в циклах?
11. Что такое массивы? Как объявить и инициализировать массив?
12. Как работать с многомерными массивами?
13. Что такое строки? Какие методы работы со строками вы знаете?
14. Что такое StringBuilder и зачем он нужен?
15. Что такое метод в C#? Как объявить и вызвать метод?
16. Какие типы возвращаемых значений могут быть у методов?
17. Что такое параметры метода? Какие виды параметров существуют?
18. Что такое перегрузка методов? Приведите пример.
19. Что такое рекурсия? Приведите пример рекурсивного метода.
20. Что такое локальные и глобальные переменные?
21. Что такое область видимости переменных?
22. Что такое класс и объект? Как создать класс и объект в C#?
23. Что такое конструктор? Какие виды конструкторов существуют?
24. Что такое наследование? Как реализовать наследование в C#?
25. Что такое полиморфизм? Приведите пример использования виртуальных методов и переопределения.
26. Что такое инкапсуляция? Как реализовать инкапсуляцию в C#?
27. Что такое интерфейсы? Как их использовать в C#?
28. Что такое абстрактные классы? Чем они отличаются от интерфейсов?
29. Что такое статические классы и методы? Приведите пример.
30. Что такое свойства (properties)? Как их использовать?
31. Что такое перечисления (enum)? Приведите пример.
32. Что такое исключение? Как обрабатывать исключения в C#?
33. Какие блоки используются для обработки исключений (try, catch, finally)?
34. Как создать собственное исключение?
35. Что такое throw и throws?
36. Что такое коллекции? Какие виды коллекций вы знаете?
37. Как работать с коллекцией List<T>? Приведите пример.
38. Что такое Dictionary<TKey, TValue>? Как его использовать?
39. Что такое Queue и Stack? Приведите примеры.
40. Что такое LINQ? Как использовать LINQ для работы с коллекциями?
41. Приведите пример использования LINQ для фильтрации данных.
42. Как сортировать данные с помощью LINQ?
43. Что такое лямбда-выражения? Приведите пример.
44. Как читать и записывать данные в текстовый файл?
45. Что такое потоки (Stream)? Как работать с потоками в C#?
46. Как работать с бинарными файлами?
47. Что такое FileStream, StreamReader и StreamWriter?
48. Что такое делегаты? Как их использовать?

49. Что такое события? Приведите пример.
50. Что такое асинхронное программирование? Как использовать async и await?
51. Что такое обобщения (generics)? Как их использовать?
52. Что такое атрибуты? Приведите пример.
53. Что такое рефлексия? Как она используется в C#?
54. Что такое Nullable типы? Зачем они нужны?
55. Что такое var и dynamic? В чем разница между ними?
56. Напишите программу, которая находит сумму элементов массива.
57. Напишите программу, которая сортирует массив методом пузырьковой сортировки.
58. Напишите программу, которая рекурсивно вычисляет факториал числа.
59. Напишите программу, которая читает данные из файла и выводит их на экран.
60. Напишите программу, которая использует LINQ для поиска элементов в коллекции.
61. Напишите программу, которая реализует простой калькулятор.
62. Напишите программу, которая демонстрирует работу с наследованием.
63. Напишите программу, которая обрабатывает исключения при делении на ноль.
64. Напишите программу, которая создает и использует интерфейс.
65. Напишите программу, которая работает с Dictionary для хранения и поиска данных.
66. В чем разница между == и .Equals()?
67. В чем разница между String и StringBuilder?
68. В чем разница между List и Array?
69. В чем разница между ref и out параметрами?
70. В чем разница между abstract классом и interface?
71. В чем разница между value типами и reference типами?
72. В чем разница между async и sync методами?
73. В чем разница между IEnumerable и IQueryable?
74. В чем разница между throw и throw ex?
75. В чем разница между const и readonly?
76. Что такое Task и как его использовать?
77. Что такое async/await? Приведите пример.
78. Что такое yield return? Как его использовать?
79. Что такое extension methods? Приведите пример.
80. Что такое anonymous types? Как их использовать?
81. Что такое pattern matching? Приведите пример.
82. Что такое records в C# 9/10? Зачем они нужны?
83. Что такое nullable reference types? Как их использовать?
84. Что такое global using в C# 10?
85. Что такое file-scoped namespaces в C# 10?

Приложение 2
к рабочей программе

7.3.2. Практические задания для самостоятельной подготовки к экзамену

Тестовые задания

№ задания	Содержание задания	Правильный ответ
ОК 01.: Выбирать способы решения задач профессиональной деятельности применительно к различным контекстам		
	Открытые задания	

1	Подставьте правильные имена функций в выражение const result =____(3, 2) +____(5); // 30, записав в ответ 2 слова через запятую, если результатом выражения должно получиться число 30. Функции определены так:
	const sum = (a, b) => a + b;
	const square = num => num ** 2;
		sum, square

2	Какая строка получится в результате выполнения конкатенации "sun" + "day" + "two"?	sundaytwo
3	Какой индекс у буквы "t" в строке "laptop"?	3

4	Какое значение у переменной x после выполнения, представленного ниже кода?
	int x = 45;
	int height = x++;
		46

5	Чему равно значение выражения 14 % 4?	2
	Закрытые задания	
1	Арифметические операции в C# можно выполнять с константами, без участия самих чисел. Например, a + b + age + weight. Верно? 1. Нет. Нужно как минимум одно настоящее число в вычислении 2. Да, если у этих констант численные значения 3. Да, с любыми типами данных 4. Нет, нужны только числовые значения	2
2	Какими свойствами обладает оператор - (знак минуса) в выражении -41? *(нужно выбрать все корректные ответы)* 1. он бинарный 2. он постфиксный 3. он инфиксный 4. он префиксный 5. он унарный	4, 5
3	Как можно возвести двойку в третью степень? *(нужно выбрать все корректные ответы)* 1. 2 ^ 3	1, 2, 3

	2. 2 ** 3 3. Math.pow(2, 3)	
4	Что будет, если ни одно условие из блоков if и else if не будет истинным? 1. Ошибка компиляции 2. Программа продолжит выполнение, пропустив блоки if-else 3. Выполнение блока else 4. Выполнится первое условие if	2
5	Возможно ли написать рекурсивную функцию, которая вычисляет сумму серии чисел? (например, сумму чисел от 1 до 100) 1) Да 2) Нет 3) Можно для чисел кратных степени 2 4) Рекурсия неприменима в данном контексте	1
ОК 02.: Использовать современные средства поиска, анализа и интерпретации информации и информационные технологии для выполнения задач профессиональной деятельности;		
Открытые вопросы		

1	Какое слово необходимо вписать, чтобы функция, представленная ниже вычисляла и возвращяла квадрат числа:
	const square = (x) => {
	_____x * x;
	};
		return

2	Сколько аргументов принимает функция full, в представленном ниже отрывке кода? def full(first, second, third): return first * second * third	3
3	Функция sum_() принимает два числа и возвращает их сумму. Взгляните на код ниже. Какое число будет выведено на экран? result = sum_(sum_(1, 3), sum_(sum_(4, 2), 3)) print(result) # ?	13
4	Операции сравнения и логические операции, какая операция (символ) возвращают логическое значение, то есть значение типа bool: true, если в результате сравнения двух операндов возвращается значение true, если операнды не равны, и false, если они равны. (укажите только условное выражение, например, <=)	!=
5	Для чего служит раздел VAR?	Для описания используемых переменных
Закрытые вопросы		
1	Может ли программа содержать несколько функций, методов и классов? Выбрать вариант ответа: 1. Несколько может быть только функций и методов 2. Несколько может быть только классов и методов	5

	3. Несколько может быть только функций и классов 4. Нет, может быть только одна функция, метод, класс 5. Да, методов, классов и функций может быть несколько в одной программе	
2	Могут ли функции возвращать строки (тексты), или они могут возвращать исключительно числа? Выбрать вариант ответа 1. Функции ничего не возвращают 2. Только числовые значения 3. И числа, и строки 4. Не могут	3

3	Что будет "сохранено" в константе *y* после выполнения в представленном ниже коде? Выбрать правильный вариант ответа:
	const someFunction = (x) => {
	return 10 * 92;};
	const y = someFunction(3546);
	1. Функция неправильная, потому что аргумент x был передан в функцию, но не использовался внутри.
	2. 3546
	3. 920
	4. 0
		3

4	Можно ли использовать if для проверки одного условия, без else if и без else? 1. Можно 2. Нет, как минимум, нужен else 3. Нет, как минимум, нужен else if 4. Нет, обязательно нужны и if и else	1
5	Я смотрю на логический оператор. В нем два выражения и операция между ними. Одно из выражений истинно (true), второе выражение неизвестно, и результат операции — истина. Какой это может быть операция? (нужно выбрать все корректные ответы) 1. OR 2. AND 3. NOT 4. EQUALS	1, 2
ОК 09.: Пользоваться профессиональной документацией на государственном и иностранном языках		
Открытые вопросы		
1	Объясните, зачем важно уметь пользоваться профессиональной документацией на иностранном языке при изучении новых библиотек или API.	Большинство современных библиотек и фреймворков имеют документацию на английском языке. Умение её читать позволяет

		быстрее понять функционал, правильно использовать функции, находить примеры использования и избегать ошибок.
2	Приведите пример документации алгоритма или функции, где вы видите описание входных и выходных параметров. Объясните, как эти сведения помогают в программировании.	В документации Python функция sorted(iterable, key=None, reverse=False) имеет входные параметры iterable, key, reverse и возвращает отсортированный список. Это позволяет программисту правильно использовать функцию, подставляя нужные аргументы.
3	Опишите, какие элементы технической документации помогают понять логику работы сложной программы.	Блок-схемы, псевдокод, диаграммы UML, таблицы входных и выходных данных, комментарии к функциям и алгоритмам — все это помогает визуализировать процесс выполнения программы и правильно реализовать её.
4	Как вы используете спецификации алгоритмов в формате UML для проектирования программных модулей?	Диаграммы UML помогают определить классы, методы и связи между ними. На основе этих спецификаций создаются модули программы с правильной структурой и взаимодействием.
5	Приведите пример, когда вы пользовались официальной документацией на английском языке для изучения возможностей стандартной библиотеки какого-либо языка программирования	Использовал документацию Java java.util.Collections для изучения методов sort(), reverse() и shuffle(). Благодаря описанию на английском языке удалось понять параметры и исключения функций.
6	Объясните различие между инструкциями в документации и комментариями в исходном коде.	Инструкции документации предназначены для пользователей библиотеки или разработчиков, чтобы понять использование функций. Комментарии в коде помогают понять, как именно реализован алгоритм, и чаще служат внутренней справкой.
7	Как вы проверяете корректность алгоритма, опираясь на документацию?	Сравниваю результаты работы алгоритма с тестовыми примерами, проверяю соответствие типов данных и ограничений, указанных в документации, и убеждаюсь, что все параметры используются правильно.
8	Опишите процесс перевода профессиональной документации с иностранного языка на русский. Какие трудности могут возникнуть при этом?	Перевод включает понимание термина, контекста и примеров использования. Трудности: технические термины без точного

		аналога, длинные сложные предложения, возможные неоднозначности и различия в синтаксисе языка программирования.
9	Какие профессиональные ресурсы вы используете для поиска документации по алгоритмам и программированию?	Официальные сайты языков программирования (Python.org, JavaDoc), ресурсы GitHub, Stack Overflow, стандарты ISO/IEC, документация библиотек и API.
10	Представьте, что вы разрабатываете алгоритм сортировки. Как вы оформили бы документацию для других программистов, чтобы она была понятна и на русском, и на английском языке?	Ответ: Описание алгоритма включало бы: ~ Название и краткое назначение (на русском и английском) ~ Входные и выходные параметры ~ Псевдокод ~ Блок-схему ~ Примеры работы с тестовыми данными ~ Возможные ошибки и ограничения
Закрытые вопросы		
1	Какой формат файлов обычно используется для хранения спецификаций алгоритмов в профессиональной документации? 1. .exe 2. .docx 3. .alg 4. .txt	2
2	В технической документации по программированию английский термин “function” соответствует какому понятию на русском языке? 1. Процедура 2. Функция 3. Класс 4. Метод	2
3	Какой из следующих стандартов описывает формат документации к программному обеспечению? 1. ISO/IEC 12207 2. IEEE 754 3. HTML5 4. CSS3	1
4	Что обозначает аббревиатура API в профессиональной документации? 1. Advanced Programming Interface 2. Application Programming Interface 3. Automated Process Instruction 4. Algorithmic Procedure Index	Application Programming Interface

5	В документации по алгоритмам блок-схемы чаще всего применяются для: 1. Хранения исходного кода 2. Визуального описания алгоритма 3. Тестирования программного продукта 4. Создания базы данных	2
ПК 2.2.: Разрабатывать модули программного обеспечения		
Открытые вопросы		
1	Что такое модуль программного обеспечения и какие преимущества дает модульная структура?	Модуль — это отдельный фрагмент кода, выполняющий конкретную функцию. Модульная структура повышает читаемость, упрощает тестирование, поддержку и повторное использование кода
2	Опишите процесс разработки модуля от постановки задачи до тестирования	Сначала формулируются требования, затем создается дизайн модуля (функции, интерфейс), реализуется код, пишутся юнит-тесты, проверяется корректность работы, после чего модуль интегрируется в систему
3	Приведите пример модульного разделения задачи на части в программе сортировки или поиска	Для сортировки можно выделить модуль: функция сравнения элементов, функция обмена элементов, функция самой сортировки. Для поиска: модуль поиска, модуль проверки данных, модуль вывода результата
4	Какие принципы проектирования модулей помогают создавать качественный код?	Принципы SOLID, инкапсуляция, высокая связность внутри модуля и низкая зависимость между модулями, ясный интерфейс и документация
5	Как интерфейс модуля влияет на его повторное использование?	Четко описанный интерфейс позволяет другим программистам использовать модуль без знания внутренней реализации, что упрощает повторное использование кода
6	Опишите, как модуль тестируется отдельно от всей системы	Пишутся юнит-тесты для всех функций модуля, проверяется корректность работы с разными входными данными и границами, исправляются ошибки до интеграции
7	Приведите пример модуля с несколькими функциями и объясните, как они взаимодействуют	Модуль “Калькулятор”: функции add(), subtract(), multiply(), divide() работают с общим объектом CalculatorData. Взаимодействие через параметры и возвращаемые

		значения
8	Что такое зависимость между модулями и как её уменьшить?	Зависимость — когда один модуль напрямую использует внутренности другого. Уменьшают её с помощью инкапсуляции, интерфейсов, абстракций и событийного взаимодействия.
9	Опишите, как документация модуля помогает другим разработчикам использовать его.	Документация описывает функции, входные и выходные данные, ограничения, исключения, примеры использования — это позволяет быстрее интегрировать модуль без изучения исходного кода.
10	Представьте, что вам нужно интегрировать новый модуль в существующую систему. Какие шаги вы предпримете?	Проверяю интерфейс модуля, создаю тестовые сценарии, запускаю модуль с существующими данными, проверяю взаимодействие с другими модулями, исправляю конфликты, обновляю документацию и тесты.
Закрытые вопросы		
1	Что такое модуль программного обеспечения? 1. Отдельная программа, которая не взаимодействует с другими 2. Фрагмент кода, реализующий конкретную функцию в составе системы 3. Тестовый скрипт для проверки программы 4. Пользовательский интерфейс	2
2	Какая из следующих практик улучшает модульность программного обеспечения? 1. Разработка длинного монолитного кода 2. Использование функций и классов для отдельных задач 3. Игнорирование документации 4. Разработка без тестирования	2
3	Что из перечисленного относится к интерфейсу модуля? 1. Внутренние переменные, недоступные извне 2. Функции и методы, доступные для других модулей 3. Стили оформления кода 4. Файлы документации	2
4	Для чего используются юнит-тесты при разработке модулей? 1. Для проверки работы отдельных модулей программы 2. Для анализа требований заказчика 3. Для изменения интерфейса программы	1

	4. Для генерации документации	
5	Какая концепция помогает уменьшить зависимость между модулями? 1. Инкапсуляция 2. Монотонность 3. Комментарии в коде 4. Использование глобальных переменных	1
ПК 2.4.: Выполнять тестирование и отладку программного обеспечения		
Открытые вопросы		
1	Что такое тестирование программного обеспечения и зачем оно нужно?	Тестирование — процесс проверки программы на правильность, соответствие требованиям и отсутствие ошибок. Оно позволяет выявить дефекты до передачи продукта пользователю.
2	Опишите основные этапы тестирования и отладки программы.	1. Подготовка тестовых сценариев 2. Юнит-тестирование модулей 3. Интеграционное тестирование 4. Системное тестирование 5. Отладка найденных ошибок и повторная проверка
3	Приведите примеры инструментов, используемых для тестирования и отладки.	Дебаггеры (Visual Studio Debugger, GDB), фреймворки для юнит-тестирования (JUnit, PyTest), статический анализ кода, логирование, профилировщики.
4	Что такое логирование и как оно помогает в отладке?	Логирование — запись событий выполнения программы в файл или консоль. Помогает понять последовательность действий и найти причину ошибки.
5	Опишите различие между статическим и динамическим тестированием.	Статическое тестирование — проверка кода без его выполнения (анализ, ревью, линтеры). Динамическое тестирование — запуск программы с тестовыми данными для выявления ошибок.
6	Приведите пример логической ошибки и объясните, как её можно выявить.	При суммировании чисел программа всегда возвращает ноль. Выявляется тестированием с различными входными данными и сравнением результатов с ожидаемыми.
7	Что такое регрессионное тестирование и когда оно используется?	Регрессионное тестирование проверяет, что исправление одной ошибки не привело к новым дефектам. Используется после внесения изменений в программу.

8	Опишите процесс отладки программы с использованием дебаггера.	Устанавливаются точки останова (breakpoints), программа запускается пошагово, отслеживаются значения переменных, выявляются несоответствия и ошибки, затем исправляется код.
9	Как тестовые данные помогают выявить ошибки в программе?	Тестовые данные позволяют проверить работу алгоритмов в различных сценариях, включая граничные значения, пустые или некорректные данные, что помогает выявить ошибки.
10	Представьте, что модуль программы не работает как ожидалось. Какие шаги вы предпримете для его отладки?	1. Анализ требований и логики модуля 2. Проверка синтаксиса и структуры кода 3. Создание и запуск тестов с различными данными 4. Использование дебаггера для пошагового выполнения 5. Исправление ошибок и повторная проверка
Закрытые вопросы		
1	Что такое юнит-тестирование? 1. Проверка всего программного продукта целиком 2. Проверка отдельных функций или модулей программы 3. Тестирование интерфейса пользователя 4. Проверка документации	2
2	Какой из методов тестирования направлен на проверку соответствия программы требованиям заказчика? 1. Интеграционное тестирование 2. Системное тестирование 3. Юнит-тестирование 4. Статический анализ	2
3	Что из перечисленного относится к типам ошибок, которые выявляются при отладке? 1. Синтаксические, логические и ошибки выполнения 2. Только синтаксические 3. Только логические 4. Только ошибки интерфейса	1
4	Какой инструмент помогает отслеживать выполнение программы по шагам? 1. Компилятор 2. Дебаггер 3. Текстовый редактор 4. Git	2

5	Что такое интеграционное тестирование? 1. Проверка отдельных функций 2. Проверка совместной работы нескольких модулей 3. Тестирование пользовательского интерфейса 4. Проверка документации	2

7.3.2. Практические задания по междисциплинарному курсу для самостоятельной подготовки к экзамену

1. Написать программу, вычисляющую значение функции: f(x)=5x^2−3x+7 при вводе пользователем числа x.
2. Написать программу, которая запрашивает у пользователя два числа и выводит большее из них.
3. Реализовать программу, которая запрашивает у пользователя число n и выводит сумму всех чисел от 1 до n.
4. Написать программу, которая запрашивает у пользователя число и выводит его факториал.
5. Создать список из 10 случайных чисел и найти минимальное и максимальное значение.
6. Написать программу, которая определяет, является ли введенное число простым.
7. Реализовать рекурсивную функцию для вычисления числа Фибоначчи.
8. Написать программу, которая генерирует и выводит первые 10 чисел последовательности Фибоначчи.
9. Реализовать сортировку массива пузырьком или методом выбора.
10. Реализовать алгоритм бинарного поиска в отсортированном массиве.
11. Написать функцию, которая принимает два числа и возвращает их наибольший общий делитель (НОД).
12. Написать программу, которая проверяет, является ли введенное слово палиндромом.
13. Написать программу, которая принимает список чисел и удаляет из него все четные элементы.
14. Создать словарь с названиями месяцев и количеством дней в них, затем вывести его.
15. Написать программу, которая читает текстовый файл и считает количество слов в нем.
16. Создать список квадратов чисел от 1 до 20 с использованием list comprehension.
17. Написать программу, которая запрашивает у пользователя число и обрабатывает ошибку, если введены нечисловые данные.
18. Написать программу, которая принимает список чисел и возвращает новый список, содержащий только числа, кратные 3.
19. Создать класс "Круг" с атрибутом радиуса и методом вычисления площади.
20. Создать класс "Прямоугольник" и унаследовать от него класс "Куб", добавив метод вычисления объема.
21. Реализовать стек (LIFO) с методами push(), pop() и is_empty().
22. Реализовать очередь (FIFO) с методами enqueue(), dequeue() и is_empty().
23. Создать класс односвязного списка и реализовать методы добавления и удаления элементов.
24. Реализовать простую хеш-таблицу с методом разрешения коллизий цепочками.
25. Реализовать поиск в глубину (DFS) и поиск в ширину (BFS) для представленного в виде списка смежности графа.
26. Написать программу, которая загружает данные из JSON-файла и выводит их в удобочитаемом формате.
27. Использовать библиотеку для парсинга заголовков новостей с веб-страницы.
28. Создать базу данных SQLite и добавить в нее таблицу "Студенты" с полями "ФИО", "Возраст", "Группа".
29. Написать программу, которая извлекает всех студентов старше 20 лет из базы данных SQLite.
30. Написать скрипт, который отправляет GET-запрос к публичному API и выводит полученные данные.
31. Написать тесты для функции вычисления НОД с использованием unittest.
32. Добавить логирование в программу обработки файлов, фиксируя ошибки и события.

33. Найти неэффективный участок кода в программе и оптимизировать его.
34. Описать код программы с использованием docstrings и комментариев.
35. Создать репозиторий, закоммитить код программы и отправить его на GitHub.