ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ
ВЫСШЕГО ОБРАЗОВАНИЯ ЛЕНИНГРАДСКОЙ ОБЛАСТИ
**«ЛЕНИНГРАДСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ
ИМЕНИ А. С. ПУШКИНА»**

РАБОЧАЯ ПРОГРАММА
учебной дисциплины

ОП.04 ОСНОВЫ АЛГОРИТМИЗАЦИИ И ПРОГРАММИРОВАНИЯ

по специальности среднего профессионального образования
09.02.07 Информационные системы и программирование
(общеобразовательная подготовка)

(год начала подготовки – 2025)

Санкт-Петербург
2025

Программа учебной дисциплины **«Основы алгоритмизации и программирования»** является частью основной профессиональной образовательной программы по специальности **09.02.07 «Информационные системы и программирование»,** составлена в соответствии с требованиями ФГОС СПО и примерной основной образовательной программы по специальности.

Организация-разработчик: ГАОУ ВО ЛО «ЛГУ им. А.С. Пушкина».

Разработчик: Талантов Илья Анатольевич, преподаватель ГАОУ ВО ЛО «ЛГУ им. А.С. Пушкина».

Рассмотрено на заседании ПЦК информационных, экономических и естественно - научных дисциплин

Протокол № 2 от «11» октября 2024 г.

1. ПАСПОРТ РАБОЧЕЙ ПРОГРАММЫ УЧЕБНОЙ ДИСЦИПЛИНЫ

1.1. Область применения рабочей программы

Рабочая программа учебной дисциплины «Основы алгоритмизации и программирования» является частью основной профессиональной образовательной программы подготовки специалистов среднего звена в соответствии с ФГОС по специальности СПО 09.02.07 Информационные системы и программирование, базовая подготовка.

Обучение по дисциплине ведется на русском языке.

При реализации программы учебной дисциплины методы и средства обучения и воспитания, образовательные технологии не могут наносить вред физическому или психическому здоровью обучающихся

При реализации программы учебной дисциплины методы и средства обучения и воспитания, образовательные технологии не могут наносить вред физическому или психическому здоровью обучающихся.

Воспитание обучающихся при освоении учебной дисциплины осуществляется на основе включаемых в образовательную программу рабочей программы воспитания и календарного плана воспитательной работы на текущий учебный год.

Воспитательная деятельность, направлена на развитие личности, создание условий для самоопределения и социализации обучающихся на основе социокультурных, духовно нравственных ценностей и принятых в российском обществе правил и норм поведения в интересах человека, семьи, общества и государства, формирование у обучающихся чувства патриотизма, гражданственности, уважения к памяти защитников Отечества и подвигам Героев Отечества, закону и правопорядку, человеку труда и старшему поколению, взаимного уважения, бережного отношения к культурному наследию и традициям многонационального народа Российской Федерации, природе и окружающей среде.

1.2. Место дисциплины в структуре основной профессиональной образовательной программы

Дисциплина «Основы алгоритмизации и программирования» относится к общепрофессиональному учебному циклу.

Дисциплина имеет межпредметные связи с междисциплинарными курсами «Разработка программных модулей» и «Технология разработки программного обеспечения».

1.3. Цели и задачи учебной дисциплины – требования к результатам освоения учебной дисциплины

Целью освоения дисциплины является освоение методов решения задач на ЭВМ с использованием алгоритмических языков высокого уровня, изучением жизненного цикла и этапов разработки программного обеспечения, приобретением практических навыков по разработке, отладке и тестированию программного продукта, применения средств вычислительной техники для решения практических задач, связанных с накоплением, хранением и обработкой информации.

В результате освоения дисциплины обучающийся осваивает элементы компетенций:

Код компетенции	Планируемые результаты обучения
ОК 01 ОК 02	Знать: – понятие алгоритмизации, свойства алгоритмов, общие принципы

ОК 04 ОК 05 ОК 09 ПК 1.1- ПК 1.5 ПК 2.4, 2.5	построения алгоритмов, основные алгоритмические конструкции; – эволюцию языков программирования, их классификацию, понятие системы программирования; – основные элементы языка, структуру программы, операторы и операции, управляющие структуры, структуры данных, файлы, классы памяти; – подпрограммы, составление библиотек подпрограмм; – объектно-ориентированную модель программирования, основные принципы объектно-ориентированного программирования на примере алгоритмического языка: понятие классов и объектов, их свойств и методов, инкапсуляции и полиморфизма, наследования и переопределения. Уметь: – разрабатывать алгоритмы для конкретных задач; – использовать программы для графического отображения алгоритмов; – определять сложность работы алгоритмов; – работать в среде программирования; – моделировать прикладные задачи; – реализовывать построенные алгоритмы в виде программ на конкретном языке программирования; – оформлять код программы в соответствии со стандартом кодирования; – выполнять проверку, отладку кода программы.

1.4. Количество часов на освоение учебной дисциплины

Образовательная учебная нагрузка обучающегося составляет 172 часа, в том числе:
обязательная аудиторная учебная нагрузка обучающегося - 147 часов;
консультации - 2 часа;
самостоятельная работа обучающегося - 5 часов;
промежуточная аттестация (экзамен) 18 часов, в том числе:
- консультации к экзамену – 2 часа;
- самостоятельная работа обучающегося – 12 часов.

2. СТРУКТУРА И СОДЕРЖАНИЕ УЧЕБНОЙ ДИСЦИПЛИНЫ

2.1. Объем учебной дисциплины и виды учебных работ

Вид учебной работы	Объем часов
Максимальная учебная нагрузка (всего)	**172**
Обязательная аудиторная учебная нагрузка (всего)	**147**
в том числе	
теоретические занятия	49
практические занятия	98
Самостоятельная работа обучающегося (всего)	**5**
Консультации	**2**
Промежуточная аттестация	**18**
в том числе	
консультации к экзамену	2
самостоятельная работа обучающихся	12
Промежуточная аттестация: другая форма контроля (3 семестр), экзамен (4 семестр)	

В соответствии со структурой учебной дисциплины ниже приведена содержательная характеристика дисциплины по всем видам учебной деятельности обучающегося.

2.2. Тематический план и содержание учебной дисциплины

Наименование разделов и тем	Содержание учебного материала и формы организации деятельности обучающихся	Объем часов	Коды компетенций, формированию которых способствует элемент программы
1	2	3	
Раздел 1. Алгоритмизация			ОК 01, ОК 02, ОК 04, ОК 05, ОК 09, ПК 1.1- ПК 1.5, ПК 2.4, 2.5
Тема 1.1. Понятие алгоритма. Представление алгоритмов. Основные алгоритмические конструкции	**Содержание учебного материала**		
	Понятие алгоритма, назначение и основные характеристики алгоритмов. Государственный стандарт Российской Федерации представления блок-схем алгоритмов. Международные стандарты представления алгоритмов. Основные алгоритмические конструкции: линейные алгоритмы, ветвление, циклы.	**2**	
	Практические занятия		
	Представление алгоритмов с помощью ГОСТ 19.701-90 ЕСПД. Схемы алгоритмов, программ, данных и систем. Создание алгоритмов со структурой базовых конструкций	**2**	
Раздел 2. Введение в программирование			ОК 01, ОК 02, ОК 04, ОК 05, ОК 09, ПК 1.1- ПК 1.5, ПК 2.4, 2.5
Тема 2.1. Языки программирования. Основные понятия. Типы данных	**Содержание учебного материала**		
	Развитие языков программирования. Общие понятия о языках программирования. Классификация языков программирования. Стандарты языков программирования. Этапы разработки программы. Компиляторы и интерпретаторы. Среда программирования. Типы ошибок компилятора. Структура программы. Алфавит и словарь языка. Символы. Идентификаторы. Определение констант. Определение простых типов данных. Комментарии. Производные типы данных. Структурированные типы данных.	**2**	
	Практические занятия	**2**	
	Изучение среды разработки. Изучение встроенных средств отладки программы		
Раздел 3. Программирование на алгоритмическом языке С/С++			ОК 01, ОК 02, ОК 04, ОК 05, ОК 09, ПК 1.1- ПК 1.5, ПК 2.4, 2.5
Тема 3.1. Ввод и вывод данных	**Содержание учебного материала**		
	Оператор ввода printf. Оператор вывода scanf. Ввод символьных данных. Вывод символьных данных. Ввод строковых данных. Вывод строковых данных. Консольный ввод данных. Консольный вывод данных.	**2**	

	Практические занятия		
	Вывод числовых значений в различных системах счисления, форматированный вывод данных. Вывод данных в таблицу	**2**	
Тема 3.2. Операции и выражения	**Содержание учебного материала**		
	Операнды. Операция присваивания. Арифметические операции. Логические операции. Операции отношения. Поразрядные операции. Операция условия (?,:). Скобки (), []. Операция sizeof(). Выражения арифметические. Выражения логические. Порядок выполнения арифметических выражений. Порядок выполнения логических выражений.	**2**	
	Практические занятия		
	Вычисление выражений с арифметическими операциями, с операциями отношения и логическими операциями. поразрядными операциями	**2**	
Тема 3.3.Основные алгоритмические конструкции языка C/C++	**Содержание учебного материала**		
	Условный оператор if. Оператор выбора switch. Оператор break. Оператор цикла while. Оператор цикла for. Оператор цикла do…while. Оператор continue. Оператор goto.	**4**	
	Практические занятия	**2**	
	Программирование циклических алгоритмов разветвляющейся структуры		
Тема 3.4. Обработка производных и структурированных типов данных	**Содержание учебного материала**	**6**	
	Массивы. Двумерные массивы. Строки. Стандартные процедуры и функции для работы со строками.	2	
	Структурированный тип данных – множество. Операции над множествами. Комбинированный тип данных – запись.	2	
	Файлы последовательного доступа. Файлы прямого доступа.	2	
	Практические занятия	**14**	
	Обработка одномерных массивов.	2	
	Обработка двумерных массивов.	2	
	Работа со строками.	2	
	Работа с данными типа множество.	2	

	Работа с данными типа запись.	2	
	Типизированные файлы.	2	
	Нетипизированные файлы.	2	
Тема 3.5. Функции в С/С++	**Содержание учебного материала**		
	Объявление функций. Оператор return. Прототипы функций. Параметры функции. Аргументы функции. Формальные параметры. Фактические параметры. Область действия. Область видимости. Классы памяти. Рекурсия.	**4**	
	Практические занятия	**8**	
	Обработка одномерных массивов с использованием пользовательских функций	2	
	Обработка двумерных массивов с использованием пользовательских функций	2	
	Создание программ с использованием рекурсивных функций	2	
	Создание диалоговых программ с использованием функций пользователя	2	
Тема 3.6. Структурное и модульное программирование	**Содержание учебного материала**	**4**	
	Основы структурного программирования. Методы структурного программирования.	2	
	Модульное программирование. Понятие модуля. Структура модуля. Компиляция и компоновка программы. Стандартные модули.	2	
	Практические занятия	**12**	
	Программирование модуля.	6	
	Создание библиотеки подпрограмм.	6	
Тема 3.7. Указатели	**Содержание учебного материала**	**2**	
	Понятие указатель. Описание указателей. Основные понятия и применение динамически распределяемой памяти. Создание и удаление динамических переменных. Структуры данных на основе указателей.	2	
	Практические занятия	**10**	
	Использование структуры стека и очереди при написании программ на С++.	2	
	Использование указателей для организации связанных списков.	4	
	Использование указателей для организации деревьев	4	
Консультации		**2**	
Другая форма контроля		**2**	
Самостоятельная работа обучающихся		**5**	
Раздел 4. Объектно-ориентированное программирование на языке С++			ОК 01, ОК 02, ОК 04, ОК 05,
Тема 4.1 Введение в	**Содержание учебного материала**	**6**	

объектно-ориентированное программирование	История развития ООП. Базовые понятия ООП: объекты и классы, свойства объектов, свойства объектов. Основные принципы ООП: наследование свойств, полиморфизм объектов, инкапсуляция. Этапы ООП.	2	ОК 09, ПК 1.1- ПК 1.5, ПК 2.4, 2.5
	Классы объектов. Компоненты и их свойства.	2	
	Событийно-управляемая модель программирования. Компонентно-ориентированный подход.	2	
	Практические занятия	**6**	
	Классы и объекты: инкапсуляция	2	
	Классы и объекты: наследование	2	
	Классы и объекты: полиморфизм	2	
Тема 4.2. Интегрированная среда разработчика	**Содержание учебного материала**	**6**	ОК 01, ОК 02, ОК 04, ОК 05, ОК 09, ПК 1.1- ПК 1.5, ПК 2.4, 2.5
	Требования к аппаратным и программным средствам интегрированной среды разработчика.	2	
	Интерфейс среды разработчика: характеристика, основные окна, инструменты, объекты. Форма и размещение на ней управляющих элементов.	2	
	Панель компонентов и их свойства. Окно кода проекта. Состав и характеристика проекта. Настройка среды и параметров проекта.	2	
	Практические занятия	**8**	
	Создание проекта с использованием компонентов для работы с текстом.	4	
	Создание проекта с использованием компонентов ввода и отображения чисел, дат и времени.	4	
Тема 4.3. Визуальное событийно-управляемое программирование	**Содержание учебного материала**	**6**	
	Основные компоненты (элементы управления) интегрированной среды разработки, их состав и назначение.	2	
	Дополнительные элементы управления. Свойства компонентов. Виды свойств. Синтаксис определения свойств. Назначения свойств и их влияние на результат. Управление объектом через свойства.	2	
	События компонентов (элементов управления), их сущность и назначение.	2	
	Практические занятия	**4**	
	Создание процедур на основе событий.	2	
	Создание проекта с использованием кнопочных компонентов.	2	
Тема 4.4. Разработка оконного	**Практические занятия**	**12**	
	Разработка функционального интерфейса приложения. Создание интерфейса приложения.	4	

приложения	Разработка функциональной схемы работы приложения.	4	
	Разработка игрового приложения.	4	
Тема 4.5. Этапы разработки	**Содержание учебного материала**	**2**	
	Разработка приложения.		
приложений	**Практические занятия**	**8**	
	Проектирование объектно-ориентированного приложения.	2	
	Создание интерфейса пользователя	2	
	Тестирование, отладка приложения.	4	
Тема 4.6. Иерархия классов	**Содержание учебного материала**	**1**	
	Классы ООП: виды, назначение, свойства, методы, события. Перегрузка методов.		
	Практические занятия	**4**	
	Решение задач		
Промежуточная аттестация (экзамен)		**18**	
Всего:		**172**	

При реализации дисциплины используются следующие интерактивные формы (методы, технологии) обучения:

- лекция пресс-конференции;
- лекция с ошибками;
- лекция-визуализация;
- метод обучения в парах;
- метод обучения в малых группах;
- кейс-метод.

3. УСЛОВИЯ РЕАЛИЗАЦИИ РАБОЧЕЙ ПРОГРАММЫ УЧЕБНОЙ ДИСЦИПЛИНЫ

3.1. Материально-техническое обеспечение дисциплины

Учебный кабинет архитектуры электронно-вычислительных машин и вычислительных систем, включающая автоматизированные рабочие места обучающихся с процессором Intel (R) Core (TM) i3-3220 CPU (3.30 ГГц), оперативной памятью 8 Гб, HDD 500 Гб, программное обеспечение – Linux 7, лазерный принтер Canon LBP6020, интерактивная доска PROMETHEAN, коммутатор D-Link DGS-1024C, посадочные места для обучающихся, маркерная доска, проектор Aser p1220

Учебная аудитория для проведения лекций, практических занятий / семинаров, групповых и индивидуальных консультаций, текущего контроля и промежуточной аттестации, включающая презентационную технику (проектор, экран, компьютер, звуковоспроизводящее оборудование); рабочее место преподавателя; столы, стулья для обучающихся.

Учебная аудитория для самостоятельной работы, включающая автоматизированные рабочие места обучающихся с доступом в Интернет.

3.2. Информационное обеспечение обучения

a) основная литература:

1. Трофимов, В. В. Основы алгоритмизации и программирования : учебник для среднего профессионального образования / В. В. Трофимов, Т. А. Павловская. — 4-е изд. — Москва : Издательство Юрайт, 2024. — 108 с. — (Профессиональное образование). — ISBN 978-5-534-20429-2. — Текст : электронный // Образовательная платформа Юрайт [сайт]. — URL: https://urait.ru/bcode/558137

2. Черпаков, И. В. Основы программирования : учебник и практикум для среднего профессионального образования / И. В. Черпаков. — 2-е изд., перераб. и доп. — Москва : Издательство Юрайт, 2024. — 196 с. — (Профессиональное образование). — ISBN 978-5-534-18760-1. — Текст : электронный // Образовательная платформа Юрайт [сайт]. — URL: https://www.urait.ru/bcode/545507 (дата обращения: 13.05.2024).

b) дополнительная литература:

1. Кувшинов, Д. Р. Основы программирования : учебное пособие для среднего профессионального образования / Д. Р. Кувшинов. — Москва : Издательство Юрайт, 2022. — 105 с. — (Профессиональное образование). — ISBN 978-5-534-07560-1. — Текст : электронный // Образовательная платформа Юрайт [сайт]. — URL: https://www.urait.ru/bcode/493565 (дата обращения: 13.05.2024).

2. Кудрина, Е. В. Основы алгоритмизации и программирования на языке C# : учебное пособие для среднего профессионального образования / Е. В. Кудрина, М. В. Огнева. — Москва : Издательство Юрайт, 2024. — 322 с. — (Профессиональное образование). — ISBN 978-5-534-10772-2. — Текст : электронный // Образовательная платформа Юрайт [сайт]. — URL: https://www.urait.ru/bcode/541725 (дата обращения: 13.05.2024).

3. Кудрявцева, И. А. Программирование: комбинаторная логика : учебное пособие для среднего профессионального образования / И. А. Кудрявцева, М. В. Швецкий. — 2-е изд., перераб. и доп. — Москва : Издательство Юрайт, 2024. — 524 с. —

(Профессиональное образование). — ISBN 978-5-534-15128-2. — Текст : электронный // Образовательная платформа Юрайт [сайт]. — URL: https://www.urait.ru/bcode/542030 (дата обращения: 13.05.2024).

4. Огнева, М. В. Программирование на языке С++: практический курс : учебное пособие для среднего профессионального образования / М. В. Огнева, Е. В. Кудрина, А. А. Казачкова. — 2-е изд., перераб. и доп. — Москва : Издательство Юрайт, 2024. — 342 с. — (Профессиональное образование). — ISBN 978-5-534-18975-9. — Текст : электронный // Образовательная платформа Юрайт [сайт]. — URL: https://www.urait.ru/bcode/555593 (дата обращения: 13.05.2024).

5. Паронджанов, В. Д. Алгоритмические языки и программирование: ДРАКОН : учебное пособие для среднего профессионального образования / В. Д. Паронджанов. — Москва : Издательство Юрайт, 2024. — 436 с. — (Профессиональное образование). — ISBN 978-5-534-14733-9. — Текст : электронный // Образовательная платформа Юрайт [сайт]. — URL: https://www.urait.ru/bcode/543511 (дата обращения: 13.05.2024).

6. Трофимов, В. В. Основы алгоритмизации и программирования : учебник для среднего профессионального образования / В. В. Трофимов, Т. А. Павловская ; под редакцией В. В. Трофимова. — 4-е изд. — Москва : Издательство Юрайт, 2024. — 119 с. — (Профессиональное образование). — ISBN 978-5-534-17498-4. — Текст : электронный // Образовательная платформа Юрайт [сайт]. — URL: https://www.urait.ru/bcode/539994 (дата обращения: 13.05.2024).

7. Федоров, Д. Ю. Программирование на языке высокого уровня Python : учебное пособие для среднего профессионального образования / Д. Ю. Федоров. — 5-е изд., перераб. и доп. — Москва : Издательство Юрайт, 2024. — 227 с. — (Профессиональное образование). — ISBN 978-5-534-17319-2. — Текст : электронный // Образовательная платформа Юрайт [сайт]. — URL: https://www.urait.ru/bcode/539652 (дата обращения: 13.05.2024).

c) ресурсы информационно-телекоммуникационной сети «Интернет», электронные ресурсы (в том числе электронные библиотечные системы):

№	Ссылка на информационный ресурс	Наименование разработки в электронной форме	Доступность
1.	ЭБС «Юрайт» https://urait.ru	ЭБС на платформе «Юрайт». Учебники и учебные пособия издательства «Юрайт» и др.	Индивидуальный неограниченный доступ
2.	ЭБС «Университетская библиотека онлайн» https://biblioclub.ru/	ЭБС на платформе «Университетская библиотека онлайн». Учебники и учебные пособия издательств «Дашков и К°», «Проспект», «Юнити-Дана», и др.	Индивидуальный неограниченный доступ

d) информационные технологии, используемые при осуществлении образовательного процесса по дисциплине (включая перечень программного обеспечения и информационно-справочных систем):

- лицензионное ПО общего назначения;
- специализированное лицензионное ПО.

4. КОНТРОЛЬ И ОЦЕНКА РЕЗУЛЬТАТОВ ОСВОЕНИЯ УЧЕБНОЙ ДИСЦИПЛИНЫ

4.1. Оценивание уровня учебных достижений обучающихся

Оценивание уровня учебных достижений обучающихся по дисциплине осуществляется в виде текущего и промежуточного контроля.

Текущий контроль успеваемости по дисциплине осуществляется в форме (формах):

- выполнение и защита лабораторных работ;
- контрольная работа;
- тестирование.

Отдельно оцениваются личностные качества студента (аккуратность, исполнительность, инициативность).

Знания, умения и навыки обучающихся при текущем контроле определяются оценками «отлично», «хорошо», «удовлетворительно», «неудовлетворительно».

Промежуточный контроль по дисциплине осуществляется в форме экзамена, при этом проводится оценка компетенций, сформированных по дисциплине.

Критерии оценивания результатов обучения по дисциплине:

Знания, умения и навыки обучающихся при промежуточном контроле в форме экзамена определяются оценками «отлично», «хорошо», «удовлетворительно», «неудовлетворительно» (зачтено (отлично), зачтено (хорошо), зачтено (удовлетворительно), не зачтено (неудовлетворительно)).

1. «Отлично» – обучающийся глубоко и прочно усвоил весь программный материал, исчерпывающе, последовательно, грамотно и логически стройно его излагает, не затрудняется с ответом при видоизменении задания, свободно справляется с задачами и практическими заданиями, правильно обосновывает принятые решения, умеет самостоятельно обобщать и излагать материал, не допуская ошибок.

2. «Хорошо» – обучающийся твердо знает программный материал, грамотно и по существу излагает его, не допускает существенных неточностей в ответе на вопрос, может правильно применять теоретические положения и владеет необходимыми умениями и навыками при выполнении практических заданий.

3. «Удовлетворительно» – обучающийся усвоил только основной материал, но не знает отдельных деталей, допускает неточности, недостаточно правильные формулировки, нарушает последовательность в изложении программного материала и испытывает затруднения в выполнении практических заданий.

4. «Неудовлетворительно» – обучающийся не знает значительной части программного материала, допускает существенные ошибки, с большими затруднениями выполняет практические задания, задачи.

Результаты обучения (освоенные умения, усвоенные знания)	**Формы и методы контроля и оценки результатов обучения**
1	**2**
Умения:	
– разрабатывать алгоритмы для конкретных задач; – использовать программы для графического	экспертное наблюдение и оценивание выполнения лабораторных работ;

отображения алгоритмов; – определять сложность работы алгоритмов; – моделировать прикладные задачи; – работать в среде программирования; – реализовывать построенные алгоритмы в виде программ на конкретном языке программирования; – оформлять код программы в соответствии со стандартом кодирования; – выполнять проверку, отладку кода программы.	текущий контроль в форме защиты лабораторных работ; экспертное наблюдение и оценивание выполнения практического экзаменационного задания.
Знания:	
– понятие алгоритмизации, свойства алгоритмов, общие принципы построения алгоритмов, основные алгоритмические конструкции; – эволюцию языков программирования, их классификацию, понятие системы программирования; – основные элементы языка, структуру программы, операторы и операции, управляющие структуры, структуры данных, файлы, классы памяти; – подпрограммы, составление библиотек подпрограмм; – объектно-ориентированную модель программирования, основные принципы объектно-ориентированного программирования на примере алгоритмического языка: понятие классов и объектов, их свойств и методов, инкапсуляции и полиморфизма, наследования и переопределения.	текущий контроль в форме тестирования; устный индивидуальный опрос (в рамках текущего контроля); оценивание выполнения теоретического экзаменационного задания.

4.2. Методические указания для обучающихся по освоению учебной дисциплины. Организация образовательного процесса

Приступая к изучению дисциплины, студентам необходимо ознакомиться с содержанием рабочей программы дисциплины.

Для подготовки к лекционному занятию необходимо скачать и ознакомиться с соответствующей презентацией и подготовить вопросы для обсуждения. После лекции необходимо ознакомиться с рекомендуемыми источниками из списков основной и дополнительной литературы.

Для подготовки к выполнению лабораторных работ необходимо ознакомиться с документацией к средству разработки.

4.3. Фонд оценочных средств

Порядок оценки освоения обучающимися учебного материала определяется содержанием следующих разделов дисциплины:

№ п/п	Наименование раздела дисциплины	Компетенции (части компетенций)	Критерии оценивания	Оценочные средства текущего контроля успеваемости	Шкала оценивания
1.	**Алгоритмизация**	ОК 01, ОК 02, ОК 04, ОК 05, ОК 09, ПК 1.1, ПК 1.2, ПК 1.3, ПК 1.4, ПК 1.5, ПК 2.4, ПК 2.5.	Знать и уметь применять методы и способы решения профессиональных задач в области разработки программных модулей. Уметь интерпретировать полученную информацию для решения стандартных и нестандартных профессиональных задач в области в области разработки программных модулей. Дать определение понятию алгоритма учитывая различные подходы. Перечислить основные свойства алгоритмов. Продемонстрировать информационную и библиографическую культуру применения информационно-коммуникационных технологий с учетом основных требований информационной безопасности. Продемонстрировать способы представления алгоритмов и применения основных алгоритмических конструкций для реализации поставленной профессиональной задачи. Проиллюстрировать применение основные модели алгоритмов при разработке алгоритмы для конкретных задач. Знание методов построения алгоритмов с ориентацией на нотации и программные продукты для графического отображения алгоритмов. Уметь применять методы эффективного взаимодействия с обучающимися, преподавателями и работодателями в процессе обучения и реализации поставленных профессиональных задач. Уметь анализировать и корректировать результаты собственной работы.	Практические занятия	Отлично Хорошо Удовлетворительно Неудовлетворительно

№ п/п	Наименование раздела дисциплины	Компетенции (части компетенций)	Критерии оценивания	Оценочные средства текущего контроля успеваемости	Шкала оценивания
2.	**Введение в программирование**	ОК 01, ОК 02, ОК 04, ОК 05, ОК 09, ПК 1.1, ПК 1.2, ПК 1.3, ПК 1.4, ПК 1.5, ПК 2.4, ПК 2.5.	Перечислить и охарактеризовать этапы решения задачи на компьютере. Владеть актуальной нормативно-правовой базой в области документирования алгоритмов. Продемонстрировать навыки оформления документации на программные средства. Реализовывать поставленные профессиональные задачи в среде программирования. Реализовывать построенные алгоритмы в виде программ на конкретном языке программирования. Выполнять отладку и тестирование программы на уровне модуля. Применять инструментальные средства отладки программного обеспечения. Отслеживать возможности современных и перспективных средств разработки программных продуктов. Уметь применять методы эффективного взаимодействия с обучающимися, преподавателями и работодателями в процессе обучения для достижения наилучших результатов в обучении. Уметь анализировать и корректировать результаты собственной работы, а также рационально распределять свободное время для самообразования в рамках дисциплины. Уметь выявлять требования к ПО. Знать и уметь применять методики сбора, анализа и обработки требований. Знать возможности современных и перспективных средств разработки программных продуктов	Практические занятия	Отлично Хорошо Удовлетворительно Неудовлетворительно
3.	**Программирование на алгоритмическом языке С/С++**	ОК 01, ОК 02, ОК 04, ОК 05, ОК 09, ПК 1.1, ПК 1.2, ПК 1.3, ПК 1.4, ПК 1.5, ПК 2.4, ПК 2.5.	Перечислить и охарактеризовать этапы решения задачи на компьютере. Перечислить принципы структурного и модульного программирования. Владеть актуальной нормативно-правовой базой в области документирования алгоритмов. Уметь формировать алгоритмы разработки программных модулей в соответствии с техническим заданием.	Практические занятия Тест	Отлично Хорошо Удовлетворительно Неудовлетворительно

№ п/п	Наименование раздела дисциплины	Компетенции (части компетенций)	Критерии оценивания	Оценочные средства текущего контроля успеваемости	Шкала оценивания
			Продемонстрировать навыки оформления документации на программные средства. Реализовывать поставленные профессиональные задачи в среде программирования. Реализовывать построенные алгоритмы в виде программ на конкретном языке программирования. Выполнять отладку и тестирование программы на уровне модуля. Применять инструментальные средства отладки программного обеспечения. Отслеживать возможности современных и перспективных средств разработки программных продуктов. Уметь применять методы эффективного взаимодействия с обучающимися, преподавателями и работодателями в процессе обучения для достижения наилучших результатов в обучении. Уметь анализировать и корректировать результаты собственной работы, а также рационально распределять свободное время для самообразования в рамках дисциплины. Уметь выявлять требования к ПО. Знать и уметь применять методики сбора, анализа и обработки требований. Знать возможности современных и перспективных сред		
4.	**Объектно-ориентированное программирование на языке С++**	ОК 01, ОК 02, ОК 04, ОК 05, ОК 09, ПК 1.1, ПК 1.2, ПК 1.3, ПК 1.4, ПК 1.5, ПК 2.4, ПК 2.5.	Уметь интерпретировать полученную информацию для выполнения поставленной задачи. Знать основные парадигмы в объектно-ориентированном программировании. Дать определение понятию класс. Дать определение понятию инкапсуляции. Уметь работать с объектами. Выполнять включение классов. Дать определение понятию наследование. Дать определение понятию полиморфизм. Уметь описывать и использовать полиморфные иерархии классов. Знать механизмы раннего и позднего связывания. Уметь работать с шаблонами классов.	Практические занятия	Отлично Хорошо Удовлетворительно Неудовлетворительно

№ п/п	Наименование раздела дисциплины	Компетенции (части компетенций)	Критерии оценивания	Оценочные средства текущего контроля успеваемости	Шкала оценивания
			Знать механизм обработки исключительных ситуаций. Уметь использовать механизм обработки исключительных ситуаций Знать правила и методы преобразования типов данных. Знать иерархию потоковых классов. Уметь использовать файловые, строковые и стандартные потоки и выполнять форматирование данных в них. Знать типы итераторов и правила их описания. Знать типы стандартных контейнерных классов. Уметь использовать алгоритмы стандартной библиотеки C++. Уметь разрабатывать элементы приложения в интегрированной среде программирования. Использовать различные источники для выполнения поставленной задачи, включая электронные. Быть готовым решать стандартные и нестандартные профессиональных задач в области в области разработки программных модулей.		
Итого:		ОК 01, ОК 02, ОК 04, ОК 05, ОК 09, ПК 1.1, ПК 1.2, ПК 1.3, ПК 1.4, ПК 1.5, ПК 2.4, ПК 2.5.	**Форма контроля**	**Оценочные средства промежуточной аттестации**	**Шкала оценивания**
			Экзамен	Устно-практический экзамен – перечень вопросов и типовых заданий	Отлично Хорошо Удовлетворительно Неудовлетворительно

ТИПОВЫЕ КОНТРОЛЬНЫЕ ЗАДАНИЯ ИЛИ ИНЫЕ МАТЕРИАЛЫ, НЕОБХОДИМЫЕ ДЛЯ ОЦЕНКИ ЗНАНИЙ, УМЕНИЙ, НАВЫКОВ И (ИЛИ) ОПЫТА ДЕЯТЕЛЬНОСТИ, ХАРАКТЕРИЗУЮЩИХ ЭТАПЫ ФОРМИРОВАНИЯ КОМПЕТЕНЦИЙ В ПРОЦЕССЕ ОСВОЕНИЯ ОПОП СПО

Контроль успеваемости по дисциплине осуществляется с помощью следующих оценочных средств:

ПРАКТИЧЕСКИЕ ЗАНЯТИЯ

№ п/п	Номер раздела дисциплины	Наименование лабораторной работы	Трудоемкость, часов
1	1	Представление алгоритмов с помощью ГОСТ	2

		19.701-90 ЕСПД. Схемы алгоритмов, программ, данных и систем. Создание алгоритмов со структурой базовых конструкций	
2	2	Изучение среды разработки. Изучение встроенных средств отладки программы	2
3	3	Вывод числовых значений в различных системах счисления, форматированный вывод данных. Вывод данных в таблицу	2
4	3	Вычисление выражений с арифметическими операциями, с операциями отношения и логическими операциями. поразрядными операциями	2
5	3	Программирование циклических алгоритмов разветвляющейся структуры	2
6	3	Обработка одномерных массивов.	2
7	3	Обработка двумерных массивов.	2
8	3	Работа со строками.	2
9	3	Работа с данными типа множество	2
10	3	Работа с данными типа запись.	2
11	3	Типизированные файлы.	2
12	3	Нетипизированные файлы.	2
13	3	Обработка одномерных массивов с использованием пользовательских функций	2
14	3	Обработка двумерных массивов с использованием пользовательских функций	2
15	3	Создание программ с использованием рекурсивных функций	2
16	3	Создание диалоговых программ с использованием функций пользователя	2
17	3	Программирование модуля.	6
18	3	Создание библиотеки подпрограмм.	6
19	3	Использование структуры стека и очереди при написании программ на выбранном ЯП.	2
20	3	Использование указателей для организации связанных списков.	4
21	3	Использование указателей для организации деревьев	4
22	4	Классы и объекты: инкапсуляция	2
23	4	Классы и объекты: наследование	2
24	4	Классы и объекты: полиморфизм	2
25	4	Создание проекта с использованием компонентов для работы с текстом.	4
26	4	Создание проекта с использованием компонентов ввода и отображения чисел, дат и времени.	4
27	4	Создание процедур на основе событий.	2
28	4	Создание проекта с использованием кнопочных компонентов.	2
29	4	Разработка функционального интерфейса приложения. Создание интерфейса	4

		приложения.	
30	4	Разработка функциональной схемы работы приложения.	4
31	4	Разработка игрового приложения.	4
32	4	Проектирование объектно-ориентированного приложения.	2
33	4	Создание интерфейса пользователя	2
34	4	Тестирование, отладка приложения.	4
35	4	Решение задач	4
		Итого:	96

Пример практического занятия

Практическое занятие № 1

Тема: Представление алгоритмов с помощью ГОСТ 19.701-90 ЕСПД. Схемы алгоритмов, программ, данных и систем

Цель: закрепить теоретические знания и приобрести практические навыки по разработке алгоритмов по обработке файлов с учетом разработанных функциональных требований к программе, содержащей функции.

Практическое задание:
Разработать алгоритм, согласно варианту задания и ГОСТ 19.701-90.

Варианты заданий:

1	X(20)	Сумму элементов массива, расположенных до последнего положительного по модулю элемента.
2	X(20)	Произведение элементов массива, расположенных между максимальным и минимальным элементами.
3	X(20)	Произведение элементов массива, расположенных между максимальным по модулю и минимальным по модулю элементами.
4	X(20)	Сумму элементов массива, расположенных между первым и последним нулевыми элементами.
5	X(20)	Сумму элементов массива, расположенных между первым и последним отрицательными элементами.

Технология выполнения работы:

Формализовать задачу согласно варианту задания. Определить входные и выходные данных. Выделить базовые алгоритмы, составляющие решение задачи.

Шкала оценивания и критерии оценки:

Оценка	Минимальное количество баллов	Максимальное количество баллов	Критерий

«5» (отлично)	12	13	выполнены все задания практической работы, обучающийся четко и без ошибок ответил на все контрольные вопросы
«4» (хорошо)	10	11	выполнены все задания практической; обучающийся ответил на все контрольные вопросы с замечаниями
«3» (удовлетворительно)	7	9	выполнены все задания практической работы с замечаниями; обучающийся ответил на все контрольные вопросы с замечаниями
«2» (неудовлетворительно)	0	6	обучающийся не выполнил или выполнил неправильно задания практической работы; обучающийся ответил на контрольные вопросы с ошибками или не ответил на контрольные вопросы

ТЕСТ

Тест является формой текущего контроля и содержит теоретические или (и) практические задания. Тестирование проводится во внеаудиторное время, либо в заранее установленные часы преподавателем. При проведении тестирования тест выполняется индивидуально, в письменной форме на бланке, выданном преподавателем. Бланки содержат вопросы теста с вариантами ответов. Бланки должны удовлетворять следующим требованиям: в работе указывается ФИО студента, номер группы и выделенный ответ (ы). На выполнение отводится 2 академических часа.

Примеры тестовых заданий

Внимание! Вопросы, связанные с ЯП, могут быть заменены на аналогичные, но для других ЯП

Тест по разделу 1, 2 и 3

Вопрос 1.1.1. Выберите правильное определение для термина алгоритм

1. Предписание относительно последовательности действий, преобразующих исходные данные в искомый результат.
2. Описание какого-либо класса явлений внешнего мира, выраженное с помощью математической символики, геометрических фигур.
3. Множество графических символов, которые заменяют объекты программы, с введенным в нем отношением эквивалентности между символами.
4. Модель программы, в которой такие понятия, как оператор, операнд, переменная, выполнение и т.д., являются обобщением соответствующих понятий существующих языков программирования, представленные в ином виде

Вопрос 1.1.2. Укажите, в каком виде можно представить алгоритм

1. Табличный
2. Формульный
3. Программный
4. Псевдокод
5. Структурограммы
6. Графический
7. Естественный

Вопрос 1.1.3. Из приведенных свойств выберите свойства алгоритмов

8. Дискретность
9. Массовость
10. Детерминированность
11. Линейность
12. Вероятность
13. Читаемость
14. Краткость

Вопрос 1.2.1. Линейный алгоритм это

1. Набор указаний, выполняемых последовательно во времени.
2. Алгоритм, который дает программу решения задачи несколькими путями или способами, приводящими к вероятному достижению результата
3. Такой алгоритм, в котором достижение конечного результата программы действий однозначно не предопределено, так же как не обозначена вся последовательность действий.
4. Алгоритм, который строится из отдельных шагов (действий, операций, команд), при этом множество шагов, из которых составлен алгоритм, конечно.
5. Алгоритм, содержащий хотя бы одно условие, в результате проверки которого исполнитель обеспечивает переход на один из двух возможных шагов.
6. Алгоритм, предусматривающий многократное повторение одного и того же действия (одних и тех же операций) над новыми исходными данными.

Вопрос 1.2.2. Разветвляющий алгоритм это

1. Набор указаний, выполняемых последовательно во времени.
2. Алгоритм, который дает программу решения задачи несколькими путями или способами, приводящими к вероятному достижению результата
3. Такой алгоритм, в котором достижение конечного результата программы действий однозначно не предопределено, так же как не обозначена вся последовательность действий.
4. Алгоритм, который строится из отдельных шагов (действий, операций, команд), при этом множество шагов, из которых составлен алгоритм, конечно.
5. Алгоритм, содержащий хотя бы одно условие, в результате проверки которого исполнитель обеспечивает переход на один из двух возможных шагов.
6. Алгоритм, предусматривающий многократное повторение одного и того же действия (одних и тех же операций) над новыми исходными данными.

Вопрос 1.2.3. Циклический алгоритм это

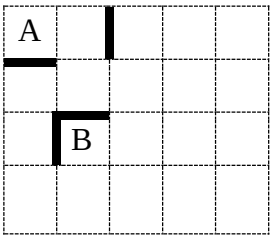
1. Набор указаний, выполняемых последовательно во времени.
2. Алгоритм, который дает программу решения задачи несколькими путями или способами, приводящими к вероятному достижению результата
3. Такой алгоритм, в котором достижение конечного результата программы действий однозначно не предопределено, так же как не обозначена вся последовательность действий.
4. Алгоритм, который строится из отдельных шагов (действий, операций, команд), при этом множество шагов, из которых составлен алгоритм, конечно.

5. Алгоритм, содержащий хотя бы одно условие, в результате проверки которого исполнитель обеспечивает переход на один из двух возможных шагов.
6. Алгоритм, предусматривающий многократное повторение одного и того же действия (одних и тех же операций) над новыми исходными данными.

Вопрос 1.3.1. Некий исполнитель умеет выполнять три команды:
FD<число шагов> – движение вперед на указанное число шагов
RT<число градусов> – поворот направо на указанное число градусов
REPEAT<число повторений>[<повторяющиеся действия>] – команда повторения
Например, REPEAT 4[FD 20 RT 90] строит квадрат со стороной 20. Какую фигуру будет представлять собой траектория движения данного исполнителя в результате выполнения команды
REPEAT 8 [FD 60 RT 45]
1) Равносторонний треугольник
2) Ромб
3) Правильный шестиугольник
4) Правильный восьмиугольник

Вопрос 1.3.2. Исполнитель Робот действует на клетчатом поле, между соседними клетками которого могут стоять стены. Робот передвигается по клеткам поля и может выполнять следующие команды:
Вверх (1), Вниз (2), Вправо (3), Влево (4).
При выполнении каждой такой команды Робот перемещается в соседнюю клетку в указанном направлении. Если же в этом направлении между клетками стоит стена, то робот разрушается.
Какую последовательность из 5 команд выполнил Робот, чтобы переместиться из клетки A в клетку B, не разрушившись от встречи со стенами? Ответы записаны в виде последовательности цифр, соответствующих командам.

1) 32323 2) 23324 3) 32324 4) 22211



Вопрос 1.3.3. Исполнитель Черепашка перемещается на экране компьютера, оставляя след в виде линии. В каждый конкретный момент известно положение исполнителя и направление его движения. У исполнителя существуют две команды:
Вперед n, где n – целое число, вызывающая передвижение черепашки на n шагов в направлении движения.
Направо m, где m – целое число, вызывающая изменение направления движения на m градусов по часовой стрелке.
Запись Повтори 5 [Команда1 Команда2] означает, что последовательность команд в скобках повторится 5 раз.
Черепашке был дан для исполнения следующий алгоритм:
Повтори 5 [Вперед 10 Направо 72]

Какая фигура появится на экране?
1) Незамкнутая ломаная линия
2) Правильный треугольник
3) Квадрат
4) Правильный пятиугольник

Вопрос 2.1.1. Редактор связей это

1. Программа, которая собирает части программы и подключает стандартные функции
2. Программа, которая разбивает код программы на логически-связанные части
3. Программа, которая распространяет код по заданным адресам
4. Программа, которая вводит в код программы новые компоненты.

Вопрос 2.1.2. Укажите правильное определение термину компилятор

1. Программа, которая принимает на вход один или несколько объектных модулей и собирает по ним исполнимый модуль.
2. Программа, которая транслирует исходного кода компьютерной программы или отдельного программного модуля, составленных на языке программирования (исходная программа, исходный модуль) в программу или модуль на машинном языке или языке, близком к машинному
3. Преобразование программы, представленной на одном из языков программирования, в программу на другом языке и, в определённом смысле, равносильную первой.
4. Языковой процессор, который построчно анализирует исходную программу и одновременно выполняет предписанные действия

Вопрос 2.1.3. Транслятор это

1. Распознавание кода программы
2. Распространение кода по заданным адресам
3. Перевод текста программы в машинный код
4. Введение в код программы новых компонентов.

Вопрос 2.2.1. Отладка это

1. Процесс локализации и устранения ошибок
2. Процесс пошагового выполнения программы
3. Процесс вывода сигнальных сообщений в программе
4. Процесс исследования программы с целью получения информации о её качестве

Вопрос 2.2.2. Тестирование это

1. Процесс исследования программы с целью получения информации о её качестве
2. Процесс локализации и устранения ошибок
3. Процесс пошагового выполнения программы
4. Процесс вывода сигнальных сообщений в программе

Вопрос 2.2.3. Трассировка это

1. Процесс пошагового выполнения программы
2. Процесс выполнение отдельного кода программы
3. Процесс вывода сигнальных сообщений в программе

4. Процесс, который выводит блок схему программы

Вопрос 2.3.1. Укажите правильные имена переменных для языка C/C++

1. _gets
2. 1_gets
3. gets
4. gets!

Вопрос 2.3.2. Переменная это

1. Область в памяти компьютера, которая имеет имя и хранит некоторое значение
2. Область памяти на диске которая имеет имя и расширение
3. Область системы, которая изменяется в зависимости от условий
4. Значение возникающие при запуске программы

Вопрос 2.3.3. Укажите как определяются комментарии в языке C/C++

1. / ... /
2. /* … */
3. {…}
4. \\...

Вопрос 2.4.1. Операторные скобки это

1. Часть кода, которая сгруппирована и воспринимается как единое целое.
2. Скобки, в которых указываются аргументы операторов или функций.
3. Скобки, в которые заключаются операнды
4. Скобки, которые позволяют комментировать используемые в программе операторы

Вопрос 2.4.2. Что является минимальной единицей в структуре программы на языке C/C++

1. Функция
2. Переменная
3. Поле структуры
4. Индекс массива

Вопрос 2.4.3. Что означает ключевое слово void в языке C/C++

1. Отсутствие параметров в функции
2. Безразмерная величина параметра функции
3. Составляющая имени основной функции в программе
4. Особый тип данных

Вопрос 3.1.1. Как переименовать тип в языке C/C++

1. Команда typedef
2. Команда rename
3. Этого сделать не возможно
4. С помощью директивы #define

Вопрос 3.1.2. Что храниться в переменной типа char

1. Один символ
2. Код символа
3. Целое число типа int

4. Литера

Вопрос 3.1.3. Соотнесите правильно спецификаторы типов данных и типы данных (ответ дайте в виде 1-а, 2-б и т.д.)

1. %u а) вещественный тип double
2. %g б) беззнаковое десятичное число
3. %Lf в) десятичное число
4. %Ld г) целочисленное значение

Вопрос 3.2.1. Сколько унарных операций в языке C/C++

1. 8
2. 5
3. 4
4. 6

Вопрос 3.2.2. Расположите в порядке убывания по приоритету представленные операции (например, 42513)

1. !
2. >
3. /
4. %
5. &&

Вопрос 3.2.3. Какие операции относятся к логическим операциям

1. >=
2. &
3. NOT
4. ~
5. ||
6. >>

Вопрос 3.3.1. Определите значение целочисленных переменных a и b после выполнения фрагмента программы:

```
a := 3 + 8*4;
b := (a / 10) + 14;
a := (b % 10) + 2;
```

1. a = 0, b = 18 2. a = 11, b = 19 3. a = 10, b = 18 4. a = 9, b = 17

Вопрос 3.3.2. Определите значение целочисленных переменных a и b после выполнения фрагмента программы:

```
a := 1819;
b := (a % 100)*10+9;
a := (10*b–a) / 100;
```

1. a = 81, b = 199 2. a = 81, b = 189 3. a = 71, b = 199 4. a = 71, b = 18

Вопрос 3.3.3. Укажите, какое логическое выражение равносильно выражению !(A || !B || C) ?

1. $\neg A \vee B \vee \neg C$ 2. $A \wedge \neg B \wedge C$ 3. $\neg A \vee \neg B \vee \neg C$ 4. $\neg A \wedge B \wedge \neg C$

Вопрос 4.1.1. Укажите все функции вывода в языке C/C++

1. print()
2. write()
3. puts()
4. putch()

Вопрос 4.1.2. Укажите верное использование функции вывода printf()

1. printf(“выражение = %-5d”,3+5)
2. printf(“Введите %d – й элемент /n”, &mas[i])
3. printf(“Значение выражения = %9.6Lf” x)
4. printf(“y=%d^%d+(%f+%f)/%f=%f”,y,a,b,cos(a),sin(b),3.14,y)
5. printf(“%s”,’Вывод строки')
6. printf(“%c%c%c%c”,’Ф’,’С’,’П’,’О’)

Вопрос 4.1.3. Сколько строк сформируется используя указанную функцию puts(“Строка символов\n\n выводящая\t\t\t в несколько\\n строк”)

1. 1
2. 2
3. 3
4. 4
5. 5

Вопрос 4.2.1. Укажите все функции ввода в языке C/C++

1. cin
2. input()
3. getche()
4. readln()
5. getstr()
6. scanf()

Вопрос 4.2.2. Укажите верное использование функции ввода scanf()

1. scanf(”%d,%d”,&a,&b)
2. scanf(”%d*%d”,&i,&j)
3. scanf(”%5s”,&s)
4. scanf(”Vvedi a >>%d”,&a)
5. scanf(”%c”,ch)

Вопрос 4.2.3. Что означает содержимое управляющей строки, указанной в функции scanf(“%[A-Z^1-5]c”,&ch)

1. В переменную ch можно вводить символы из указанных диапазонов
2. В переменную ch можно вводить символы из указанного диапазона, при этом, ширина поля отводимого под символ не более 5 позиций
3. Ничего, потому что в функции scanf() всё символы, отличные от спецификаторов типа игнорируются
4. В переменную ch можно вводить символы из первого диапазона, а из второго диапазона символы вводится не будут

Вопрос 5.1.1. Что появиться на экране после выполнения фрагмента программы (switch)

```
n=2;
switch(n)
{
case 1: printf(“45 ”); break;
case 2: printf(“87 ”);
case 3: printf(“34 ”);break;
default: printf(“net chisla”);
}
```

1. 45
2. 34 net chisla
3. 87 34
4. 8734 net chisla

Вопрос 5.1.2. Какой результат выдаст функция printf()

```
c=5;
if(c==5)
{
c++;
if(c==6) printf(“1”);
else printf(“2”);
}
else      {
c--;
if(c==4) printf(“4”);
}
```

1. 1
2. 2
3. 3
4. 4

Вопрос 5.1.3. Чему будет равно значение переменной S после выполнения фрагмента программы

```
int s=5;
s=s/5;
if(s==2.5)
{ s*=2;
If(s==5) {print(“%d”,s);}
else s=4;
}
Else
if(s==2){ s+=1;}
else s*=3;
```

1. 4
2. 5
3. 3

4. 6

Вопрос 5.2.1. Укажите иную интерпретацию представленного фрагмента циклического алгоритма

```
for(i=0,j=5;i<=5,j>=0;i++,j--)
{
printf(“Цикл повторяется i= %d j= %d”,i,j);
}
```

1.
```
for(i=0;i<=5;i++)

{
    for(j=5;j>=0;j--) Printf(“Цикл повторяется i= %d j= %d \n”,i,j);
}
```

2.
```
int i=0;

int j=5;
do
{
Printf(“Цикл повторяется i= %d j= %d \n”,i,j);
i++;
j--;
} while{(i<=5)&&(j>=0)}
```

3.
```
int i=0;

int j=5;
while((i<5)&&(j>0))
{
printf(“Цикл повторяется i= %d j= %d \n”,i,j);
i++;
j--;
}
```

4.
```
int i=0;

int j=5;
do
{
Printf(“Цикл повторяется i= %d j= %d \n”,i,j);
i++;
j--;
} while{(i<=5)||(j>=0)}
```

Вопрос 5.2.2. Укажите циклические конструкции в языке C/C++

1. case
2. switch
3. do while
4. repeat

Вопрос 5.2.3. Что такое итерация цикла

1. Лишнее прохождение цикла
2. Часть цикла, которая выходит за рамки программы
3. Однократное прохождение тела цикла
4. Полное прохождение цикла.

Вопрос 5.3.1. Сколько раз выполниться итерация цикла:

```
int j=0;
for(int i=0;i<=10;i++)
{ i++; j++;
}
```

1. 10
2. 4
3. 5
4. Не сработает

Вопрос 5.3.2. Выберите цикл с бесконечным числом итераций:

1. while(0){}
2. int a=5; for(i=-500;i<=0;i++){}
3. for(;;){}
4. do{} while(0);

Вопрос 5.3.3. Где будет объявлена переменная i?

```
int main()
{
        for(int i=0;i>=-5;i--)
        {…}
}
```

1. Во всей программе
2. В функции main
3. В цикле for
4. Так переменные объявлять нельзя

Вопрос 6.1.1. Сколько размерностей максимально может быть у массива в языке C/C++

1. Одна размерность
2. Две размерности
3. Три размерности
4. Четыре размерности
5. Все варианты подходят

Вопрос 6.1.2. Как обращаются к элементу массива

1. По его индексу
2. По его имени
3. Указав имя массива и через точку поле элемента массива
4. По адресу элемента

Вопрос 6.1.3. Как определить, что элемент находится над главной диагональю

1. Номер строки элемента равен номеру столбца
2. Номер строки элемента больше номера столбца
3. Номер строки элемента меньше номера столбца
4. Сумма номера строки и номера столбца элемента равна количеству строк в матрице

Вопрос 6.2.1. Значения двух массивов A[100] и B[100] задаются с помощью следующего фрагмента программы:

```
for(n=0;n<100;n++)
 A[n]= n - 10;
for(n=0;n<100;n++)
 B[n]= A[n]*n
```

Сколько элементов массива B будут иметь положительные значения?

1. 10 2. 50 3. 90 4. 100

Вопрос 6.2.2. Некоторые значения двумерного массива задаются с помощью вложенного оператора цикла в представленном фрагменте программы:

```
for(n=1;n<=5;n++)
  for(k=1;k<=5;k++)
    B[n,k]= n + k;
```

Чему будет равно значение B(2,4)?

1. 9 2. 8 3. 7 4. 6

Вопрос 6.2.3. Каким из описанных способов можно заполнить массив temp целочисленными значениями переменных?

1. for (index=0; index<n; index++) scanf(“%d”,&temp[index]);
2. int temp[5]={45,56,12,98,12};
3. int temp[5]={-5,8,10};

Вопрос 6.3.1. Значения двух массивов a[100] и b[100] задаются с помощью следующего фрагмента программы:

```
for(n=0;n<100;n++)
a[n]=(n-80)*(n-80);
for(n=0;n<100;n++)
b[101-n]=a[n]
```

Какой элемент массива b будет наибольшим?

1. B[0] 2.b[20] 3.b[79] 4.b[99]

Вопрос 6.3.2. Значения элементов двух массивов A и B размером 100 задаются с помощью следующего фрагмента программы:

```
for(i=0;i<100;i++)
  A[i]= 50 – i;
for(i=1;i<100;i++)
  B[i]= A[i] + 49;
```

Сколько элементов массива B будут иметь отрицательные значения?

1. 0 2. 10 3. 50 4. 100

Вопрос 6.3.3. Какой из предложенных способов применяют для инициализации массива символов?

1. char sim[6]={‘П’,’р’,’и’,’в’,’е’,’т’};
2. char sim[6]=”Привет”;
3. char sim[]=”Привет\n”;
4. char sim[10]={”Привет”}.

Вопрос 6.4.1. Значения элементов двумерного массива A были равны 0. Затем значения некоторых элементов были изменены:

```
n=0;
for(i=1;i<=5;i++)
  for(j=1;j<=6-i;j++)
  {
    n= n + 1:
    A[i,j]= n;
  }
```

Какой элемент массива будет иметь в результате максимальное значение?

1. A[1,1] 2. A[1,5] 3. A[5,1] 4. A[5,5]

Вопрос 6.4.2. Значения элементов двумерного массива A размером 5x5 задаются с помощью вложенного цикла в представленном фрагменте программы:

```
for(i=0;i<5;i++)
  for(j=0;j<5;j++)
   {
    A[i,j]= i*j;
   }
```

Сколько элементов массива будут иметь значения больше 10?

1. 12 2. 8 3. 10 4. 4

Вопрос 6.4.3. Значения элементов двумерного массива A размером 5x5 задаются с помощью вложенного цикла в представленном фрагменте программы:

```
for(i=0;i<5;i++)
  for(j=0;j<5;j++)
   {
    A[i,j]= i + j;
   }
```

Сколько элементов массива будут иметь значения больше 5?

1. 5 2. 20 3. 10 4. 15

Вопрос 7.1.1. Структурным типом данных называется:

1. Тип данных, который может включать в себя несколько полей – элементов разных типов (в том числе и другие структуры).
2. Тип данных, который хранит информацию о переменных объявленных во всей программе
3. Тип данных, который помогает упорядочить программу с примером шаблона.

4. Нет такого стандартного типа в С

Вопрос 7.1.2. Выберите правильно объявленную структуру данных:

1. struct Book {

```
char author[40];
char title[80];
int year;
int pages;
};
```

2. structura Book (

```
char author[40];
char title[80];
int year;
int pages;
);
```

3. Book (char author[40]; char title[80]; int year; int pages;):struct;
4. structura (char author[40]; char title[80]; int year; int pages;):Book;

Вопрос 7.1.3. Выберете правильный формат обращения к полю структуры:

1. имя_структуры.название_поля
2. имя_структуры(название_поля)
3. имя_структуры{название_поля}
4. название_поля-имя_структуры

Вопрос 8.1.1. Функция это

1. Выделенная последовательность инструкций, предназначена для решения определенной задачи
2. Независимая область программы, которая вычисляет заданный результат
3. Код программы, который ограничен операторными скобками.
4. Зависимая переменная величина

Вопрос 8.1.2. Формальный параметр

1. Параметр, который используется только в прототипе функции
2. Параметр, который используется в описании функции
3. Параметр, который меняет своё значение в теле функции
4. Параметр, который используется при вызове функции

Вопрос 8.1.3. Фактический параметр

1. Параметр, который используется в прототипе функции
2. Параметр, который используется в описании функции
3. Параметр, который меняет своё значение в теле функции
4. Параметр, который используется при вызове функции

Вопрос 8.2.1. Из представленных прототипов функций какие из вызовов функций будут правильными:

```
double calc( double b, int c );
int count(char v[10]);
void sum(int primer);
```

1. calc(123,45,3245);
2. count(“Slovo nomer odin”)
3. sum(“3+4 семь”)
4. Ни одно из вышеперечисленных.

Вопрос 8.2.2. Значения переменных a и b до вызова функции равны 5 и 10 соответственно. Чему они равны будут после выполнения функции

```
void func(int a, int b)
{
int buf;
buf=a;
a=b;
b=buf;
}
```

1. a=5, b=10. 2. a=10, b=5. 3. a=5.0, b=10.0 4. a=10.0, b=5.0.

Вопрос 8.2.3. Какие прототипы функций с параметром массив является верным:

1. func(int mas[]);
2. void func(float mas[10])
3. int func(int mas[][m], int n, int m)
4. void func(int mas[][],int str, int st)
5. Ни одно из вышеперечисленных.

Вопрос 8.3.1. Область действий это

1. Часть кода, в которой есть возможность использования той или иной функции.
2. Это правила, которые определяют, известен ли фрагменту кода другой фрагмент кода или данных, или имеет ли он доступ к этому другому фрагменту.
3. Определенный фрагмент кода, который передается в другую программу.
4. В Си нет такого понятия т.к. все переменные и функции доступны в любом месте программе вне зависимости от условий.

Вопрос 8.3.2. Область видимости это

1. Часть кода, в которой есть возможность использования той или иной функции.
2. Это правила, которые определяют, где переменная может применяться.
3. Видимый функции фрагмент кода, который передается в основную программу.
4. В Си нет такого понятия т.к. все объекты видимы в любом месте программе вне зависимости от условий.

Вопрос 8.3.3. Статическая переменная это

1. Это такая переменная, которая при первом вызове функции и сохраняют свое значение даже после выхода из функции. В следующий раз при новом вызове функции статическая переменная будет иметь то же значение

2. Это такая переменная, которая объявлена в функции и доступна как глобальная переменная
3. Это такая переменная, которая меняет свое значение в зависимости от прохождения программы.
4. Это такая переменная, которая иметь одно постоянное значение.

Вопрос 8.4.1. Что возвращает функция

```
int kvadur(float a, float b, float c, float *x1, float *x2)
{
float d;
if (a==0) return(-1);

d=b*b-4*a*c;
if(d<0)
return(0);

*x1=(-b+sqrt(d))/(2*a);
*x2=(-b-sqrt(d))/(2*a);

if (*x1!=*x2) return(2);
   else return(1);
}
```

1. Количество корней уравнения
2. Значение корней уравнения
3. Количество и значения корней уравнения

Вопрос 8.4.2. Что возвращает функция

```
float vcil(float h,float r)
{ return(Pi*r*r*h); }
```

1. Значение выражения.
2. Выражение
3. Произведение указателей

Вопрос 8.4.3. Выберите правильный вариант вызова функции для переменной z короткого целого типа.

```
int GetSqr(int k)
{
int x=k*k;
return(x);
}
```

1. GetSqr(y);
2. z=(short int)GetSqr(y);
3. z=GetSqr(y)
4. short int z= GetSqr(y)
5. Ничего из вышеперечисленного

Вопрос 9.1.1. Что такое файловая структура?

1. Способ организации файлов в памяти машины;
2. Связующее звено между программой и операционной системой, позволяющее вводить-выводить данные в файл и из файла, а также печатать информацию на принтере.
3. Это заголовочный файл stdio.h.

Вопрос 9.1.2. Почему необходимо закрыть файл перед завершением режима работы программы с файлом?
1. Закрывая файл - сообщаем системе о завершении обмена между оперативной памятью и внешним устройством.
2. При закрытии файла получаем гарантию, что вся информация, имевшаяся в буфере, действительно записана в файл, кроме того, будет записан символ конца файла, впоследствии обеспечивающий доступ к файлу.
3. Это требует синтаксис.

Вопрос 9.1.3. Функция fwrite() обеспечивает:
1. Построчную запись данных в файл или вывод их на принтер.
2. Форматированный вывод символов, строк или чисел на диск или на принтер.
3. Запись целой структуры.
4. Посимвольную запись данных в файл или вывод их на принтер.
Вопрос 9.2.1. Что выполняет фрагмент программы

```
 …
int letter;
….
while((letter=fgetc(fp1))!=EOF)
{
  putchar(letter);
  fputc(letter,fp2);
}
…
```

1. Копирует содержимое одного файла в другой
2. Читает данные из файла 2
3. Отображает на экран копируемый файл
4. Определяет количество записанных символов

Вопрос 9.2.2. Сколько строк в файл запишется после выполнения фрагмента программы

```
…
char s1[]=”Hello”, s2[]=” world!!!”;
…..
for(i=1;i<=10;i++)
{
fputs(s1,fp);
fputs(s2,fp);
i++,j++,i++;
}
…..
```

1. 1

2. 2
3. 3
4. 4

Вопрос 9.2.3. Что выполняет представленный фрагмент программы?

```
…
len=fread(&(mas[i]),sizeof(int),1,fp);
while (len !=0 && i<7)
{
  i++;
  len=fread(&(mas[i]),sizeof(int),1,fp);
}
```

1. Поэлементно считывается массив из файла.
2. Считывание данных из файла в массив целого типа
3. Отслеживание выхода за границы массива
4. Считывание компоненты - структуры из файла в массив, хранящийся в оперативной памяти
5. Вычисление длины массива
6. Вычисление фактической длины файла

Вопрос 9.3.1. Запись какого количества данных и в какую область памяти компьютера обеспечивает представленный фрагмент программы

```
…
printf(“Введите имя: ”);
while (strlen(gets(name))>0)
{
  fputs(name,fp);
  fputs(“\n”,fp);
  printf(“Введите имя: ”);
}
```

1. Произвольное количество строк в файл
2. Имена идентификаторов в динамический файл
3. Две строки
4. Ничего не записывается
5. Нет правильного ответа

Вопрос 9.3.2. Какой тип данных обрабатывает фрагмент программы

```
….
printf(“Введите наименование товара: ”);
gets(name);
while (strlen(name)>0)
{
printf(“Введите цену товара ”);
scanf(“%f”,&cost);
puts(“Введите количество”);
scanf(“%d”,&kol);
```

fprintf(fp,”%s %f %d\n”,name,cost,kol);
…
}

1. Структура.
2. Числовые данные
3. Бинарные данные
4. Текстовый файл

Вопрос 9.3.3. Укажите правильный вариант записи массива из 5 элементов в файл

1. fwrite(&(mas[i]),sizeof(int),1,fp);
2. fwrite(mas[i],sizeof(int),5,fp);
3. fwrite(&mas[5],sizeof(int),1,fp);
4. fwrite(&(mas[0]),sizeof(int),5,fp);

Шкала оценивания и критерии оценки:

Критерий	Баллы обучающегося	Минимальное количество баллов	Максимальное количество баллов
1. Количество правильных ответов на вопросы теста при общем количестве правильных ответов не менее, чем на 48 баллов и более		48	81
Итого:		**48**	**81**

Соответствие баллов шкале оценивания:

Количество баллов	Оценка обучающегося
73-81	отлично
61-72	хорошо
48-60	удовлетворительно
менее 48	неудовлетворительно

УСТНО-ПРАКТИЧЕСКИЙ ЭКЗАМЕН

Экзамен проводится в устной форме и с применением ПК для выполнения практического задания. На подготовку ответа студенту отводится 1-2 академических часа.

В билет включается по одному вопросу из пройденных тем примерного перечня вопросов, а также одна задача.

Первостепенной задачей студента является составление схемы алгоритма и написание текста программы практической части задания (задачи). При успешно сданной задаче студент переходит к теоретической (устной) части экзамена. Ответ должен содержать определения понятий, входящих в вопрос, синтаксические конструкции общего вида, примеры применения.

Для получения оценки «хорошо» или «отлично» необходимо дать содержательный и исчерпывающий ответ, привести примеры применения понятий к решению задач. Помимо этого, обучающемуся предлагается кратко ответить на два дополнительных вопроса по темам семестра (дать определение понятия, провести классификацию, привести примеры применения операций/алгоритмических конструкций). Вопросы выбираются из перечня вопросов к экзамену и формулируются преподавателем во время устной беседы или включаются в билет при проведении письменного экзамена.

Процедура проведения экзамена в устной и письменной форме описана в разделе Методические материалы, определяющие процедуры оценивания знаний, умений, навыков и (или) опыта деятельности, характеризующих этапы формирования компетенций настоящего документа.

Примерный перечень вопросов для подготовки к экзамену в 2-м семестре:

1. Алгоритм, свойства алгоритма, исполнители алгоритмов, способы записи алгоритмов.
2. Базовые алгоритмические структуры.
3. Характеристики алгоритмов.
4. Блок-схемы алгоритмов.
5. Теоретическая и практическая функция сложности.
6. Определение функции сложности алгоритма по результатам эксперимента
7. Алфавит языка. Управляющая последовательность.
8. Идентификаторы.
9. Служебные слова. Комментарии.
10. Этапы разработки программы.
11. Структура программы.
12. Идентификаторы.
13. Локальные и глобальные переменные.
14. Переменные и константы.
15. Операция присваивания и инициализации переменных.
16. Функция ввода данных printf.
17. Функция вывода данных scanf.
18. Спецификация преобразования.
19. Ввод/вывод символьных данных.
20. Ввод/вывод строковых данных.
21. Потоковый ввод/вывод (cin/cout).
22. Типы данных.
23. Унарные операции.
24. Бинарные операции.
25. Тернарные операции.
26. Мультипликативные и аддитивные операции.
27. Операции сдвига и поразрядные операции.
28. Операции увеличения и уменьшения.
29. Составное присваивание.
30. Операция sizeof() и операции отношения.
31. Логические операции.
32. Преобразование типов данных.
33. Выражения (арифметические и логические). Порядок выполнения выражений.
34. Условный оператор if.
35. Оператор выбора switch.
36. Организация цикла с оператором for.
37. Организация цикла с оператором while.
38. Организация цикла с оператором do while.
39. Операция запятая.
40. Оператор break, continue.
41. Объявление, инициализация и обработка одномерных массивов.
42. Объявление, инициализация и обработка двумерных массивов.
43. Статические массивы.
44. Динамические массивы.

45. Строки.
46. Функции. Основные понятия.
47. Объявление функций.
48. Прототип функции.
49. Определение функции, тело функции.
50. Вызов и тип возврата функций.
51. Формальные и фактические параметры функций.
52. Оператор return.
53. Передача данных по значению.
54. Передача данных по ссылке.
55. Определение указателей. Операция адрес и операция разадресации.
56. Использование указателей в качестве аргумента функции.
57. Область действия.
58. Область видимости.
59. Классы памяти.
60. Директивы препроцессора.
61. Структуры. Описание структуры. Объявление переменных типа структура.
62. Доступ к элементам структуры и операции с элементами структуры.
63. Общие понятия о файле. Указатель на файловую переменную. Открытие файла на чтение и запись.
64. Обращение к файлам.
65. Файловое хранение числовых данных. Форматированный ввод и вывод в файл и из файла.
66. Файловое хранение текстовых данных. Работа с символами в файле. Посимвольное чтение и запись в файл.
67. Работа со структурами. Чтение структур из файла и в файл.
68. Базовые понятия ООП: объекты и классы.
69. Свойства объектов
70. Основные принципы ООП: наследование свойств, полиморфизм объектов, Инкапсуляция.
71. Классы объектов.
72. Компоненты и их свойства.
73. Событийно-управляемая модель программирования.
74. Компонентно-ориентированный подход.
75. Требования к аппаратным и программным средствам интегрированной среды разработчика.
76. Интерфейс среды разработчика: характеристика, основные окна, инструменты, объекты.
77. Форма и размещение на ней управляющих элементов.
78. Панель компонентов и их свойства.
79. Окно кода проекта.
80. Состав и характеристика проекта.
81. Настройка среды и параметров проекта.
82. Основные компоненты (элементы управления) интегрированной среды разработки, их состав и назначение.
83. Дополнительные элементы управления.
84. Свойства компонентов.
85. Виды свойств.
86. Синтаксис определения свойств.
87. Назначения свойств и их влияние на результат.
88. Управление объектом через свойства.
89. События компонентов (элементов управления), их сущность и назначение.

Перечень задач для подготовки к экзамену

1. Определить, принадлежит ли точка с координатами x и y прямоугольнику с координатами x1, x2, y1, y2 (условие x>=x1 и x=<x2 и y>=y1 и y=<y2).
2. В компьютерном классе имеется n рабочих мест. В течении дня первый компьютер работал m часов, а каждый последующий на 10 минут больше, чем предыдущий. Определите дневной фонд рабочего времени класса.
3. Выполнить табуляцию функции y = x + sin x для x, изменяющегося на отрезке [x_нач.,x_кон.] с шагом dx. Результат оформить в виде таблицы с заголовком.
4. Даны массивы A(10), B(10). Сформировать массив C(10) из сумм соответствующих элементов массивов A и B. В массиве C сосчитать количество элементов, равных нулю. Массив C распечатать.
5. В заданной матрице определить максимальный и минимальный элементы. Переставить местами строки, содержащие их.
6. Создать массив из записей, которые содержат сведения о студентах: ФИО, год рождения, адрес. Выдать на экран данные студентов, год рождения которых соответствует году, введенному в виде запроса.
7. Заданы два квадратных уравнения: ax2+bx +c=0 и ax2+bx+d=0. Найти наименьший из вещественных корней этих уравнений. Решение уравнения оформить в виде функции.
8. Составить подпрограмму, которая формирует массив, каждый элемент которого равен сумме соответствующих элементов двух других массивов. Ввод исходных массивов выполняется вне данной подпрограммы.
9. Переписать файлы A и B в файл C последовательно друг за другом. Известно, что в файлах хранятся целые числа. Считать файлы A и B уже существующими.
10. Написать функцию power(), которая принимает два аргумента x и n и вычисляет x в степени n.
11. Написать функцию cum_sum(), которая вычисляет кумулятивную сумму для каждого элемента массива.
12. Написать функцию reverse(), которая разворачивает массив элементов, таким образом, все элементы следуют в обратном порядке.
13. Написать функцию unique(), которая подсчитывает число уникальных элементов в массиве.
14. Написать функцию isort(), которая упорядочивает элементы массива, используя алгоритм вставками.
15. Написать функцию datedelta(), которая вычисляет сколько дней прошло между двумя датами.
16. Написать функцию timedelta, которая находит разницу между двумя точками времени t2 > t1.
17. Написать функцию bin_search(), которая ищет указанный элемент e в массиве методом бинарного (двоичного) поиска.
18. Написать функцию shift(), которая циклически сдвигает все элементы массива на указанное число позиций m.
19. Написать две небольших функции, которые проверяют, что слово и число являются палиндромами.
20. Написать функцию transpose(), которая транспортирует матрицу (двумерный массив).
21. Написать функцию, которая проверяет сбалансированность скобок в строке (каждой открывающей должна соответствовать закрывающая), строка задана массивом символов.
22. Создать структуру rational, которая описывает рациональное число вида m/n и написать основные функции для работы с рациональными числами.

Пример экзаменационного билета

Экзаменационный билет №____

Часть I: теоретическая

1. Пользовательские типы в языке Си. Объявление, область применения, примеры.
2. Основные компоненты (элементы управления) интегрированной среды разработки, их состав и назначение.

Часть II: практическая

Составить программу, которая формирует массив, каждый элемент которого равен сумме элементов двух других массивов. Ввод исходных массивов выполняется вне данной подпрограммы.

Экзаменационный билет №____

Часть I: теоретическая

1. Директивы препроцессора.
2. Событийно-управляемая модель программирования.

Часть II: практическая

Создать структуру rational, которая описывает рациональное число вида m/n и написать основные функции для работы с рациональными числами.

Условия проведения экзамена:

Задание теоретической части предполагает устный ответ обучающегося.

Задания практической части выполняются на ПК с использованием MS Office, MS Visio, IDE Visual Studio C++ или на экзаменационном бланке.

Шкала оценивания и критерии оценки:

Критерии оценки	Баллы обучающегося	Минимальное количество баллов	Максимальное количество баллов
Уровень усвоения теоретического материала, предусмотренного программой		3	4
Умение выполнять практические задания, предусмотренные программой		3	4
Уровень знакомства с основной литературой, предусмотренной программой		1	2
Уровень знакомства с дополнительной литературой		0,5	1
Уровень раскрытия причинно-следственных связей		1	2
Уровень раскрытия междисциплинарных связей		1	2
Качество ответа (его общая композиция, логичность, убежденность, общая эрудиция)		1	2
Ответы на вопросы: полнота, аргументированность, убежденность, умение использовать ответы на вопросы для более полного раскрытия содержания вопроса		1	2
Деловые и волевые качества докладчика: ответственное отношение к работе, стремление к достижению высоких результатов, готовность к дискуссии, контактность		0,5	1
Итого баллов:		**12**	**20**

Соответствие баллов шкале оценивания:

Количество баллов	Оценка обучающегося
18-20	отлично
15-17	хорошо
12-14	удовлетворительно
менее 12	неудовлетворительно

Знания, умения и навыки обучающихся при промежуточном контроле в **форме экзамена** определяются оценками «отлично», «хорошо», «удовлетворительно», «неудовлетворительно».

1. «Отлично» – обучающийся глубоко и прочно усвоил весь программный материал, исчерпывающе, последовательно, грамотно и логически стройно его излагает, не затрудняется с ответом при видоизменении задания, свободно справляется с практическими заданиями, правильно обосновывает принятые решения, умеет самостоятельно обобщать и излагать материал, не допуская ошибок.

2. «Хорошо» – обучающийся твердо знает программный материал, грамотно и по существу излагает его, не допускает существенных неточностей в ответе на вопрос, может правильно применять теоретические положения и владеет необходимыми умениями и навыками при выполнении практических заданий.

3. «Удовлетворительно» – обучающийся усвоил только основной материал, но не знает отдельных деталей, допускает неточности, недостаточно правильные формулировки, нарушает последовательность в изложении программного материала и испытывает затруднения в выполнении практических заданий.

4. «Неудовлетворительно» – обучающийся не знает значительной части программного материала, допускает существенные ошибки, с большими затруднениями выполняет практические задания, а также если обучающийся после начала экзамена отказался его сдавать или нарушил правила сдачи экзамена (списывал, подсказывал и т.д.).

МЕТОДИЧЕСКИЕ МАТЕРИАЛЫ, ОПРЕДЕЛЯЮЩИЕ ПРОЦЕДУРЫ ОЦЕНИВАНИЯ ЗНАНИЙ, УМЕНИЙ, НАВЫКОВ И (ИЛИ) ОПЫТА ДЕЯТЕЛЬНОСТИ, ХАРАКТЕРИЗУЮЩИХ ЭТАПЫ ФОРМИРОВАНИЯ КОМПЕТЕНЦИЙ

Во время проведения лекционных занятий учитывается посещаемость обучающихся, оценивается их познавательная активность на занятии, предварительная подготовка к занятию.

Контрольная работа по разделам дисциплины проводится преподавателем по завершении изучения темы (тем) в часы аудиторных занятий. Образцы контрольной работы и перечень примерных задач, а также требования к оформлению работ предоставляются обучающимся заранее. Проверка и оценивание контрольных работ проводится преподавателем в течение одной-двух недель, но не позднее окончания рубежной аттестации. Баллы переводятся в систему оценок преподавателем в соответствии с утвержденной шкалой оценивания.

В случае невыполнения контрольной работы в установленные сроки обучающемуся необходимо погасить задолженность до проведения промежуточной аттестации. График погашения задолженности устанавливается преподавателем в индивидуальном порядке с учетом причин невыполнения.

Устный опрос проводится на лабораторных занятиях, и затрагивает как тематику предшествующих занятий, так и лекционный материал.

В случае невыполнения заданий в процессе обучения, их необходимо «отработать» до экзамена. Вид заданий, которые необходимо выполнить для ликвидации задолженности определяется в индивидуальном порядке, с учетом причин невыполнения.

Допуск обучающегося к защите лабораторной работы происходит при условии наличия у обучающегося печатной версии отчета по лабораторной работе.

Отчет по лабораторной работе представляется в печатном виде в формате, предусмотренном шаблоном отчета по лабораторной работе. Защита отчета проходит в форме доклада обучающегося по выполненной работе и ответов на вопросы преподавателя.

По окончании освоения дисциплины проводится промежуточная аттестация в виде устно-практического экзамена, что позволяет оценить достижение результатов обучения по дисциплине.

Экзамен может проводиться в устной или письменной форме. На подготовку к устному ответу на вопросы билета и решение задачи отводится 1 академический час. Для письменного ответа отводится 2 академических часа. Оценивание проводится преподавателем(-ями) непосредственно во время экзамена.

Перечень вопросов и список учебной литературы для подготовки к экзамену предоставляется в начале семестра.

Во время сдачи промежуточной аттестации в устной форме в аудитории может находиться одновременно не более 4-5 обучающихся, при выполнении заданий на компьютере – по одному обучающемуся за персональным компьютером (не более 12 студентов).