

## Рабочая программа дисциплины

# Основы алгоритмизации

Закреплена за подразделением

Кафедра промышленного менеджмента

Направление подготовки

01.03.05 СТАТИСТИКА

Квалификация

Бакалавр

Форма обучения

очная

Общая трудоемкость

5 ЗЕТ

Часов по учебному плану

180

Формы контроля:

экзамен 6

в том числе:

аудиторные занятия

51

самостоятельная работа

93

часов на контроль

36

### Распределение часов дисциплины по семестрам

| Семестр<br>( <b>&lt;Курс&gt;.&lt;Семестр на<br/>курсе&gt;</b> ) | <b>6 (3.2)</b> |     | Итого |     |
|---|----------------|-----|-------|-----|
| Неделя  | 17 2/6         |     |       |     |
| Вид занятий   | УП             | РП  | УП    | РП  |
| Практические  | 51             | 51  | 51    | 51  |
| Итого ауд.  | 51             | 51  | 51    | 51  |
| Контактная работа   | 51             | 51  | 51    | 51  |
| Сам. работа   | 93             | 93  | 93    | 93  |
| В том числе сам. работа в<br>рамках ФОС                         |                |     |       |     |
| Часы на контроль  | 36             | 36  | 36    | 36  |
| Итого   | 180            | 180 | 180   | 180 |

Программу составил(и):

-, *ст.преп., Богачев Андрей Сергеевич*

Рабочая программа дисциплины

**Основы алгоритмизации**

Разработана в соответствии с ОС ВО НИТУ МИСИС, приказ № 796 о.в. от 10.12.2025.

Составлена на основании учебного плана:

01.03.05 СТАТИСТИКА, 01.03.05-БСТ-26.plx, утвержденного Ученым советом НИТУ МИСИС в составе соответствующей ОПОП ВО 20.11.2025, протокол № 9-25.

Утверждена в составе ОПОП ВО:

01.03.05 СТАТИСТИКА, утвержденной Ученым советом НИТУ МИСИС 20.11.2025, протокол № 9-25.

Рабочая программа одобрена на заседании

**Кафедры промышленного менеджмента**

Протокол от 21.01.2025 г., №5.

Руководитель подразделения Костюхин Юрий Юрьевич, д.э.н., доцент.

**1. ЦЕЛИ ОСВОЕНИЯ**

|     |   |
|-----|---|
| 1.1 | Целью освоения дисциплины является формирование у обучающихся фундаментальных навыков алгоритмического мышления и основ программирования, необходимых для решения вычислительных и логических задач с использованием современных информационных технологий. |
|-----|---|

**2. МЕСТО В СТРУКТУРЕ ОБРАЗОВАТЕЛЬНОЙ ПРОГРАММЫ**

|            |   |      |
|------------|---|------|
| Блок ОП:   |   | Б1.В |
| <b>2.1</b> | <b>Требования к предварительной подготовке обучающегося:</b>  |      |
| 2.1.1      | Основы информационных технологий  |      |
| 2.1.2      | Основы сбора, хранения и управления данными   |      |
| 2.1.3      | Excel для анализа данных  |      |
| 2.1.4      | Продвинутый Excel   |      |
| <b>2.2</b> | <b>Дисциплины (модули) и практики, для которых освоение данной дисциплины (модуля) необходимо как предшествующее:</b> |      |
| 2.2.1      | Подготовка к процедуре защиты и защита выпускной квалификационной работы  |      |
| 2.2.2      | Основы интеграции массивов данных   |      |

**3. РЕЗУЛЬТАТЫ ОБУЧЕНИЯ ПО ДИСЦИПЛИНЕ, СООТНЕСЕННЫЕ С ФОРМИРУЕМЫМИ КОМПЕТЕНЦИЯМИ**

**ОПК-4: Способен понимать принципы работы современных информационных технологий и использовать их для решения задач профессиональной деятельности**

**Знать:**

ОПК-4-31 Понятие алгоритма, его свойства и способы описания (словесный, блок-схемы, псевдокод).

Базовые алгоритмические конструкции: следование, ветвление, цикл.

Основные типы данных (целые, вещественные, символьные, логические) и операции над ними.

Базовые структуры данных: массивы (списки), строки.

Принципы процедурного и структурного программирования.

Основы оценки сложности алгоритмов (O-нотация).

**Уметь:**

ОПК-4-У1 Разрабатывать пошаговые алгоритмы для решения типовых вычислительных и логических задач.

Представлять алгоритмы в виде блок-схем и псевдокода.

Реализовывать разработанные алгоритмы на языке программирования высокого уровня (например, Python).

Проводить отладку и тестирование программ.

Применять базовые алгоритмы сортировки и поиска.

**Владеть:**

ОПК-4-В1 Навыками алгоритмического мышления для декомпозиции задач.

Синтаксисом языка программирования Python для реализации алгоритмов.

Инструментами разработки и отладки программ (IDE, отладчик).

Методами решения задач обработки данных с использованием массивов и циклов.

**4. СТРУКТУРА И СОДЕРЖАНИЕ**

| Код занятия | Наименование разделов и тем /вид занятия/ | Семестр / Курс | Часов | Формируемые индикаторы компетенций | Литература и эл. ресурсы | Примечание | КМ | Выполняемые работы |
|-------------|---|----------------|-------|------------------------------------|--------------------------|------------|----|--------------------|
|-------------|---|----------------|-------|------------------------------------|--------------------------|------------|----|--------------------|

|     |  |   |    |                            |                   |  |  |  |
|-----|--|---|----|----------------------------|-------------------|--|--|--|
|     | <b>Раздел 1. Раздел 1. Введение в алгоритмизацию и программирование</b>  |   |    |                            |                   |  |  |  |
| 1.1 | Тема 1.1: Понятие алгоритма, его свойства. Способы записи алгоритмов. Настройка рабочей среды (Python, IDE).<br>Тема 1.2: Первая программа. Переменные. Основные типы данных: int, float, str, bool. Операции ввода- вывода. /Пр/  | 6 | 6  | ОПК-4-31 ОПК-4-У1 ОПК-4-В1 | Л1.1 Л1.2 Л1.3 Э1 |  |  |  |
| 1.2 | Решение простых задач на ввод, обработку и вывод данных. Установка и настройка ПО. /Ср/  | 6 | 16 | ОПК-4-31 ОПК-4-У1 ОПК-4-В1 | Л1.1 Л1.2 Л1.3 Э1 |  |  |  |
|     | <b>Раздел 2. Раздел 2. Базовые алгоритмические конструкции</b>   |   |    |                            |                   |  |  |  |
| 2.1 | Тема 2.1: Линейные алгоритмы. Арифметические операции.<br>Тема 2.2: Условный оператор if-elif-else. Логические операции. Разветвляющиеся алгоритмы.<br>Тема 2.3: Цикл while. Циклы с условием.<br>Тема 2.4: Цикл for. Циклы со счетчиком. Операторы break и continue. /Пр/   | 6 | 12 | ОПК-4-31 ОПК-4-У1 ОПК-4-В1 | Л1.1 Л1.2 Л1.3 Э1 |  |  |  |
| 2.2 | Решение задач на использование условных операторов и циклов (например, нахождение максимума, суммы ряда, проверка на простоту). /Ср/   | 6 | 16 | ОПК-4-31 ОПК-4-У1 ОПК-4-В1 | Л1.1 Л1.2 Л1.3 Э1 |  |  |  |
|     | <b>Раздел 3. Раздел 3. Структуры данных: списки и строки</b>   |   |    |                            |                   |  |  |  |
| 3.1 | Тема 3.1: Списки (массивы) в Python. Создание, индексация, срезы.<br>Тема 3.2: Основные методы списков. Обход элементов списка с помощью цикла for.<br>Тема 3.3: Обработка списков: поиск элемента, нахождение суммы, среднего, минимума/максимума.<br>Тема 3.4: Строки как последовательности символов. Основные строковые методы. /Пр/ | 6 | 12 | ОПК-4-31 ОПК-4-У1 ОПК-4-В1 | Л1.1 Л1.2 Л1.3 Э1 |  |  |  |

|     |   |   |    |                            |                         |  |     |    |
|-----|---|---|----|----------------------------|-------------------------|--|-----|----|
| 3.2 | Решение задач на обработку одномерных массивов (списков). Работа со строками.<br><br>/Ср/   | 6 | 16 | ОПК-4-31 ОПК-4-У1 ОПК-4-В1 | Л1.1 Л1.2<br>Л1.3<br>Э1 |  |     |    |
|     | <b>Раздел 4. Раздел 4. Функции и модули</b>   |   |    |                            |                         |  |     |    |
| 4.1 | Тема 4.1: Функции. Определение и вызов функции. Параметры и возвращаемое значение. Тема 4.2: Области видимости переменных. Локальные и глобальные переменные. Тема 4.3: Использование стандартных модулей (например, math, random).<br>/Пр/ | 6 | 9  | ОПК-4-31 ОПК-4-У1 ОПК-4-В1 | Л1.1 Л1.2<br>Л1.3<br>Э1 |  | КМ1 | Р1 |
| 4.2 | Декомпозиция ранее решенных задач с использованием функций. Создание собственного модуля с набором функций.<br>/Ср/   | 6 | 16 | ОПК-4-31 ОПК-4-У1 ОПК-4-В1 | Л1.1 Л1.2<br>Л1.3<br>Э1 |  |     |    |
|     | <b>Раздел 5. Раздел 5. Алгоритмы сортировки и поиска</b>  |   |    |                            |                         |  |     |    |
| 5.1 | Тема 5.1: Понятие сложности алгоритма. О- нотация (вводное). Тема 5.2: Алгоритмы сортировки: пузырьковая сортировка, сортировка выбором. Тема 5.3: Алгоритмы поиска: линейный поиск, бинарный поиск. /Пр/                                   | 6 | 9  | ОПК-4-31 ОПК-4-У1 ОПК-4-В1 | Л1.1 Л1.2<br>Л1.3<br>Э1 |  | КМ2 | Р2 |
| 5.2 | Самостоятельная реализация и тестирование алгоритмов сортировки и поиска. Сравнение их производительности на небольших наборах данных.<br>/Ср/  | 6 | 16 | ОПК-4-31 ОПК-4-У1 ОПК-4-В1 | Л1.1 Л1.2<br>Л1.3<br>Э1 |  |     |    |
|     | <b>Раздел 6. Раздел 6. Работа с файлами</b>   |   |    |                            |                         |  |     |    |
| 6.1 | Тема 6.1: Основы работы с текстовыми файлами: открытие, чтение, запись.<br>/Пр/   | 6 | 3  | ОПК-4-31 ОПК-4-У1 ОПК-4-В1 | Л1.1 Л1.2<br>Л1.3<br>Э1 |  | КМ3 | Р3 |
| 6.2 | Решение задач на чтение данных из файла, их обработку и запись результата в другой файл.<br>/Ср/  | 6 | 13 | ОПК-4-31 ОПК-4-У1 ОПК-4-В1 | Л1.1 Л1.2<br>Л1.3<br>Э1 |  |     |    |
|     | <b>Раздел 7. Подготовка к контрольным мероприятиям и выполняемым работам</b>  |   |    |                            |                         |  |     |    |
| 7.1 | Объем часов самостоятельной работы на подготовку к КМ /Ср/  | 6 | 0  | ОПК-4-31 ОПК-4-У1 ОПК-4-В1 | Л1.1 Л1.2<br>Л1.3<br>Э1 |  |     |    |

|     |  |   |   |                            |                   |  |  |  |
|-----|--|---|---|----------------------------|-------------------|--|--|--|
| 7.2 | Объем часов самостоятельной работы на подготовку к ВР /Ср/ | 6 | 0 | ОПК-4-31 ОПК-4-У1 ОПК-4-В1 | Л1.1 Л1.2 Л1.3 Э1 |  |  |  |
|-----|--|---|---|----------------------------|-------------------|--|--|--|

## 5. ФОНД ОЦЕНОЧНЫХ МАТЕРИАЛОВ

### 5.1. Контрольные мероприятия (контрольная работа, тест, коллоквиум, экзамен и т.п), вопросы для самостоятельной подготовки

| Код КМ | Контрольное мероприятие                    | Проверяемые индикаторы компетенций | Вопросы для подготовки   |
|--------|--|------------------------------------|--|
| КМ1    | Контрольная работа №1: Базовые конструкции | ОПК-4-31;ОПК-4-У1;ОПК-4-В1         | <p>Что такое переменная? Типы данных в Python.</p> <p>Как работает условный оператор if?</p> <p>В чем разница между циклом while и for?</p> <p>Как получить данные от пользователя и вывести результат?</p> <p>Основные арифметические и логические операции.</p> <p>Нарисуйте блок-схему для алгоритма нахождения большего из двух чисел.</p> |
| КМ2    | Контрольная работа №2: Списки и функции    | ОПК-4-31;ОПК-4-У1;ОПК-4-В1         | <p>Как создать список и обратиться к его элементу по индексу?</p> <p>Как перебрать все элементы списка с помощью цикла?</p> <p>Что такое функция и для чего она нужна?</p> <p>Как передать параметры в функцию и вернуть результат?</p> <p>Что такое срез списка?</p> <p>Как работать со строками?</p>   |
| КМ3    | Контрольная работа №3: Сортировка и поиск  | ОПК-4-31;ОПК-4-У1;ОПК-4-В1         | <p>Опишите алгоритм пузырьковой сортировки.</p> <p>Опишите алгоритм линейного поиска.</p> <p>В чем преимущество бинарного поиска перед линейным? Какое у него ограничение?</p> <p>Что такое сложность алгоритма (общее понятие)?</p> <p>Как открыть файл на чтение в Python?</p> <p>Как прочитать данные из файла построчно?</p>               |

### 5.2. Перечень работ, выполняемых по дисциплине (Курсовая работа, Курсовой проект, РГР, Реферат, ЛР, ПР и т.п.)

| Код работы | Название работы   | Проверяемые индикаторы компетенций | Содержание работы  |
|------------|---|------------------------------------|--|
| P1         | Практическая работа №1: Реализация циклических алгоритмов | ОПК-4-31;ОПК-4-У1;ОПК-4-В1         | Написать программу, которая вычисляет сумму или произведение членов заданного ряда (например, факториал, сумму степеней) с использованием циклов while и for.                  |
| P2         | Практическая работа №2: Обработка одномерного массива     | ОПК-4-У1;ОПК-4-31;ОПК-4-В1         | Содержание работы: Написать программу, которая для заданного списка чисел находит количество элементов, их сумму, среднее арифметическое, максимальный и минимальный элементы. |

|    |   |                            |   |
|----|---|----------------------------|---|
| РЗ | Практическая работа №3: Разработка программы с использованием функций | ОПК-4-В1;ОПК-4-31;ОПК-4-У1 | Содержание работы: Разработать программу, которая представляет пользователю меню с выбором арифметической операции. Каждая операция должна быть реализована в виде отдельной функции. |
|----|---|----------------------------|---|

### 5.3. Оценочные материалы, используемые для экзамена (билеты, тесты и т.п.)

1. Дайте определение алгоритма и объясните его основные свойства: дискретность, определённости, результативность и массовость. Раскройте различие между алгоритмом и программой в контексте их отношения к компьютеру. Почему понимание фундаментальных свойств алгоритма необходимо для эффективного решения вычислительных задач?
2. Объясните различие между словесным описанием, блок-схемой и псевдокодом при представлении алгоритмов. Укажите преимущества и ограничения каждого метода представления. Приведите пример одного алгоритма, представленного всеми тремя способами, и объясните преимущества каждого представления.
3. Дайте определение блок-схемы и объясните обозначения основных элементов: овал для начала и конца, прямоугольник для процесса, ромб для условия. Раскройте правила построения блок-схемы, включая правильное направление потока управления. Почему блок-схемы остаются полезным инструментом при визуализации сложных алгоритмов несмотря на развитие других методов?
4. Объясните понятие псевдокода и укажите, как он занимает промежуточное положение между естественным языком и языком программирования. Раскройте правила написания псевдокода, позволяющие сохранять его независимостью от конкретного языка программирования. Почему использование псевдокода часто является первым шагом при проектировании алгоритма?
5. Дайте определение последовательности (следования) как базовой алгоритмической конструкции и объясните, как серия операций выполняется одна за другой. Укажите, как порядок операций влияет на результат алгоритма. Приведите примеры задач, где изменение последовательности операций приводит к неправильному результату.
6. Объясните структуру условного оператора (ветвления) и укажите различие между полным и неполным условиями. Раскройте применение вложенных условных операторов при решении сложных логических задач. Почему правильное использование условий критично для создания корректных алгоритмов?
7. Дайте определение цикла как базовой алгоритмической конструкции и объясните, как циклы позволяют повторять набор операций многократно. Укажите различие между циклами "пока" (while), "для" (for) и циклами с постусловием. Приведите примеры задач, где использование циклов значительно упрощает решение.
8. Объясните механизм работы цикла "пока" и укажите условие его завершения. Раскройте опасность бесконечных циклов и способы их предотвращения. Почему правильная формулировка условия выхода из цикла является критически важной?
9. Дайте определение цикла "для" и объясните, как счётчик цикла управляет количеством повторений. Укажите различие между циклом "для" и циклом "пока" с точки зрения удобства использования. Приведите примеры задач, где цикл "для" является предпочтительным выбором.
10. Объясните понятие вложенных циклов и укажите, как количество вложений влияет на количество итераций. Раскройте применение вложенных циклов при обработке двумерных массивов. Почему анализ сложности вложенных циклов требует особого внимания?
11. Дайте определение целого типа данных и объясните, какой диапазон значений он может хранить. Укажите различие между знаковыми и беззнаковыми целыми числами. Почему выбор типа целых чисел влияет на диапазон допустимых значений и использование памяти?
12. Объясните вещественный тип данных и укажите, как компьютер представляет дробные числа в двоичной форме. Раскройте концепцию точности и переполнения при работе с вещественными числами. Приведите примеры проблем, возникающих из-за ограниченной точности вещественной арифметики.
13. Дайте определение логического типа данных и объясните, как логические значения (истина, ложь) используются при управлении потоком программы. Укажите применение логических операций: конъюнкция (И), дизъюнкция (ИЛИ), отрицание (НЕ). Почему логические типы являются основой условных выражений?
14. Объясните символьный тип данных и укажите, как символы кодируются в компьютере. Раскройте различие между символом, строкой и текстом. Приведите примеры операций над символами, включая преобразование между символами и числовыми кодами.
15. Дайте определение строкового типа данных и объясните, как строки хранятся и обрабатываются в памяти компьютера. Укажите основные операции над строками: конкатенация, поиск подстроки, замена. Почему понимание внутреннего представления строк важно для оптимизации алгоритмов?

16. Объясните понятие типа данных и укажите, почему явное указание типа необходимо для корректной работы алгоритма. Раскройте концепцию преобразования типов и потенциальные проблемы при автоматическом преобразовании. Приведите примеры ошибок, возникающих из-за неправильного использования типов данных.
17. Дайте определение массива (списка) и объясните, как элементы хранятся в последовательных ячейках памяти. Укажите различие между индексированием массива и доступом к элементам по значению. Почему массивы являются одной из наиболее часто используемых структур данных?
18. Объясните механизм индексирования массивов и укажите, почему нумерация начинается с нуля в большинстве языков программирования. Раскройте опасность выхода за границы массива (buffer overflow) и способы предотвращения. Приведите примеры алгоритмов, работающих с массивами.
19. Дайте определение двумерного массива (матрицы) и объясните, как элементы организованы в строках и столбцах. Укажите различие между хранением матриц по строкам и по столбцам в памяти. Почему работа с двумерными массивами требует вложенных циклов?
20. Объясните применение массивов при решении задач обработки данных. Раскройте преимущества использования массивов по сравнению с отдельными переменными. Приведите примеры практических задач, где использование массивов существенно упрощает решение.
21. Дайте определение функции в программировании и объясните, почему разделение кода на функции улучшает его читаемость и переиспользуемость. Укажите различие между формальными параметрами функции и фактическими аргументами. Почему структурированное программирование требует разумного использования функций?
22. Объясните механизм вызова функции и возврата управления в точку вызова. Раскройте понятие локальных переменных и области видимости. Приведите примеры того, как изменения локальных переменных внутри функции не влияют на переменные вне функции.
23. Дайте определение рекурсии и объясните, как функция может вызывать саму себя. Укажите необходимое условие для предотвращения бесконечной рекурсии: наличие базового случая. Приведите классические примеры рекурсивных алгоритмов: факториал, числа Фибоначчи, поиск в дереве.
24. Объясните различие между прямой рекурсией (функция вызывает саму себя) и косвенной рекурсией (функции вызывают друг друга). Раскройте роль стека вызовов в управлении рекурсией. Почему глубокая рекурсия может привести к переполнению стека?
25. Дайте определение модуля в контексте структурного программирования и объясните, как модули упрощают разработку больших программ. Укажите различие между импортированием модуля и использованием его функций. Почему модульное программирование способствует повторное использованию кода?
26. Объясните понятие пространства имён и укажите, как модули помогают избежать конфликтов имён. Раскройте различие между глобальными и локальными пространствами имён. Приведите примеры того, как правильное использование пространств имён улучшает организацию кода.
27. Дайте определение алгоритма поиска и объясните основные виды: линейный поиск и бинарный поиск. Укажите различие между поиском в неупорядоченном и упорядоченном массиве. Почему выбор алгоритма поиска зависит от структуры данных?
28. Объясните алгоритм линейного поиска и укажите его временную сложность. Раскройте простоту реализации линейного поиска и его применимость к любым массивам. Приведите примеры практических задач, где линейный поиск является приемлемым решением.
29. Дайте определение алгоритма бинарного поиска и объясните, как разделение на половины ускоряет поиск. Укажите необходимое условие для применения бинарного поиска: массив должен быть отсортирован. Почему бинарный поиск значительно более эффективен, чем линейный поиск для больших массивов?
30. Объясните процесс реализации бинарного поиска и укажите временную сложность  $O(\log n)$ . Раскройте роль переменных left, right и middle при поиске элемента. Приведите примеры расчёта количества итераций при поиске в массиве различных размеров.
31. Дайте определение алгоритма сортировки и объясните, почему сортировка данных является одной из наиболее важных операций в программировании. Укажите различие между сортировкой по возрастанию и убыванию. Почему эффективность алгоритма сортировки существенно влияет на общую производительность системы?
32. Объясните алгоритм пузырьковой сортировки (Bubble Sort) и укажите его временную сложность  $O(n^2)$ . Раскройте механизм сравнения соседних элементов и их обмена. Почему пузырьковая сортировка редко используется в практических



33. Дайте определение алгоритма сортировки выбором (Selection Sort) и объясните, как он находит минимальный элемент на каждом проходе. Укажите временную сложность Selection Sort. Приведите примеры сравнения Selection Sort с другими алгоритмами сортировки.
34. Объясните алгоритм сортировки вставками (Insertion Sort) и укажите его применимость к частично отсортированным массивам. Раскройте временную сложность в среднем случае  $O(n^2)$  и лучшем случае  $O(n)$ . Почему Insertion Sort часто используется для сортировки небольших массивов?
35. Дайте определение алгоритма быстрой сортировки (Quick Sort) и объясните метод "разделяй и властвуй". Укажите роль опорного элемента (pivot) в разделении массива. Почему Quick Sort считается одним из наиболее эффективных алгоритмов сортировки на практике?
36. Объясните алгоритм сортировки слиянием (Merge Sort) и укажите его временную сложность  $O(n \log n)$  в худшем случае. Раскройте процесс разделения массива на половины и последующего слияния. Почему Merge Sort гарантирует  $O(n \log n)$  сложность и стабильность сортировки?
37. Дайте определение O-нотации (Big O notation) и объясните, как она описывает рост времени выполнения алгоритма. Укажите различие между  $O(1)$ ,  $O(n)$ ,  $O(n^2)$ ,  $O(\log n)$  и  $O(n \log n)$ . Почему анализ сложности алгоритма необходим при выборе оптимального решения?
38. Объясните различие между временной сложностью (time complexity) и пространственной сложностью (space complexity). Раскройте понятия лучшего, среднего и худшего случаев при анализе сложности. Приведите примеры алгоритмов с различной сложностью.
39. Дайте определение файла в компьютерной системе и объясните различие между текстовыми и двоичными файлами. Укажите основные операции над файлами: открытие, чтение, запись, закрытие. Почему правильное управление файлами критично для безопасности данных?
40. Объясните процесс открытия файла и укажите различие между режимами: чтение (read), запись (write) и добавление (append). Раскройте роль указателя файла и его движения при последовательном чтении. Приведите примеры программ, работающих с файлами на диске.
41. Дайте определение структурного программирования и объясните основные принципы: использование последовательности, ветвления и циклов. Укажите преимущества структурного подхода по сравнению с неструктурированным кодом. Почему структурное программирование и сегодня остаётся основой для написания чистого кода?
42. Объясните понятие процедурного программирования и укажите, как функции (процедуры) упрощают разработку программ. Раскройте различие между процедурным и объектно-ориентированным подходом. Приведите примеры задач, где процедурный подход остаётся наиболее эффективным выбором.
43. Дайте определение отладки программ и объясните основные методы: использование точек останова, пошаговое выполнение, просмотр значений переменных. Укажите роль отладчика (debugger) в современных интегрированных средах разработки. Почему навыки отладки являются критически важными для программистов?
44. Объясните процесс тестирования программ и укажите различие между модульным тестированием и интеграционным тестированием. Раскройте концепцию граничных случаев и их значение при тестировании алгоритмов. Почему комплексное тестирование необходимо для обеспечения надёжности программ?
45. Дайте определение оптимизации алгоритмов и объясните, как различные подходы влияют на производительность программ. Укажите типовые техники оптимизации: кэширование результатов, использование более эффективных структур данных, параллельная обработка. Почему преждевременная оптимизация часто приводит к более сложному и менее понятному коду?

**5.4. Методика оценки освоения дисциплины (модуля, практики. НИР)**

Предполагается следующая шкала оценок:

- а) «отлично» (90 баллов и выше) – студент показывает глубокие, исчерпывающие знания в объеме пройденной программы, уверенно действует по применению полученных знаний на практике, твердые и достаточно полные знания в объеме пройденной программы, грамотно и логически стройно излагает материал при ответе, умеет формулировать выводы из изложенного теоретического материала, знает дополнительно рекомендованную литературу;
- б) «хорошо» (75 - 90 баллов) – студент допускает незначительные ошибки при освещении заданных вопросов, правильно действует по применению знаний на практике, четко излагает материал;
- в) «удовлетворительно» (51 - 74 балла) – студент показывает знания в объеме пройденной программы, ответы излагает хотя и с ошибками, но уверенно исправляемыми после дополнительных и наводящих вопросов, правильно действует по применению знаний на практике;
- г) «неудовлетворительно» (50 баллов и ниже) – студент допускает грубые ошибки в ответе, не понимает сущности излагаемого вопроса, не умеет применять знания на практике, дает неполные ответы на дополнительные и наводящие вопросы.

Допуск к экзамену осуществляется на основании выполненных контрольных мероприятий. Оценка за дисциплину выставляется по итогам результатов экзамена.

**6. УЧЕБНО-МЕТОДИЧЕСКОЕ И ИНФОРМАЦИОННОЕ ОБЕСПЕЧЕНИЕ****6.1. Рекомендуемая литература****6.1.1. Основная литература**

|      | Авторы, составители             | Заглавие  | Библиотека             | Издательство, год   |
|------|---------------------------------|---|------------------------|---|
| Л1.1 | Буйначев С. К.,<br>Боклаг Н. Ю. | Основы программирования на языке Python: учебное пособие  | Электронная библиотека | Екатеринбург: Издательство Уральского университета, 2014      |
| Л1.2 | Северенс Ч.                     | Введение в программирование на Python: учебное пособие  | Электронная библиотека | Москва: Национальный Открытый Университет «ИНТУИТ», 2016      |
| Л1.3 | Шелудько В. М.                  | Язык программирования высокого уровня Python: функции, структуры данных, дополнительные модули: учебное пособие | Электронная библиотека | Ростов-на-Дону, Таганрог: Южный федеральный университет, 2017 |

**6.2. Перечень ресурсов информационно-телекоммуникационной сети «Интернет»**

|    |           |   |
|----|-----------|---|
| Э1 | LMS MISIS | <a href="https://lk.misis.ru/ru/sep/">https://lk.misis.ru/ru/sep/</a> |
|----|-----------|---|

**6.3 Перечень программного обеспечения**

|     |                  |
|-----|------------------|
| П.1 | Microsoft Office |
| П.2 | MS Teams         |
| П.3 | LMS Moodle       |

**6.4. Перечень информационных справочных систем и профессиональных баз данных**

|     |   |
|-----|---|
| И.1 |   |
| И.2 | LeetCode: <a href="https://leetcode.com">https://leetcode.com</a>       |
| И.3 | Codeforces: <a href="https://codeforces.com">https://codeforces.com</a> |

**7. МАТЕРИАЛЬНО-ТЕХНИЧЕСКОЕ ОБЕСПЕЧЕНИЕ**

| Ауд.                  | Назначение                           | Оснащение   |
|-----------------------|--------------------------------------|---|
| Б-1102                | Компьютерный класс                   | Комплект учебной мебели на 30 рабочих мест, моноблоки для студентов (20 шт.), 1 маркерная доска, телевизор для презентаций, рабочее место для преподавателя с моноблоком (1 шт). Цифровой флипчарт (передвижной).                 |
| Б-1104                | Компьютерный класс                   | Комплект учебной мебели на 30 рабочих мест, моноблоки для студентов (20 шт.), 1 маркерная доска, Телевизор для презентаций, рабочее место для преподавателя с моноблоком (1 шт).  |
| Б-1117                | Учебная аудитория                    | комплект учебной мебели на 42 рабочих мест, 1 компьютер для преподавателя, проектор + мультимедийный экран, 1 маркерная доска   |
| Б-1134                | Учебная аудитория (лекторий)         | Комплект учебной мебели на 128 рабочих мест, проектор, экран, 1 Цифровой флипчарт (передвижной).  |
| Читальный зал № 3 (Б) | Аудитория для самостоятельной работы | Комплект учебной мебели на 44 места для обучающихся, МФУ Xerox VersaLink B7025 с функцией масштабирования текстов и изображений, 8 ПК с доступом к ИТС «Интернет», ЭИОС университета через личный кабинет на платформе LMS Moodle |

#### **8. МЕТОДИЧЕСКИЕ УКАЗАНИЯ ДЛЯ ОБУЧАЮЩИХСЯ**

1. Лекции и практические занятия проводятся с использованием компьютерной презентационной программы PowerPoint.
2. Практические занятия проводятся с использованием кейсовых ситуаций.
3. Текущий контроль, контрольные работы и зачет проводятся на основе использования специальных компьютерных программ тестирования знаний навыков и умений студентов.
4. Для самостоятельной работы и текущего контроля в системе «смешанного обучения» студенты используют специальные базы данных (электронные учебники) в среде LMS Moodle по разработанным траекториям.
5. Консультации по курсу проводятся с использованием e-mail и среды LMS Moodle
6. Текущий контроль проводится в электронной форме на компьютерах в центре тестирования кафедры.