МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ

Федеральное государственное бюджетное образовательное учреждение высшего образования
«Воронежский государственный технический университет»

УТВЕРЖДАЮ

Декан факультета ФИТКБ

_____ /Гусев П.Ю./

28.02.2023 г.

РАБОЧАЯ ПРОГРАММА ДИСЦИПЛИНЫ

«Языки программирования»

Специальность 10.05.01 Компьютерная безопасность

Специализация специализация № 4 "Безопасность компьютерных систем и сетей (связь, информационные и коммуникационные технологии)"

Квалификация выпускника специалист по защите информации

Нормативный период обучения 5 лет и 6 м.

Форма обучения очная

Автор программы _____ /А.С. Кольцов /

Заведующий кафедрой Систем информационной безопасности _____ / А.Г. Остапенко /

Руководитель ОПОП _____ / А.Г. Остапенко /

Воронеж 2023

1. ЦЕЛИ И ЗАДАЧИ ДИСЦИПЛИНЫ

1.1. Цели дисциплины: Овладение фундаментальными знаниями и практическими навыками по применению языков программирования для реализации алгоритмов и разработки программного обеспечения для решения профессиональных, исследовательских и прикладных задач, в том числе в сфере информационной безопасности.

1.2. Задачи освоения дисциплины

– изучение основных принципов программирования на языках высокого уровня;

– изучение базовых конструкций структурного программирования;

– изучение объектно-ориентированного программирования;

– изучение основные алгоритмов обработки массивов, ссылочных структур;

– изучение современных технологий программирования

2. МЕСТО ДИСЦИПЛИНЫ В СТРУКТУРЕ ОПОП

Дисциплина «Языки программирования» относится к дисциплинам обязательной части блока Б1.

3. ПЕРЕЧЕНЬ ПЛАНИРУЕМЫХ РЕЗУЛЬТАТОВ ОБУЧЕНИЯ ПО ДИСЦИПЛИНЕ

Процесс изучения дисциплины «Языки программирования» направлен на формирование следующих компетенций:

ОПК-7 - Способен создавать программы на языках высокого и низкого уровня, применять методы и инструментальные средства программирования для решения профессиональных задач, осуществлять обоснованный выбор инструментария программирования и способов организации программ;

Компетенция	**Результаты обучения, характеризующие сформированность компетенции**
ОПК-7	Знать – общие принципы построения, области и особенности применения языков программирования высокого и низкого уровня; – языки программирования высокого и низкого уровня (структурное, объектно-ориентированное программирование) – существующие парадигмы программирования, современные технологии разработки программного обеспечения.
	Уметь – работать с интегрированной средой разработки программного обеспечения; – умеет разрабатывать и реализовывать на языке

	высокого уровня алгоритмы решения типовых профессиональных задач – выбирать необходимые языки программирования и системы разработки программных средств для решения профессиональных задач.
	Владеть – навыками создания программ на различных языках программирования; – навыками применения существующих реализаций структур данных и алгоритмов; – навыками разработки, документирования, тестирования и отладки программ.

4. ОБЪЕМ ДИСЦИПЛИНЫ

Общая трудоемкость дисциплины «Языки программирования» составляет 15 з.е.

Распределение трудоемкости дисциплины по видам занятий

очная форма обучения

Виды учебной работы	Всего часов	Семестры		
		2	3	4
Аудиторные занятия (всего)	216	72	72	72
В том числе:				
Лекции	108	36	36	36
Лабораторные работы (ЛР)	108	36	36	36
Самостоятельная работа	288	144	18	126
Курсовой проект	+		+	
Часы на контроль	36	-	-	36
Виды промежуточной аттестации - экзамен, зачет с оценкой	+	+	+	+
Общая трудоемкость: академические часы зач.ед.	 540 15	 216 6	 90 2,5	 234 6,5

5. СОДЕРЖАНИЕ ДИСЦИПЛИНЫ (МОДУЛЯ)

5.1 Содержание разделов дисциплины и распределение трудоемкости по видам занятий

очная форма обучения

№ п/п	Наименование темы	Содержание раздела	Лекц	Лаб. зан.	СРС	Всего, час
1	Основы языка программирования С/ С++	Характеристика языка, стандарты, этапы создания исполняемого файла. Инструментальные средства разработки программного обеспечения на языке С/С++. Базовые конструкции языка С/С++. Алфавит, идентификаторы, служебные слова, константы. Операторы последовательного выполнения, операторы выбора, операторы цикла, операторы передачи управления. Указатели и ссылки. Операции с указателями. Организация работы с динамической памятью. Массивы. Одномерные массивы. Двумерные массивы. Динамические массивы. Строки. Структуры, объединения, битовые поля. Перечислимые типы. Переопределение типов. Форматируемый ввод-вывод данных. Работа с файлами. Функции. Определение, описание и вызов функции. Способы передачи параметров в функцию. Рекурсивные функции. Указатели на функции. Встраиваемые функции. Параметры по умолчанию. Параметры функции main. Функции с переменным числом параметров. Перегрузка функций. Шаблоны функций. Классы памяти. Область видимости, действия, время жизни переменной. Пространства имен. Пространство имен std. Препроцессорные средства. Директивы и стадии препроцессорной обработки. Условная компиляция. Макроподстановки. Операции с разрядами. Поразрядные логические операции и операции сдвига**.** Понятия абстракции, абстрактного типа данных. Основные понятия объектно-ориентированного программирования (ООП). Инкапсуляция, наследование и полиморфизм. Классы. Понятие класса. Доступ к компонентам класса. Конструкторы и деструкторы. Поля и методы класса. Указатель this. Статические компоненты класса. Дружественные функции и классы. Перегрузка операторов. Локальные классы. Наследование. Полиморфизм. Виртуальные функции. Абстрактные классы. Потоковые средства ввода/вывода данных. Иерархия классов потоков в языке С++. Форматированный ввод/вывод. Создание собственных манипуляторов. Файловый ввод/вывод. Исключительные ситуации. Основные понятия. Механизм обработки исключений. Классы исключений. Шаблоны классов.	36	36	144	216
2	Программирование на Python	Операторы ввода-вывода Python. Простые типы данных. Программирование разветвляющихся и циклических процессов в Python. Списки, кортежи. Множества. Словари. Строки. Индексация. Срезы в Python.	36	36	18	90

		Работа с файлами и каталогами в Python. Обработка ошибок и исключений. Функции. Возвращение значений из функций. Функции с переменным числом аргументов. Лямбда-функции в Python. Область видимости переменных. Декораторы. Создание оконных приложений. Модуль tkinter. Интерфейс. Кнопки. Позиционирование. Виджеты: Label, Entry, Checkbutton. Виджеты: Frame, Listbox, Scrollbar. Виджеты: Text, Treeview Графика в Python: Canvas. Основы ООП. Классы и объекты. Наследование в Python. Инструменты для Data Science. Работа с базой данных в Python. Нейронные сети вPython.				
3	Программирование на Java	Понимание архитектуры Java и настройка среды разработки. Архитектура платформы: JDK, JRE, JVM — назначение и отличия. Установка и настройка JDK, переменная окружения JAVA_HOME. Обзор популярных IDE: IntelliJ IDEA, Eclipse, NetBeans.Основы синтаксиса и типы данных. Управление потоком выполнения. Обработка исключений: try-catch-finally, иерархия Throwable. Классы и объекты: поля, методы, конструкторы, блок инициализации. Инкапсуляция: модификаторы доступа (private, protected, public, package-private). Наследование: ключевое слово extends, переопределение методов, super. Полиморфизм: динамическая диспетчеризация, приведение типов, instanceof. Абстрактные классы и интерфейсы: abstract, default и static методы в интерфейсах. Вложенные классы: static nested classes, inner classes, локальные и анонимные классы. Перечисления (enum) и их возможности. Стандартная библиотека Java (Java SE API). Коллекции (java.util). Иерархия Collections Framework: List, Set, Queue, Map. Реализации: ArrayList, LinkedList, HashSet, TreeSet, HashMap, TreeMap. Итераторы, Comparable vs Comparator, сортировка. Дженерики (Generics). Функциональное программирование и Stream API. Лямбда-выражения: синтаксис, функциональные интерфейсы (Function, Predicate, Consumer, Supplier). Stream API: создание потоков, промежуточные и терминальные операции. Параллельные потоки (parallelStream()), особенности производительности. Optional: работа с возможными отсутствующими значениями. Многопоточность и конкурентность. Продвинутые темы и инструменты. Знакомство с профессиональными практиками разработки на Java.	36	36	126	234
		Итого	**108**	**108**	**288**	**504**

5.2 Перечень лабораторных работ

2 Семестр. C++

1. Настройка среды разработки и линейные вычисления
2. Ветвление и циклы

3. Основы работы с массивами. Динамические массивы.
4. Символы и строки в формате C и C++
5. Работа с файлами в формате C и C++.
6. Основы объектно-ориентированного программирования
7. Наследование и полиморфизм
8. Виртуальные методы, абстрактные классы, интерфейсы
9. Ассоциация и зависимость, агрегация и композиция

2 Семестр. Python

1. Настройка среды и виртуального окружения
2. Функции и модули
3. Работа с файлами и JSON
4. Классы и объекты
5. Магические методы и декораторы
6. Генераторы и итераторы
7. Стандартные коллекции
8. Работа с сетью и API
9. Работа с базами данных

4 Семестр. Java.

1. Настройка среды и создание консольного приложения
2. Реализация алгоритмов с использованием циклов и массивов
3. Разработка иерархии классов с наследованием и полиморфизмом
4. Работа с коллекциями: сортировка, фильтрация, поиск
5. Введение в библиотеку Java Swing. Основные визуальные компоненты Java Swing
6. Обработка событий в Java Swing
7. Создание изображений средствами JAVA
8. Сборка проекта с помощью Maven/Gradle
9. Итоговый проект: консольное или GUI-приложение (JavaFX/Swing)

6. ПРИМЕРНАЯ ТЕМАТИКА КУРСОВЫХ ПРОЕКТОВ (РАБОТ) И КОНТРОЛЬНЫХ РАБОТ

В соответствии с учебным планом освоение дисциплины предусматривает выполнение курсового проекта в 3 семестре для очной формы обучения.

Примерная тематика курсового проекта: «Разработка программного обеспечения на базе принципов объектно-ориентированного программирования» (по вариантам)

Варианты заданий:

1) Разработка программы учета больных вирусом COVID-19
2) Разработка программы расселения детей в лагере
3) Разработка программы расписания движения самолетов
4) Разработка программы продажи билетов на самолёт

5) Разработка программы учета переговоров работников такси
6) Разработка программы расчета заработной платы сотрудников Яндекс Такси
7) Разработка программы учета проживающих в общежитии
8) Разработка программы учета выполненных заказов Яндекс такси
9) Разработка программы учета научной деятельности сотрудников
10) Разработка программы учета выплат заработной платы работникам приюта
11) Разработка программы учета клиентов салона красоты
12) Разработка программы ассортимента машин в таксопарке
13) Разработка программы подбора материалов для ремонта квартиры
14) Разработка программы учета выполнения работы лаборанта
15) Разработка программы учета слушателей переподготовки
16) Разработка программы учета сведений о музыкальном конкурсе
17) Разработка программы учета сведений о пациентах медицинского центра
18) Разработка программы продажи железнодорожных билетов
19) Разработка программы планирования факультативных учебных дисциплин для студентов
20) Разработка программы учета сведений об игроках хоккейной команды
21) Разработка программы продажи авиабилетов
22) Разработка программы учета автомобилей автопарка
23) Разработка программы расчета стипендии
24) Разработка программы учета выполнения работы мастера по работе с волосами
25) Разработка программы планирования бюджета при ремонте машины
26) Разработка программы расчета оплаты лечения
27) Разработка программы подбора санатория
28) Разработка программы учета продаж билетов в аквапарк
29) Разработка программы учета продаж квартир
30) Разработка программы автоматизации учета кадров на предприятии

Задачи, решаемые при выполнении курсового проекта:

– Анализ объекта, определение статических и динамических характеристик.
– Разработка алгоритма, реализующего задачу.
– Реализация алгоритма, реализующего задачу на выбранном языке программирования.

Курсовой проект включат в себя расчетно-пояснительную записку и презентационные материалы.

7. ОЦЕНОЧНЫЕ МАТЕРИАЛЫ ДЛЯ ПРОВЕДЕНИЯ ПРОМЕЖУТОЧНОЙ АТТЕСТАЦИИ ОБУЧАЮЩИХСЯ ПО ДИСЦИПЛИНЕ

7.1. Описание показателей и критериев оценивания компетенций на различных этапах их формирования, описание шкал оценивания

7.1.1 Этап текущего контроля

Результаты текущего контроля знаний и межсессионной аттестации оцениваются по следующей системе:

«аттестован»;

«не аттестован».

Компетенция	Результаты обучения, характеризующие сформированность компетенции	Критерии оценивания	Аттестован	Не аттестован
ОПК-7	Знать – общие принципы построения, области и особенности применения языков программирования высокого уровня; – языки программирования высокого уровня (структурное, объектно-ориентированное программирование) – существующие парадигмы программирования, современные технологии разработки программного обеспечения.	защита лабораторных работ, ответы на теоретические и практические вопросы промежуточного контроля	Выполнение работ в срок, предусмотренный в рабочих программах	Невыполнение работ в срок, предусмотренный в рабочих программах
	Уметь – работать с интегрированной средой разработки программного обеспечения; – умеет разрабатывать и реализовывать на языке высокого уровня алгоритмы решения типовых профессиональных задач – выбирать необходимые языки программирования и системы разработки программных средств для решения профессиональных	Выполнение лабораторных работ, выполнение этапов курсового проекта.	Выполнение работ в срок, предусмотренный в рабочих программах	Невыполнение работ в срок, предусмотренный в рабочих программах

	задач.			
	Владеть – навыками создания программ на различных языках программирования; – навыками применения существующих реализаций структур данных и алгоритмов; – навыками разработки, документирования, тестирования и отладки программ.	Решение прикладных задач в конкретной предметной области	Выполнение работ в срок, предусмотренный в рабочих программах	Невыполнение работ в срок, предусмотренный в рабочих программах

7.1.2 Этап промежуточного контроля знаний

Результаты промежуточного контроля знаний оцениваются в 2, 3, 4 семестре для очной формы обучения по четырехбалльной системе:

«отлично»;

«хорошо»;

«удовлетворительно»;

«неудовлетворительно».

Компетенция	**Результаты обучения, характеризующие сформированность компетенции**	**Критерии оценивания**	**Отлично**	**Хорошо**	**Удовл.**	**Неудовл.**
ОПК-7	Знать – общие принципы построения, области и особенности применения языков программирования высокого и низкого уровня; – языки программирования высокого и низкого уровня (структурное, объектно-ориентированное	Тест	Выполнение теста на 90-100%	Выполнение теста на 80-90%	Выполнение теста на 70-80%	В тесте менее 70% правильных ответов

	программирование) – программному обеспечению существующие парадигмы программирования, современные технологии разработки программного обеспечения.					
	Уметь – работать с интегрированной средой разработки программного обеспечения; – умеет разрабатывать и реализовывать на языке высокого уровня алгоритмы решения типовых профессиональных задач – выбирать необходимые языки программирования и системы разработки программных средств для решения профессиональных задач.	Решение стандартных практических задач	Задачи решены в полном объеме и получены верные ответы	Продемонстрирован верный ход решения всех, но не получен верный ответ во всех задачах	Продемонстрирован верный ход решения в большинстве задач	Задачи не решены
	Владеть – навыками создания программ на различных языках программирования; – навыками применения существующи	Решение прикладных задач в конкретной предметной области	Задачи решены в полном объеме и получены верные ответы	Продемонстрирован верный ход решения всех, но не получен верный ответ во всех задачах	Продемонстрирован верный ход решения в большинстве задач	Задачи не решены

	х реализаций структур данных и алгоритмов; – навыками разработки, документирования, тестирования и отладки программ.					

7.2 Примерный перечень оценочных средств (типовые контрольные задания или иные материалы, необходимые для оценки знаний, умений, навыков и (или) опыта деятельности)

7.2.1 Примерный перечень заданий для подготовки к тестированию

1. Класс - это:

а) любой тип данных, определяемый пользователем

б) тип данных, определяемый пользователем и сочетающий в себе данные и функции их обработки

в) структура, для которой в программе имеются функции работы с нею

2. Членами класса могут быть

а) как переменные, так и функции, могут быть объявлены как private и как public

б) только переменные, объявленные как private

в) только функции, объявленные как private

г) только переменные и функции, объявленные как private

д) только переменные и функции, объявленные как public

3. Что называется конструктором?

а) метод, имя которого совпадает с именем класса и который вызывается автоматически при создании объекта класса

б) метод, имя которого совпадает с именем класса и который вызывается автоматически при объявлении класса (до создания объекта класса)

в) метод, имя которого необязательно совпадает с именем класса и который вызывается при создании объекта класса

г) метод, имя которого совпадает с именем класса и который необходимо явно вызывать из головной программы при объявлении объекта класса

4. Объект - это

а) переменная, содержащая указатель на класс

б) экземпляр класса

в) класс, который содержит в себе данные и методы их обработки

5. Отметьте правильные утверждения

а) конструкторы класса не наследуются

б) конструкторов класса может быть несколько, их синтаксис определяется программистом

в) конструкторов класса может быть несколько, но их синтаксис должен подчиняться правилам перегрузки функций

г) конструктор возвращает указатель на объект

д) конструктор не возвращает значение

6. Что называется деструктором?

а) метод, который уничтожает объект

б) метод, который удаляет объект

в) метод, который освобождает память, занимаемую объектом

г) системная функция, которая освобождает память, занимаемую объектом

7. Выберите правильные утверждения

а) у конструктора могут быть параметры

б) конструктор наследуется, но должен быть перегружен

в) конструктор должен явно вызываться всегда перед объявлением объекта

г) конструктор вызывается автоматически при объявлении объекта

д) объявление каждого класса должно содержать свой конструктор

е) если конструктор не создан, компилятор создаст его автоматически

8. Выберите правильные утверждения

а) деструктор - это метод класса, применяемый для удаления объекта

б) деструктор - это метод класса, применяемый для освобождения памяти, занимаемой объектом

в) деструктор - это отдельная функция головной программы, применяемая для освобождения памяти, занимаемой объектом

г) деструктор не наследуется

д) деструктор наследуется, но должен быть перегружен

9. Что называется наследованием?

а) это механизм, посредством которого производный класс получает элементы родительского и может дополнять либо изменять их свойства и методы

б) это механизм переопределения методов базового класса

в) это механизм, посредством которого производный класс получает все поля базового класса

г) это механизм, посредством которого производный класс получает элементы родительского, может их дополнить, но не может переопределить

10. Выберите правильное объявление производного класса

а) class MoreDetails:: Details;

б) class MoreDetails: public class Details;

в) class MoreDetails: public Details;

г) class MoreDetails: class(Details);

11. Выберите правильные утверждения:

а) если элементы класса объявлены как private, то они доступны только наследникам класса, но не внешним функциям

б) если элементы класса объявлены как private, то они недоступны ни наследникам класса, ни внешним функциям

в) если элементы объявлены как public, то они доступны наследникам класса, но не внешним функциям

г) если элементы объявлены как public, то они доступны и наследникам класса, и внешним функциям

12. Возможность и способ обращения производного класса к элементам базового определяется

а) ключами доступа: private, public, protected в теле производного класса

б) только ключом доступа protected в заголовке объявления производного класса

в) ключами доступа: private, public, protected в заголовке объявления производного класса

г) ключами доступа: private, public, protected в теле базового класса

13. Выберите правильные соответствия между спецификатором базового класса, ключом доступа в объявлении производного класса и правами доступа прозводного класса к элементам базового

а) ключ доступа - public; в базовом классе: private; права доступа в производном классе - protected

б) ключ доступа - любой; в базовом классе: private; права доступа в производном классе - нет прав

в) ключ доступа - protected или public ; в базовом классе: protected; права доступа в производном классе - protected

г) ключ доступа - private; в базовом классе: public; права доступа в производном классе - public

д) ключ доступа – любой; в базовом классе: public; права доступа в производном классе – такие же, как ключ доступа

14. Дружественная функция - это

а) функция другого класса, среди аргументов которой есть элементы данного класса

б) функция, объявленная в классе с атрибутом friend, но не являющаяся

членом класса;

в) функция, являющаяся членом класса и объявленная с атрибутом friend;

г) функция, которая в другом классе объявлена как дружественная данному.

7.2.2 Примерный перечень заданий для решения стандартных задач

1. Какой тип переменной используется в коде: int a = 5;

а) Знаковое 8-бит целое
б) Знаковое 64-бит целое
в) Знаковое 32-бит целое

2. Что делает оператор «%»

а) Возвращает процент от суммы
б) Возвращает остаток от деления
в) Возвращает тригонометрическую функцию
г) Ни чего из выше перечисленного.

3. Что сделает программа выполнив следующий код: Console.WriteLine(«Hello, World!»);

а) Напишет на новой строчке Hello, World! (+)
б) Напишет Hello, World!
в) Удалит все значения с Hello, World!
г) Вырежет слово Hello, World! из всего текста

4. Как сделать инкрементацию числа

а) ++
б) —
в) %%
г) !=

5. Как сделать декрементация числа

а) %%
б) —
в) !=
г) ++

6. Чему будет равен c, если int a = 10; int b = 4; int c = a % b;

а) 11
б) 2
в) 3
г) 1

7. Чему будет равен c, если int a = 10; int b = 4; bool c = (a == 10 && b == 4);

а) True

б) False
в) Null
г) 14

8. Чему будет равен c, если int a = 0; int c = a—;
а) Null
б) -1
в) 0
г) 1

9. Чему будет равен c, если int a = 0; int c = —a;
а) Null
б) -1
в) 0
г) 1

10. Чему равен d, если int a = 0; int b = a++; int c = 0; int d = a + b + c + 3;
а) 3
б) True
в) False
г) 4

7.2.3 Примерный перечень заданий для решения прикладных задач

1. Определить комбинированный (структурный) тип для представления анкеты студента, состоящей из его фамилии, дня рождения и пола. «День рождения» состоит из полей: «число», «месяц», «год». Ввести информацию по 25 студентам из группы. Вывести фамилию самого старшего мальчика из группы.
2. Определить комбинированный (структурный) тип для представления анкеты жителя, состоящей из его фамилии, названия города, где он проживает, и городского адреса. Адрес состоит из полей: «улица», «дом», «квартира». Ввести информацию по 100 жителям. Вывести фамилии жителей, которые живут в одном городе с первым жителем из списка.
3. Определить комбинированный (структурный) тип для представления анкеты студента, состоящей из его фамилии, дня рождения и пола. «День рождения» состоит из полей: «число», «месяц», «год». Ввести информацию по 25 студентам из группы. Вывести все фамилии девочек, родившихся в декабре.
4. Дана квадратная матрица порядка 10. Заменить нулями элементы, лежащие одновременно выше ниже главной диагонали (включая эту диагональ)
5. Дана квадратная матрица порядка 10 Вывести матрицу, отсортированную по столбцам.
6. Дана целочисленная матрица размера 10x8. Последний столбец, содержащий минимальный элемент, поменять местами с первым столбцом, содержащим максимальный элемент двумерного массива.

7. Дана квадратная матрица порядка 10. Транспонировать матрицу.

8. Дана квадратная матрица порядка 10. Вычистить определитель матрицы.

9. Дана квадратная матрица порядка 10. Выполнить сортировку элементов главной диагонали по возрастанию.

10. Открыть текстовый файл в необходимом режиме (на чтение, на чтение и запись, на добавление). Променять местами четные и нечетные строки текста. Записать измененные строки в новый файл.

11. Открыть текстовый файл в необходимом режиме (на чтение, на чтение и запись, на добавление). Вставить перед каждым восклицательным знаком вопросительный. Записать измененные строки в новый файл.

12. Открыть текстовый файл в необходимом режиме (на чтение, на чтение и запись, на добавление). Вывести на экран и записать в новый текстовый файл текст, исключив повторяющиеся пробелы между словами.

13. Задан текст. Проверить правильность написания знаков пунктуации. Слева от знака пунктуации пробел не ставится, справа ставится один пробел. Исправленный текст вывести на экран.

14. Задан текст. Сформировать строку из всех аббревиатур из текста через запятую и вывести ее на экран.

15. Задан текст. Вывести на экран предложения в порядке возрастания количества слов в них.

7.2.4 Примерный перечень вопросов для подготовки к зачету с оценкой

Семестр 2: C++

1. В чем заключаются основные концептуальные отличия языка C++ от языка C?
2. Объясните разницу между размещением данных в стеке (stack) и в куче (heap).
3. Каковы различия между указателями и ссылками в C++?
4. Для чего используется ключевое слово const и что такое const-корректность?
5. Опишите назначение конструкторов и деструкторов, а также порядок их вызова при наследовании.
6. Что такое конструктор копирования и оператор присваивания? Когда они вызываются компилятором автоматически?
7. Объясните концепцию семантики перемещения (move semantics) и rvalue-ссылок (C++11).
8. Что такое принцип RAII (Resource Acquisition Is Initialization) и как он помогает управлять ресурсами?
9. Как реализуется полиморфизм времени выполнения с помощью виртуальных функций?
10. Что такое абстрактный класс и чисто виртуальная функция?

11. В чем заключается проблема ромба (diamond problem) при множественном наследовании и как она решается?
12. Для чего используются дружественные функции и классы (friend)?
13. Что такое шаблоны (templates) функций и классов?
14. Перечислите основные контейнеры стандартной библиотеки шаблонов (STL) и их назначение.
15. Что такое итераторы в C++ и какие виды итераторов существуют?
16. Опишите различия между умными указателями unique_ptr, shared_ptr и weak_ptr.
17. Как устроена обработка исключений в C++ (try, catch, throw)?
18. В каких контекстах используется ключевое слово static в C++?
19. Что такое пространства имен (namespace) и зачем они нужны?
20. Что такое перегрузка операторов и какие операторы нельзя перегружать?
21. Что такое лямбда-выражения и какой у них синтаксис?
22. Для чего предназначено ключевое слово auto и как работает вывод типов?
23. Какова роль препроцессора и директив #include, #define?
24. Что понимается под неопределенным поведением (Undefined Behavior) в C++?
25. Что такое inline функции и какова цель их использования?

Семестр 3: Python

1. В чем разница между компилируемыми и интерпретируемыми языками на примере Python?
2. Что такое динамическая типизация и какие у нее есть преимущества и недостатки?
3. Чем отличаются изменяемые (mutable) и неизменяемые (immutable) типы данных?
4. В чем разница между списком (list) и кортежем (tuple)?
5. Как реализованы словари (dict) в Python и какова сложность основных операций?
6. Для чего используются множества (set) и чем они отличаются от списков?
7. Что такое генераторы списков (List Comprehensions) и в чем их преимущество?
8. В чем различие между объявлением функции через def и lambda?
9. Что означают аргументы *args и **kwargs в определении функции?
10. Что такое декораторы и как они работают?
11. Что такое генераторы и как используется ключевое слово yield?
12. Для чего нужны менеджеры контекста и конструкция with?

13. Опишите структуру обработки исключений в Python (try, except, finally, else).

14. Как создаются классы в Python и какова роль метода __init__?

15. Как реализуется наследование в Python и для чего нужна функция super()?

16. Что такое MRO (Method Resolution Order) при множественном наследовании?

17. Что такое магические методы (Dunder methods) и приведите примеры их использования.

18. В чем разница между областями видимости global и nonlocal?

19. Что такое GIL (Global Interpreter Lock) и как он влияет на многопоточность?

20. Как работает система модулей и пакетов (import) в Python?

21. Для чего используется конструкция if __name__ == "__main__"?

22. В чем разница между поверхностным (copy) и глубоким (deepcopy) копированием объектов?

23. Что такое виртуальные окружения (venv) и зачем они нужны?

24. Что регламентирует документ PEP 8?

25. Каковы основы асинхронного программирования в Python (asyncio)?

7.2.5 Примерный перечень вопросов для подготовки к экзамену

Семестр 4: Java

1. В чем различия между JVM, JRE и JDK?

2. Как обеспечивается платформонезависимость Java («Write Once, Run Anywhere»)?

3. Чем отличаются примитивные типы данных от ссылочных в Java?

4. Что такое автоупаковка (Autoboxing) и автораспаковка (Unboxing)?

5. Почему класс String в Java является неизменяемым (immutable)?

6. Объясните смысл каждой части сигнатуры метода public static void main(String[] args).

7. Опишите жизненный цикл объекта в Java.

8. Перечислите модификаторы доступа в Java и их область видимости.

9. Как реализуется наследование в Java и какие существуют ограничения?

10. В чем ключевые различия между интерфейсом и абстрактным классом?

11. Что такое полиморфизм и в чем разница между перегрузкой (overloading) и переопределением (overriding) методов?

12. Для чего используется ключевое слово static (поля, методы, блоки инициализации)?

13. Что означает ключевое слово final при применении к переменным, методам и классам?

14. В чем разница между проверяемыми (Checked) и непроверяемыми (Unchecked) исключениями?

15. Что такое конструкция try-with-resources?

16. Опишите иерархию коллекции (Collections Framework) в Java.

17. В чем практическая разница между ArrayList и LinkedList?

18. Как внутренне устроен класс HashMap?

19. Что такое дженерики (Generics) и что такое стирание типов (Type Erasure)?

20. Какими способами можно создать поток в Java (Thread, Runnable)?

21. Для чего используется ключевое слово synchronized?

22. Опишите базовые принципы работы Сборщика мусора (Garbage Collection).

23. Что такое лямбда-выражения и Stream API, появившиеся в Java 8?

24. Что такое перечисления (enum) в Java?

25. Что такое сериализация объектов и как она реализуется в Java?

26. Сравните модели управления памятью в C++, Python и Java (ручное управление, подсчет ссылок, сборка мусора).

27. В чем заключается разница между статической и динамической типизацией на примере этих трех языков?

28. Сравните модели многопоточности: нативные потоки ОС (C++, Java) и модель глобальной блокировки интерпретатора (Python).

29. Как реализована концепция интерфейсов в Java, абстрактных классов в C++ и протоколов (ABC) в Python?

30. Сравните модели исполнения кода: компиляция в нативный код (C++), байт-код JVM (Java) и байт-код виртуальной машины Python

31. Сравните модели компиляции и исполнения: компиляция в машинный код (C++), байт-код с JIT-компиляцией (Java) и интерпретация байт-кода (Python). Как это влияет на время запуска и производительность?

32. Как различается организация памяти для объектов в C++ (явное выделение), Java (куча + GC) и Python (куча + подсчет ссылок + GC)?

33. Сравните механизмы передачи аргументов в функции: передача по значению/ссылке в C++ vs. передача «по присваиванию» (pass-by-assignment) в Python и Java.

34. В чем разница между обработкой ошибок через коды возврата/исключения (C++), иерархию Checked/Unchecked исключений (Java) и подход EAFP (Python)?

35. Сравните возможности метапрограммирования: шаблоны и макросы (C++), аннотации и Reflection (Java), декораторы и метаклассы (Python).

36. Сравните реализацию обобщенного программирования (Generics): шаблоны времени компиляции в C++ vs. стирание типов (Type Erasure) в Java vs. утиная типизация и дженерики в Python.

37. Как реализуется полиморфизм: через виртуальные таблицы (C++), динамическую диспетчеризацию JVM (Java) и словари атрибутов/протоколы (Python)?

38. Сравните подходы к инкапсуляции: жесткий контроль доступа (C++, Java) vs. соглашение об именовании («private» через подчеркивание) в Python.

39. В чем разница между статической типизацией с выводом типов (auto в C++, var в Java) и динамической типизацией с аннотациями (Python)?

40. Как каждый из языков поддерживает функциональное программирование: лямбда-исчисления, замыкания, неизменяемость и потоки данных?

41. Сравните возможности кроссплатформенной разработки: зависимость от компилятора (C++), виртуальная машина (Java) и интерпретатор (Python).

42. В каких сценариях предпочтительнее использовать C++ (системное программирование, High-Frequency Trading), Java (корпоративные системы, Big Data) или Python (Data Science, скриптинг, прототипирование)?

43. Как каждый язык поддерживает работу с внешними библиотеками (FFI): C API и обертки (C++), JNI (Java) и модули C-extensions/ctypes (Python)?

44. Сравните возможности рефакторинга и статического анализа кода: компилятор как гарант (C++), мощные IDE на основе типа (Java) и линтеры/муру (Python).

45. Как эволюционируют языки: комитет стандартизации ISO (C++), JCP и релизный цикл Oracle (Java), PEP и сообщество (Python)? Как это влияет на обратную совместимость?

7.2.6. Методика выставления оценки при проведении промежуточной аттестации

Промежуточная аттестация в форме зачета с оценкой проводятся по тест-билетам, каждый из которых содержит 10 вопросов и задачу. Каждый правильный ответ на вопрос в тесте оценивается 1 баллом, задача оценивается в 10 баллов (5 баллов верное решение и 5 баллов за верный ответ). Максимальное количество набранных баллов – 20.

Оценка «Неудовлетворительно» ставится в случае, если студент набрал менее 6 баллов.

Оценка «Удовлетворительно» ставится в случае, если студент набрал от 6 до 10 баллов

Оценка «Хорошо» ставится в случае, если студент набрал от 11 до 15 баллов.

Оценка «Отлично» ставится, если студент набрал от 16 до 20 баллов.

Промежуточная аттестация в форме экзамена проводится по билетам, каждый из которых содержит 2 теоретических вопроса и задачу. Оценивается по принятой шкале («отлично», «хорошо», «удовлетворительно», «неудовлетворительно»).

Оценка «отлично» - за правильное выполнение всех заданий билета и правильный ответ на дополнительные вопросы по всей дисциплине.

Оценка «хорошо» - за правильное решение задачи и ответ на теоретические вопросы билета при не ответе на дополнительные вопросы.

Оценка «удовлетворительно» - за правильное решение задачи и раскрытии одного теоретического вопроса.

Оценка «неудовлетворительно» - если не решена задача.

7.2.7 Паспорт оценочных материалов

№ п/п	Контролируемые разделы (темы) дисциплины	Код контролируемой компетенции	Наименование оценочного средства
1	Основы языка программирования С/ С++	ОПК-7	Тест, защита лабораторных работ
2	Программирование на Python	ОПК-7	Тест, защита лабораторных работ, требования к курсовому проекту
3	Программирование на Java	ОПК-7	Тест, защита лабораторных работ

7.3. Методические материалы, определяющие процедуры оценивания знаний, умений, навыков и (или) опыта деятельности

Тестирование осуществляется, либо при помощи компьютерной системы тестирования, либо с использованием выданных тест-заданий на бумажном носителе. Время тестирования 30 мин. Затем осуществляется проверка теста экзаменатором и выставляется оценка согласно методики выставления оценки при проведении промежуточной аттестации.

Решение стандартных задач осуществляется, либо при помощи компьютерной системы тестирования, либо с использованием выданных задач на бумажном носителе. Время решения задач 30 мин. Затем осуществляется проверка решения задач экзаменатором и выставляется оценка, согласно методики выставления оценки при проведении промежуточной аттестации.

Решение прикладных задач осуществляется, либо при помощи интегрированной среды разработки (IDE)**,** либо с использованием выданных задач на бумажном носителе. Время решения задач 30 мин. Затем осуществляется проверка решения задач экзаменатором и выставляется

оценка, согласно методики выставления оценки при проведении промежуточной аттестации.

Защита курсового проекта осуществляется согласно требованиям, предъявляемым к проекту, описанным в методических материалах. Примерное время защиты на одного студента составляет 20 мин.

8 УЧЕБНО МЕТОДИЧЕСКОЕ И ИНФОРМАЦИОННОЕ ОБЕСПЕЧЕНИЕ ДИСЦИПЛИНЫ)

8.1 Перечень учебной литературы, необходимой для освоения дисциплины

Основная литература:

1. Карпеев, Д.О. Объектно-ориентированное программирование на языке С++ [Электронный ресурс]: Учеб. пособие / Д. О. Карпеев, В. А. Транин. - Электрон. текстовые, граф. дан. (630 Кб). - Воронеж: ГОУВПО "Воронежский государственный технический университет", 2011. – 1 файл. – 30-00.

2. Молдованова, О.В. Языки программирования и методы трансляции [Электронный ресурс]: учебное пособие/ Молдованова О.В. — Электрон. текстовые данные. — Новосибирск: Сибирский государственный университет телекоммуникаций и информатики, 2012. – 134 c. – Режим доступа: http://www.iprbookshop.ru/54809.html. — ЭБС «IPRbooks».

3. Стенли Липпман Язык программирования С++ [Электронный ресурс]: полное руководство/ Стенли Липпман, Жози Лажойе – Электрон. текстовые данные. - Саратов: Профобразование, 2014. - 1104 c. – Режим доступа: http://www.iprbookshop.ru/63964.html. — ЭБС «IPRbooks».

Дополнительная литература:

4. Основы объектно-ориентированного программирования [Электронный ресурс]: учеб. пособие / А. Г. Остапенко [и др.]. - Воронеж: ВГТУ, 2001. - 110 с. - 20.00.

5. Меньшиков, Ф. Олимпиадные задачи по программированию / Ф. Меньшиков. - СПб.: Питер, 2006. - 315 с.: ил. - ISBN 5-469-00765-0: 175-00**.**

6. Стивенсон, Б. Python : сборник упражнений / Б. Стивенсон ; перевод А. Ю. Гинько. — Москва : ДМК Пресс, 2021. — 238 с.

7. Ермаков А.В. Объектно-ориентированное программирование в задачах на языке Java : учебное пособие / Ермаков А.В.. — Саратов : Саратовский государственный технический университет имени Ю.А. Гагарина, ЭБС АСВ, 2022. — 156 с.

8. Методические указания и задания на курсовую работу по дисциплине Теория языков программирования [Электронный ресурс]/ – Электрон. текстовые данные. – М.: Московский технический университет связи и информатики, 2016. – 20 c. – Режим доступа: http: //www. iprbookshop. ru/61767. html. — ЭБС «IPRbooks».

9. Морозова Ю.В. Практикум по объектно-ориентированному

программированию : учебное пособие / Морозова Ю.В.. — Томск : Томский государственный университет систем управления и радиоэлектроники, 2021. — 186 с. – Режим доступа: https://www.iprbookshop.ru/152837.html. — ЭБС «IPRbooks».

8.2 Перечень информационных технологий, используемых при осуществлении образовательного процесса по дисциплине, включая перечень лицензионного программного обеспечения, ресурсов информационно-телекоммуникационной сети «Интернет», современных профессиональных баз данных и информационных справочных систем:

1. Среды разработки: MS Visual Studio Community, VS Code, JetBrains CLion, Intellij IDEA, Pycharm community edition.
2. Система контроля версий Git.
3. Компиляторы: GCC (MinGW) или MSVC, NASM.

9 МАТЕРИАЛЬНО-ТЕХНИЧЕСКАЯ БАЗА, НЕОБХОДИМАЯ ДЛЯ ОСУЩЕСТВЛЕНИЯ ОБРАЗОВАТЕЛЬНОГО ПРОЦЕССА

Специализированная лекционная аудитория, оснащенная оборудованием для лекционных демонстраций и проекционной аппаратурой.

Дисплейный класс, оснащенный компьютерными программами для проведения лабораторного практикума.

10. МЕТОДИЧЕСКИЕ УКАЗАНИЯ ДЛЯ ОБУЧАЮЩИХСЯ ПО ОСВОЕНИЮ ДИСЦИПЛИНЫ (МОДУЛЯ)

По дисциплине «Языки программирования» читаются лекции, проводятся лабораторные работы, выполняется курсовой проект.

Основой изучения дисциплины являются лекции, на которых излагаются наиболее существенные и трудные вопросы, а также вопросы, не нашедшие отражения в учебной литературе.

Лабораторные работы выполняются на учебных рабочих местах в дисплейных классах в соответствии с методиками, приведенными в указаниях к выполнению работ.

Методика выполнения курсового проекта изложена в соответствующих методических рекомендациях. Выполнять этапы курсового проекта должны своевременно и в установленные сроки.

Контроль усвоения материала дисциплины производится защитой лабораторных работ, защитой курсового проекта.

Вид учебных занятий	Деятельность студента
Лекция	Написание конспекта лекций: кратко, схематично, последовательно фиксировать основные положения, выводы, формулировки, обобщения; помечать важные мысли, выделять ключевые слова, термины. Проверка терминов, понятий с помощью энциклопедий, словарей, справочников с выписыванием толкований в тетрадь. Обозначение вопросов, терминов, материала, которые вызывают трудности, поиск ответов в

	рекомендуемой литературе. Если самостоятельно не удается разобраться в материале, необходимо сформулировать вопрос и задать преподавателю на лекции или на практическом занятии.
Лабораторная работа	Лабораторные работы позволяют научиться применять теоретические знания, полученные на лекции при решении конкретных задач. Чтобы наиболее рационально и полно использовать все возможности лабораторных для подготовки к ним необходимо: следует разобрать лекцию по соответствующей теме, ознакомится с соответствующим разделом учебника, проработать дополнительную литературу и источники, решить задачи и выполнить другие письменные задания.
Самостоятельная работа	Самостоятельная работа студентов способствует глубокому усвоения учебного материала и развитию навыков самообразования. Самостоятельная работа предполагает следующие составляющие: - работа с текстами: учебниками, справочниками, дополнительной литературой, а также проработка конспектов лекций; - выполнение домашних заданий и расчетов; - работа над темами для самостоятельного изучения; - участие в работе студенческих научных конференций, олимпиад; - подготовка к промежуточной аттестации.
Подготовка к промежуточной аттестации	Готовиться к промежуточной аттестации следует систематически, в течение всего семестра. Интенсивная подготовка должна начаться не позднее, чем за месяц-полтора до промежуточной аттестации. Данные перед зачетом с оценкой, зачетом с оценкой, экзаменом три дня эффективнее всего использовать для повторения и систематизации материала.

ЛИСТ РЕГИСТРАЦИИ ИЗМЕНЕНИЙ

№ п/п	Перечень вносимых изменений	Дата внесения изменений	Подпись заведующего кафедрой, ответственного за реализацию ОПОП