


МИНОБРНАУКИ РОССИИ
ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ
ВЫСШЕГО ОБРАЗОВАНИЯ
«ВОРОНЕЖСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ»
(ФГБОУ ВО «ВГУ»)

УТВЕРЖДАЮ

Заведующий кафедрой
Программирования и информационных технологий


_____ проф. Махортов С.Д.,
03.07.2023

РАБОЧАЯ ПРОГРАММА УЧЕБНОЙ ДИСЦИПЛИНЫ

Б1.О.16 Алгоритмы и структуры данных

1. Код и наименование направления подготовки/специальности:

09.03.02 Информационные системы и технологии

2. Профиль подготовки/специализация:

Информационные технологии в цифровом дизайне

3. Квалификация (степень) выпускника: Бакалавр

4. Форма обучения: Очная

5. Кафедра, отвечающая за реализацию дисциплины:

Программирования и информационных технологий

6. Составители программы:

ст. преподаватель каф. ПиИТ Соломатин Дмитрий Иванович
e-mail: solomatin@cs.vsu.ru
факультет: Компьютерных наук
кафедра: Программирования и информационных технологий

7. Рекомендована:

НМС ф-та компьютерных наук, протокол № 7 от 03.07.2023

8. Учебный год: 2023-2024

Семестр(ы): 2

9. Цели и задачи учебной дисциплины:

Изучение студентами классических структур данных (связные списки, различные виды деревьев, хеш-таблицы, графы) и алгоритмов, которые лежат в их основе или используют данные структуры, развитие базовых навыков проектирования и анализа алгоритмов, а также применения изученных алгоритмов и структур данных в решении практических задач.

10. Место учебной дисциплины в структуре ООП:

Учебная дисциплина относится к обязательной части блока Б1.

Для успешного освоения дисциплины необходимы знание одного из современных языков программирования и практические навыки составления программ.

11. Планируемые результаты обучения по дисциплине/модулю (знания, умения, навыки), соотнесенные с планируемыми результатами освоения образовательной программы (компетенциями выпускников):

| Код | Название компетенции | Код(ы) | Индикатор(ы) | Планируемые результаты обучения |
|-------|-----------------------------------------------------------------------------------------------------------------------------------|---------|------------------------------------------------------------------------------------------------------------------------------------------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| ОПК-6 | Способен разрабатывать алгоритмы и программы, пригодные для практического применения в области информационных систем и технологий | ОПК-6.1 | Знает методы алгоритмизации, языки и технологии программирования, пригодные для практического применения в области информационных систем и технологий | Знать: классические структуры данных (связные списки, различные виды деревьев, хеш-таблицы, графы) и типичные варианты практического применения данных структур в области информационных систем и технологий |
| | | ОПК-6.2 | Умеет применять методы алгоритмизации, языки и технологии программирования при решении профессиональных задач в области информационных систем и технологий | Уметь: применять динамические структуры данных и базовые алгоритмы (поиск, сортировка, перебор и т.п.) при решении профессиональных задач в области информационных систем и технологий |

12. Объем дисциплины в зачетных единицах/час. (в соответствии с уч. планом) – 4 / 144.

Форма промежуточной аттестации – Экзамен

13. Виды учебной работы

| Вид учебной работы | Трудоемкость | | | | |
|---------------------------------------------------------------------|--------------|--------------|----|---|---|
| | Всего | По семестрам | | | |
| | | 2 сем. | – | – | |
| Аудиторные занятия | 66 | 66 | – | – | |
| в том числе: | лекции | 34 | 34 | – | – |
| | практические | 16 | 16 | – | – |
| | лабораторные | 16 | 16 | – | – |
| Самостоятельная работа | 42 | 42 | – | – | |
| в том числе: курсовая работа (проект) | – | – | – | – | |
| Форма промежуточной аттестации (зачет – 0 час. / экзамен – 36 час.) | 36 | 36 | – | – | |

| Вид учебной работы | Трудоемкость | | | |
|--------------------|--------------|--------------|---|---|
| | Всего | По семестрам | | |
| | | 2 сем. | – | – |
| Итого: | 144 | 144 | – | – |

13.1. Содержание дисциплины

| № п/п | Наименование раздела дисциплины | Содержание раздела дисциплины | Реализация раздела дисциплины с помощью онлайн-курса, ЭУМК |
|------------------|------------------------------------------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|------------------------------------------------------------|
| 1. Лекции | | | |
| 1.1 | Введение в предмет | Цели и задачи изучения дисциплины, связь алгоритмов и структур данных | |
| 1.2 | Основы объектно-ориентированного программирования (инкапсуляция) | Принципы объектно ориентированного подхода; понятие класса и его экземпляров; различия между static-функциями и методами классов; классы как расширение концепции типа данных в виде объединения данных и методов их обработки, понятие состояния объекта, примеры; принципы инкапсуляции; понятие наследования и полиморфизма; класс Object и его методы | |
| 1.3 | Понятие сложности алгоритма | Понятие алгоритма, свойства алгоритма, задача оценки и сравнения эффективности алгоритмов; временная и пространственная сложность, асимптотические оценки сложности алгоритмов для различных случаев и их свойства; практические принципы оценки сложности алгоритмов в O-нотации, примеры алгоритмов с различной сложностью | |
| 1.4 | Алгоритмы сортировки: свойства, простые алгоритмы $O(n^2)$ | Свойства алгоритмов сортировки: время (временная сложность), память (пространственная сложность), устойчивость и естественность поведения, внутренние и внешние сортировки; простые сортировки и их свойства: пузырьковая сортировка, сортировка выбором, сортировка вставками; реализация указанных сортировок в обобщенном (generic) виде с параметром в виде функции сравнения элементов; вызов встроенной функции сортировки | |
| 1.5 | «Быстрые» алгоритмы сортировки | Классические «быстрые» ($O(n \cdot \log(n))$) сортировки со сравнением элементов: сортировка слиянием (merge sort), быстрая сортировка (quick sort), пирамидальная сортировка (heap sort); сортировки «без сравнений»: сортировка подсчетом (counting sort), блочная сортировка (bucket sort) | |
| 1.6 | Классификация структур данных, обзор коллекций языка Java | Классификация структур данных по принципу обращения к элементам: записи (классы), массивы, списки, словари, а также стеки и очереди; обзор коллекций Java как реализаций указанных структур данных: List<T> (ArrayList, LinkedList), Map<K, V> (TreeMap, HashMap), Stack<T>, Queue<T>, практические примеры их применения | |
| 1.7 | Односвязные и двусвязные списки, стеки, очереди | Архитектура односвязных и двусвязных списков, основные операции; реализация указанных структур данных в виде классов с использованием инкапсуляции; реализация стека и очереди на основе односвязного списка | |

| № п/п | Наименование раздела дисциплины | Содержание раздела дисциплины | Реализация раздела дисциплины с помощью онлайн-курса, ЭУМК |
|--------------------------------|------------------------------------------------------------------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|------------------------------------------------------------|
| 1.8 | Деревья, двоичные деревья поиска | Организация данных в виде дерева, терминология, обход и способы отображения деревьев, двоичные деревья; двоичные деревья поиска: идея и основные операции, реализация в виде классов, понятие сбалансированности дерева; реализация множества (Set<T>) и словаря (Map<T>) с помощью двоичного дерева поиска | |
| 1.9 | AVL-деревья | AVL-деревья: идея, операции вставки и удаления, балансировка (R-/L-, RL-/LR-повороты), оценка сложности операций | |
| 1.10 | 2-3-, 2-3-4- и красно-черные деревья | 2-3- и 2-3-4- деревья: идея и основные операции; красно-черные деревья как вариант представления 2-3-4-деревьев, основные операции для красно-черных деревьев и оценка сложности | |
| 1.11 | Хеширование (hash-таблицы) | Hash-таблицы как вариант структуры с потенциальным временем доступа по ключу $O(1)$, понятие hash-функции и ее качество, стратегии разрешения коллизий – открытая и закрытая адресации, реализация множества (Set<T>) и словаря (Map<T>) с помощью hash-таблиц, сравнение с деревьями поиска; комбинации hash-таблиц с деревьями поиска против «плохих» hash-функций | |
| 1.12 | Основы теории графов | Графы: основные понятия и определения; способы задания (хранения) графов (матрица смежности, списки смежности) и универсальный интерфейс представления графов для алгоритмов | |
| 1.13 | Обход графов, поиск в глубину, поиск в ширину | Реализация обхода графов (поиска) в глубину и ширину; отображение графов с помощью библиотеки Graphviz | |
| 1.14 | Оптимизационные алгоритмы на графах (кратчайшие пути, минимальное остовное дерево) | Поиск кратчайших путей в графе: алгоритм Дейкстры, алгоритм Беллмана-Форда, алгоритм Флойда-Уоршелла; поиск минимального остовного дерева: алгоритм Прима, алгоритм Краскала | |
| 1.15 | Примеры практических задач на графах | Эйлеров граф, Гамильтонов граф; примеры практических задач, которые сводятся к задачам на графах | |
| 2. Практические занятия | | | |
| 2.1 | Основы объектно-ориентированного программирования (инкапсуляция) | Принципы объектно ориентированного подхода; понятие класса и его экземпляров; различия между static-функциями и методами классов; классы как расширение концепции типа данных в виде объединения данных и методов их обработки, понятие состояния объекта, примеры; принципы инкапсуляции; понятие наследования и полиморфизма; класс Object и его методы | |
| 2.2 | Понятие сложности алгоритма | Понятие алгоритма, свойства алгоритма, задача оценки и сравнения эффективности алгоритмов; временная и пространственная сложность, асимптотические оценки сложности алгоритмов для различных случаев и их свойства; практические принципы оценки сложности алгоритмов в O-нотации, примеры алгоритмов с различной сложностью | |

| № п/п | Наименование раздела дисциплины | Содержание раздела дисциплины | Реализация раздела дисциплины с помощью онлайн-курса, ЭУМК |
|-------|------------------------------------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|------------------------------------------------------------|
| 2.3 | Алгоритмы сортировки: свойства, простые алгоритмы $O(n^2)$ | Свойства алгоритмов сортировки: время (временная сложность), память (пространственная сложность), устойчивость и естественность поведения, внутренние и внешние сортировки; простые сортировки и их свойства: пузырьковая сортировка, сортировка выбором, сортировка вставками; реализация указанных сортировок в обобщенном (generic) виде с параметром в виде функции сравнения элементов; вызов встроенной функции сортировки | |
| 2.4 | «Быстрые» алгоритмы сортировки | Классические «быстрые» ($O(n \cdot \log(n))$) сортировки со сравнением элементов: сортировка слиянием (merge sort), быстрая сортировка (quick sort), пирамидальная сортировка (heap sort); сортировки «без сравнений»: сортировка подсчетом (counting sort), блочная сортировка (bucket sort) | |
| 2.5 | Классификация структур данных, обзор коллекций языка Java | Классификация структур данных по принципу обращения к элементам: записи (классы), массивы, списки, словари, а также стеки и очереди; обзор коллекций Java как реализаций указанных структур данных: List<T> (ArrayList, LinkedList), Map<K, V> (TreeMap, HashMap), Stack<T>, Queue<T>, практические примеры их применения | |
| 2.6 | Односвязные и двусвязные списки, стеки, очереди | Архитектура односвязных и двусвязных списков, основные операции; реализация указанных структур данных в виде классов с использованием инкапсуляции; реализация стека и очереди на основе односвязного списка | |
| 2.7 | Деревья, двоичные деревья поиска | Организация данных в виде дерева, терминология, обход и способы отображения деревьев, двоичные деревья; двоичные деревья поиска: идея и основные операции, реализация в виде классов, понятие сбалансированности дерева; реализация множества (Set<T>) и словаря (Map<T>) с помощью двоичного дерева поиска | |
| 2.8 | AVL-деревья | AVL-деревья: идея, операции вставки и удаления, балансировка (R-/L-, RL-/LR-повороты), оценка сложности операций | |
| 2.9 | 2-3-, 2-3-4- и красно-черные деревья | 2-3- и 2-3-4- деревья: идея и основные операции; красно-черные деревья как вариант представления 2-3-4-деревьев, основные операции для красно-черных деревьев и оценка сложности | |
| 2.10 | Хеширование (hash-таблицы) | Hash-таблицы как вариант структуры с потенциальным временем доступа по ключу $O(1)$, понятие hash-функции и ее качество, стратегии разрешения коллизий – открытая и закрытая адресации, реализация множества (Set<T>) и словаря (Map<T>) с помощью hash-таблиц, сравнение с деревьями поиска; комбинации hash-таблиц с деревьями поиска против «плохих» hash-функций | |
| 2.11 | Основы теории графов | Графы: основные понятия и определения; способы задания (хранения) графов (матрица смежности, списки смежности) и универсальный интерфейс представления графов для алгоритмов | |
| 2.12 | Обход графов, поиск в глубину, поиск в ширину | Реализация обхода графов (поиска) в глубину и ширину; отображение графов с помощью библиотеки Graphviz | |

| № п/п | Наименование раздела дисциплины | Содержание раздела дисциплины | Реализация раздела дисциплины с помощью онлайн-курса, ЭУМК |
|-------------------------------|------------------------------------------------------------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|------------------------------------------------------------|
| 2.13 | Оптимизационные алгоритмы на графах (кратчайшие пути, минимальное остовное дерево) | Поиск кратчайших путей в графе: алгоритм Дейкстры, алгоритм Беллмана-Форда, алгоритм Флойда-Уоршелла; поиск минимального остовного дерева: алгоритм Прима, алгоритм Краскала | |
| 2.14 | Примеры практических задач на графах | Эйлеров граф, Гамильтонов граф; примеры практических задач, которые сводятся к задачам на графах | |
| 3. Лабораторные работы | | | |
| 3.1 | Основы объектно-ориентированного программирования (инкапсуляция) | Принципы объектно ориентированного подхода; понятие класса и его экземпляров; различия между static-функциями и методами классов; классы как расширение концепции типа данных в виде объединения данных и методов их обработки, понятие состояния объекта, примеры; принципы инкапсуляции; понятие наследования и полиморфизма; класс Object и его методы | |
| 3.2 | Понятие сложности алгоритма | Понятие алгоритма, свойства алгоритма, задача оценки и сравнения эффективности алгоритмов; временная и пространственная сложность, асимптотические оценки сложности алгоритмов для различных случаев и их свойства; практические принципы оценки сложности алгоритмов в O-нотации, примеры алгоритмов с различной сложностью | |
| 3.3 | Алгоритмы сортировки: свойства, простые алгоритмы $O(n^2)$ | Свойства алгоритмов сортировки: время (временная сложность), память (пространственная сложность), устойчивость и естественность поведения, внутренние и внешние сортировки; простые сортировки и их свойства: пузырьковая сортировка, сортировка выбором, сортировка вставками; реализация указанных сортировок в обобщенном (generic) виде с параметром в виде функции сравнения элементов; вызов встроенной функции сортировки | |
| 3.4 | «Быстрые» алгоритмы сортировки | Классические «быстрые» ($O(n \cdot \log(n))$) сортировки со сравнением элементов: сортировка слиянием (merge sort), быстрая сортировка (quick sort), пирамидальная сортировка (heap sort); сортировки «без сравнений»: сортировка подсчетом (counting sort), блочная сортировка (bucket sort) | |
| 3.5 | Классификация структур данных, обзор коллекций языка Java | Классификация структур данных по принципу обращения к элементам: записи (классы), массивы, списки, словари, а также стеки и очереди; обзор коллекций Java как реализаций указанных структур данных: List<T> (ArrayList, LinkedList), Map<K, V> (TreeMap, HashMap), Stack<T>, Queue<T>, практические примеры их применения | |
| 3.6 | Односвязные и двусвязные списки, стеки, очереди | Архитектура односвязных и двусвязных списков, основные операции; реализация указанных структур данных в виде классов с использованием инкапсуляции; реализация стека и очереди на основе односвязного списка | |
| 3.7 | Деревья, двоичные деревья поиска | Организация данных в виде дерева, терминология, обход и способы отображения деревьев, двоичные деревья; двоичные деревья поиска: идея и основные операции, реализация в виде классов, понятие сбалансированности дерева; реализация множества (Set<T>) и словаря (Map<T>) с помощью двоичного дерева поиска | |

| № п/п | Наименование раздела дисциплины | Содержание раздела дисциплины | Реализация раздела дисциплины с помощью онлайн-курса, ЭУМК |
|-------|------------------------------------------------------------------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|------------------------------------------------------------|
| 3.8 | AVL-деревья | AVL-деревья: идея, операции вставки и удаления, балансировка (R-/L-, RL-/LR-повороты), оценка сложности операций | |
| 3.9 | 2-3-, 2-3-4- и красно-черные деревья | 2-3- и 2-3-4- деревья: идея и основные операции; красно-черные деревья как вариант представления 2-3-4-деревьев, основные операции для красно-черных деревьев и оценка сложности | |
| 3.10 | Хеширование (hash-таблицы) | Hash-таблицы как вариант структуры с потенциальным временем доступа по ключу $O(1)$, понятие hash-функции и ее качество, стратегии разрешения коллизий – открытая и закрытая адресации, реализация множества (Set<T>) и словаря (Map<T>) с помощью hash-таблиц, сравнение с деревьями поиска; комбинации hash-таблиц с деревьями поиска против «плохих» hash-функций | |
| 3.11 | Основы теории графов | Графы: основные понятия и определения; способы задания (хранения) графов (матрица смежности, списки смежности) и универсальный интерфейс представления графов для алгоритмов | |
| 3.12 | Обход графов, поиск в глубину, поиск в ширину | Реализация обхода графов (поиска) в глубину и ширину; отображение графов с помощью библиотеки Graphviz | |
| 3.13 | Оптимизационные алгоритмы на графах (кратчайшие пути, минимальное остовное дерево) | Поиск кратчайших путей в графе: алгоритм Дейкстры, алгоритм Беллмана-Форда, алгоритм Флойда-Уоршелла; поиск минимального остовного дерева: алгоритм Прима, алгоритм Краскала | |
| 3.14 | Примеры практических задач на графах | Эйлеров граф, Гамильтонов граф; примеры практических задач, которые сводятся к задачам на графах | |

13.2. Темы (разделы) дисциплины и виды занятий

| № п/п | Наименование темы (раздела) дисциплины | Виды занятий (часов) | | | | |
|-------|------------------------------------------------------------------|----------------------|--------------|--------------|------------------------|-------|
| | | Лекции | Практические | Лабораторные | Самостоятельная работа | Всего |
| 1 | Введение в предмет | 1 | – | – | 1 | 2 |
| 2 | Основы объектно-ориентированного программирования (инкапсуляция) | 2 | 1 | 1 | 5 | 9 |
| 3 | Понятие сложности алгоритма | 2 | 1 | 1 | 3 | 7 |
| 4 | Алгоритмы сортировки: свойства, простые алгоритмы $O(n^2)$ | 2 | 1 | 1 | 2 | 6 |
| 5 | «Быстрые» алгоритмы сортировки | 2 | 1 | 1 | 3 | 7 |
| 6 | Классификация структур данных, обзор коллекций языка Java | 2 | 1 | 1 | 3 | 7 |
| 7 | Односвязные и двусвязные списки, стеки, очереди | 2 | 1 | 1 | 3 | 7 |
| 8 | Деревья, двоичные деревья поиска | 2 | 1 | 1 | 3 | 7 |
| 9 | AVL-деревья | 2 | 1 | 1 | 3 | 7 |

| № п/п | Наименование темы (раздела) дисциплины | Виды занятий (часов) | | | | |
|-------|------------------------------------------------------------------------------------|----------------------|--------------|--------------|------------------------|-------|
| | | Лекции | Практические | Лабораторные | Самостоятельная работа | Всего |
| 10 | 2-3-, 2-3-4- и красно-черные деревья | 3 | 1 | 1 | 3 | 8 |
| 11 | Хеширование (hash-таблицы) | 2 | 1 | 1 | 2 | 6 |
| 12 | Основы теории графов | 2 | 1 | 1 | 2 | 6 |
| 13 | Обход графов, поиск в глубину, поиск в ширину | 2 | 1 | 1 | 3 | 7 |
| 14 | Оптимизационные алгоритмы на графах (кратчайшие пути, минимальное остовное дерево) | 4 | 2 | 2 | 4 | 12 |
| 15 | Примеры практических задач на графах | 4 | 2 | 2 | 2 | 10 |
| | Итого: | 34 | 16 | 16 | 42 | 108 |

14. Методические указания для обучающихся по освоению дисциплины

Рекомендуется работа с конспектами лекций, презентационным материалом, выполнение всех лабораторных и контрольных работ, заданий текущей аттестации. Учебные и методические материалы по дисциплине размещены на сетевом диске, доступным на любом компьютере в локальной сети ФКН.

15. Перечень основной и дополнительной литературы, ресурсов интернет, необходимых для освоения дисциплины

а) основная литература:

| № п/п | Источник |
|-------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 1 | Седжвик, Роберт. Фундаментальные алгоритмы на Java : Пер. с англ. / Р. Седжвик .— М. : DiaSoft, 2003. — Ч.1-4: Анализ. Структуры данных. Сортировка. Поиск .— 2003 .— 680 с. : ил. — Библиогр. в конце глав. - Предм. указ.: с. 680-687 .— Парал. тит. л. англ. — ISBN 966-7992-22-5. |
| 2 | Лафоре, Роберт. Структуры данных и алгоритмы в Java = Data structures @ algorithms in Java / Роберт Лафоре ; [пер. с англ. Е. Матвеева] .— 2-е изд. — Санкт-Петербург [и др.] : Питер, 2014 . — 701 с. : ил., табл. — (Классика computer science) .— Библиогр.: с.683-685 .— Алф. указ.: с.695-701 .— ISBN 985-5-496-00740-5. |
| 3 | Кормен, Томас. Алгоритмы : Построение и анализ : [Учебник] / Т. Кормен, Ч. Лейзерсон, Р. Ривест ; Пер. с англ. К. Белов и др.; Науч. ред. А. Шень .— М. : МЦНМО, 2002 .— 955 с. : ил. — (Классические учебники: computer science) .— ISBN 5-900916-37-5. |
| 4 | Ахо, Альфред В. Структуры данных и алгоритмы : [Учебное пособие] / Альфред В. Ахо, Джон Э. Хопкрофт, Джеффри Д. Ульман; Пер. с англ. и ред. А.А. Минько .— М. и др. : Вильямс, 2003 .— 382 с. : ил., табл. — Парал. тит. л. англ. — Библиогр.: с.369-374 .— Предм. указ.: с. 375-382 .— ISBN 5-8459-0122-7. |

б) дополнительная литература:

| № п/п | Источник |
|-------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 5 | Седжвик, Роберт. Фундаментальные алгоритмы на C++ / Р. Седжвик .— М. и др. : DiaSoft, 2002. — Ч.1-4: Анализ. Структуры данных. Сортировка. Поиск .— 2002 .— 687 с. : ил. — ISBN 5-93772-047-4. |
| 6 | Седжвик, Роберт. Фундаментальные алгоритмы на C++ / Р. Седжвик .— М. и др. : DiaSoft, 2002. — Ч.5: Алгоритмы на графах .— 2002 .— 484 с. : ил. — Парал. тит. л. англ. — ISBN 5-93772-054-7 .— ISBN 0-201-36118-3. |
| 7 | Седжвик, Роберт. Фундаментальные алгоритмы на C : Пер. с англ. / Р. Седжвик ; Принстонский ун-т .— 3-е изд. — СПб. и др. : DiaSoft, 2003. — Ч.1-4: Анализ. Структуры данных. Сортировка. Поиск .— 2003 .— 670 с. : ил. — Парал. тит. л. англ. — ISBN 5-93772-081-4. |
| 8 | Седжвик, Роберт. Фундаментальные алгоритмы на C : Пер. с англ. / Р. Седжвик ; Принстонский ун-т .— 3-е изд. — СПб., и др. : DiaSoft, 2003. — Ч.5: Алгоритмы на графах .— 2003 .— С.661-1127 : ил .— Парал. тит. л. англ. — ISBN 5-93772-082-2. |

в) информационные электронно-образовательные ресурсы (официальные ресурсы интернет):

| № п/п | Источник |
|-------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 9 | MAXimal::algo [Электронный ресурс] : — Режим доступа: http://e-maxx.ru/algo/ |
| 10 | Алгоритмы: теория и практика. Методы [Электронный ресурс] : — Режим доступа: https://stepik.org/course/217/promo |
| 11 | Алгоритмы программирования и структуры данных [Электронный ресурс] : — Режим доступа: https://openedu.ru/course/ITMOUniversity/PADS/ |

16. Перечень учебно-методического обеспечения для самостоятельной работы

| № п/п | Источник |
|-------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 1 | Шилдт, Герберт. Искусство программирования на JAVA : пер. с англ. / Герберт Шилдт, Джеймс Холмс .— СПб. [и др.] : БХВ-Петербург, 2005 .— 331 с. : ил. — Парал. тит. л. англ. — Предм. указ. : с.330-331, 4000 экз. |

17. Информационные технологии, используемые для реализации учебной дисциплины, включая программное обеспечение и информационно-справочные системы (при необходимости)

| № п/п | Наименование |
|-------|-------------------------------------------------------------------------------------------------------|
| 1 | OpenJDK - бесплатен |
| 2 | Среда разработки NetBeans или IntelliJ IDEA (академическая лицензия или версия Community) - бесплатны |

18. Материально-техническое обеспечение дисциплины:

| № п/п | Наименование |
|-------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 1 | Мультимедийная лекционная аудитория (корп. 1а, ауд. № 479 или другая подходящая): рабочее место преподавателя: ПК-Intel-i3, проектор, видеоконмутатор, микрофон, аудиосистема, специализированная мебель: доски меловые 2 шт., столы и стулья/лавки в количестве, достаточном для размещения потока студентов; выход в Интернет, доступ к фондам учебно-методической документации и электронным изданиям. |
| 2 | Компьютерный класс (корп. 1а, ауд. № 382-385 или другие подходящие): ПК-Intel-i3 16 шт., специализированная мебель: доска маркерная 1 шт., столы и стулья в количестве, достаточном для размещения академической группы (подгруппы) студентов; выход в Интернет, доступ к фондам учебно-методической документации и электронным изданиям. |

19. Оценочные средства для проведения текущей и промежуточной аттестаций

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

| № п/п | Наименование раздела дисциплины (модуля) | Компетенция(и) | Индикатор(ы) достижения компетенции | Оценочные средства |
|-------|------------------------------------------------------------------|----------------|-------------------------------------|---------------------------------------------------------------------------------|
| 1 | Введение в предмет | ОПК-6 | ОПК-6.1 | Обязательные практические задания из пункта 20.1 (контроль и оценка выполнения) |
| 2 | Основы объектно-ориентированного программирования (инкапсуляция) | ОПК-6 | ОПК-6.1, ОПК-6.2 | Обязательные практические задания из пункта 20.1 (контроль и оценка выполнения) |
| 3 | Понятие сложности алгоритма | ОПК-6 | ОПК-6.1, ОПК-6.2 | Обязательные практические задания из пункта 20.1 (контроль и оценка выполнения) |
| 4 | Алгоритмы сортировки: свойства, простые алгоритмы $O(n^2)$ | ОПК-6 | ОПК-6.1, ОПК-6.2 | Обязательные практические задания из пункта 20.1 (контроль и оценка выполнения) |

| № п/п | Наименование раздела дисциплины (модуля) | Компетенция(и) | Индикатор(ы) достижения компетенции | Оценочные средства |
|---------------------------------------------------|------------------------------------------------------------------------------------|----------------|-------------------------------------|---------------------------------------------------------------------------------|
| 5 | «Быстрые» алгоритмы сортировки | ОПК-6 | ОПК-6.1, ОПК-6.2 | Обязательные практические задания из пункта 20.1 (контроль и оценка выполнения) |
| 6 | Классификация структур данных, обзор коллекций языка Java | ОПК-6 | ОПК-6.1, ОПК-6.2 | Обязательные практические задания из пункта 20.1 (контроль и оценка выполнения) |
| 7 | Односвязные и двусвязные списки, стеки, очереди | ОПК-6 | ОПК-6.1, ОПК-6.2 | Обязательные практические задания из пункта 20.1 (контроль и оценка выполнения) |
| 8 | Деревья, двоичные деревья поиска | ОПК-6 | ОПК-6.1, ОПК-6.2 | Обязательные практические задания из пункта 20.1 (контроль и оценка выполнения) |
| 9 | AVL-деревья | ОПК-6 | ОПК-6.1, ОПК-6.2 | Обязательные практические задания из пункта 20.1 (контроль и оценка выполнения) |
| 10 | 2-3-, 2-3-4- и красно-черные деревья | ОПК-6 | ОПК-6.1, ОПК-6.2 | Обязательные практические задания из пункта 20.1 (контроль и оценка выполнения) |
| 11 | Хеширование (hash-таблицы) | ОПК-6 | ОПК-6.1, ОПК-6.2 | Обязательные практические задания из пункта 20.1 (контроль и оценка выполнения) |
| 12 | Основы теории графов | ОПК-6 | ОПК-6.1, ОПК-6.2 | Обязательные практические задания из пункта 20.1 (контроль и оценка выполнения) |
| 13 | Обход графов, поиск в глубину, поиск в ширину | ОПК-6 | ОПК-6.2 | Обязательные практические задания из пункта 20.1 (контроль и оценка выполнения) |
| 14 | Оптимизационные алгоритмы на графах (кратчайшие пути, минимальное остовное дерево) | ОПК-6 | ОПК-6.1, ОПК-6.2 | Обязательные практические задания из пункта 20.1 (контроль и оценка выполнения) |
| 15 | Примеры практических задач на графах | ОПК-6 | ОПК-6.2 | Обязательные практические задания из пункта 20.1 (контроль и оценка выполнения) |
| Промежуточная аттестация форма контроля – экзамен | | | | Перечень вопросов к экзамену из пункта 20.2 |

20. Типовые оценочные средства и методические материалы, определяющие процедуры оценивания

20.1 Текущий контроль успеваемости

Контроль успеваемости по дисциплине осуществляется с помощью контроля выполнения обязательных практических заданий. Перечень заданий:

| № п/п | Задание |
|-------|-------------------------------------------------------------------------------------------------------------------------------------------|
| 1 | Задача 1 – Инкапсуляция (≥ 30 индивидуальных вариантов, размещены на общедоступном диске в сети ФКН) |
| 2 | Задача 2 – Связные списки (≥ 30 индивидуальных вариантов, размещены на общедоступном диске в сети ФКН) |
| 3 | Задача 3 – Стеки и очереди (≥ 30 индивидуальных вариантов, размещены на общедоступном диске в сети ФКН) |
| 4 | Задача 4 – Сортировки (≥ 30 индивидуальных вариантов, размещены на общедоступном диске в сети ФКН) |
| 5 | Задача 5 – Деревья (≥ 30 индивидуальных вариантов, размещены на общедоступном диске в сети ФКН) |
| 6 | Задача 6 – Словари (на основе деревьев или hash-функций, ≥ 30 индивидуальных вариантов, размещены на общедоступном диске в сети ФКН) |

| № п/п | Задание |
|-------|--------------------------------------------------------------------------------------------------------------------------------------|
| 7 | Задача 7 – Графы 1 (>= 30 индивидуальных вариантов, размещены на общедоступном диске в сети ФКН) |
| 8 | Задача 8 – Графы 2 (интерактивное редактирование графа, >= 30 индивидуальных вариантов, размещены на общедоступном диске в сети ФКН) |

20.2 Промежуточная аттестация

Для оценивания результатов обучения на зачете используются следующие содержательные показатели (формулируется с учетом конкретных требований дисциплины):

1) знание теоретических основ учебного материала, основных определений, понятий и используемой терминологии;

2) умение проводить обоснование и представление основных теоретических и практических результатов (теорем, алгоритмов, методик) с использованием математических выкладок, блок-схем, структурных схем и стандартных описаний к ним;

3) умение связывать теорию с практикой, иллюстрировать ответ примерами, в том числе, собственными, умение выявлять и анализировать основные закономерности, полученные, в том числе, в ходе выполнения лабораторно-практических заданий;

4) умение обосновывать свои суждения и профессиональную позицию по излагаемому вопросу;

5) владение навыками программирования и экспериментирования в рамках выполняемых лабораторных заданий;

Различные комбинации перечисленных показателей определяют критерии оценивания результатов обучения (сформированности компетенций) на зачете:

- высокий (углубленный) уровень сформированности компетенций;
- повышенный (продвинутый) уровень сформированности компетенций;
- пороговый (базовый) уровень сформированности компетенций.

Для оценивания результатов обучения на зачете с оценкой используется 4-балльная шкала: «отлично», «хорошо», «удовлетворительно», «неудовлетворительно».

Соотношение показателей, критериев и шкалы оценивания результатов обучения на экзамене представлено в следующей таблице.

| Критерии оценивания компетенций | Уровень сформированности компетенций | Шкала оценок |
|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|--------------------------------------|---------------------|
| Студент владеет основными понятиями учебной дисциплины, может пояснить большинство принципов на примерах; вовремя сдал все практические задания, которые выполнены на высоком уровне, без явных ошибок. | Повышенный уровень | Отлично |
| Студент владеет основными понятиями учебной дисциплины, однако в ответах на некоторые вопросы допускает неточности; сдал все практические задания, однако к некоторым решениям студента у преподавателя есть замечания. | Базовый уровень | Хорошо |
| Студент знает основные определения из учебной дисциплины, однако пояснить многие понятия на примерах затрудняется; сдал большую часть практических заданий, однако продемонстрированные решения содержат существенные ошибки. | Пороговый уровень | Удовлетворительно |
| Студент путается в основных понятиях учебной дисциплины, не может привести примеры; не сдал большую часть практических заданий. | – | Неудовлетворительно |

Перечень вопросов к экзамену (зачету):

| № п/п | Вопрос |
|-------|--------------------------------------------------------------------------|
| 1 | Понятие сложности алгоритма |
| 2 | Практические приемы оценки сложности алгоритмов |
| 3 | Сортировка слиянием (merge sort): идея, реализация, оценка сложности |
| 4 | Быстрая сортировка (quick sort): идея, реализация, оценка сложности |
| 5 | Пирамидальная сортировка (heap sort): идея, реализация, оценка сложности |

| № п/п | Вопрос |
|-------|------------------------------------------------------------------------------------------------------------------|
| 6 | Сортировки «без сравнений»: сортировка подсчетом (counting sort), блочная сортировка (bucket sort) |
| 7 | Классификация структур данных, обзор классов-коллекций стандартной библиотеки языка Java |
| 8 | Принципы реализации структур данных в виде классов с применением инкапсуляции |
| 9 | Односвязные списки |
| 10 | Двусвязные списки |
| 11 | Реализация стека и очереди в виде связанного списка |
| 12 | Очередь с приоритетом: идея, возможные варианты эффективной реализации |
| 13 | Деревья: терминология, обход и способы отображения деревьев, двоичные деревья |
| 14 | Двоичные деревья поиска: основные операции, понятие сбалансированности дерева |
| 15 | Реализация множества (Set) и словаря (Map) на основе двоичного дерева поиска |
| 16 | AVL-деревья: идея, операции вставки и удаления, балансировка (R-/L-, RL-/LR-повороты), оценка сложности операций |
| 17 | 2-3- и 2-3-4- деревья: идея и основные операции, оценка сложности |
| 18 | Красно-черные деревья: идея и основные операции, оценка сложности |
| 19 | Hash-таблицы: понятие hash-функции, коллизии и стратегии их разрешения |
| 20 | Реализация множества (Set) и словаря (Map) на основе hash-таблицы |
| 21 | Графы: основные понятия и определения, способы задания (хранения) графов |
| 22 | Реализация обхода графов (поиска) в глубину и ширину |
| 23 | Отображение графов с помощью библиотеки Graphviz |
| 24 | Алгоритм Дейкстры |
| 25 | Алгоритм Беллмана-Форда |
| 26 | Алгоритм Флойда-Уоршелла |
| 27 | Алгоритм Прима |
| 28 | Алгоритм Краскала |
| 29 | Примеры практических задач, которые сводятся к задачам на графах |

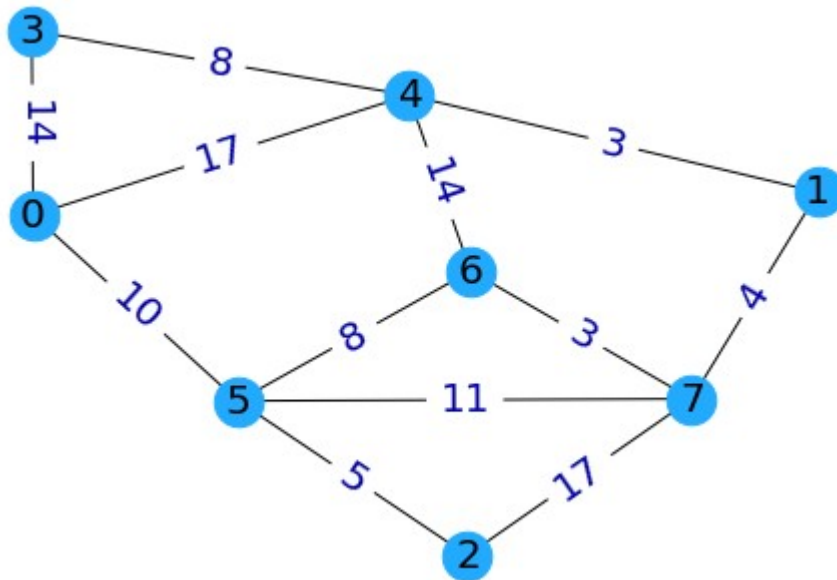
20.3. Приведённые ниже задания рекомендуется использовать при проведении диагностических работ для оценки остаточных знаний по дисциплине

Вопросы с множественным ответом (10)

- Выберите все верные для односвязного списка утверждения:
 - Каждый элемент такого списка содержит ссылку (указатель) на следующий элемент списка.
 - Односвязные списки можно использовать только для хранения целых чисел.
 - Односвязный список может быть использован для реализации стека и очереди.
 - Доступ к произвольному элементу списка по индексу осуществляется за линейное время (от размера списка).
- Выберите все верные утверждения для алгоритма быстрой сортировки (QuickSort):
 - С помощью быстрой сортировки можно сортировать только числовые данные.
 - Среднее время работы составляет $O(n \cdot \log(n))$
 - Может быть реализован рекурсивно.
 - Является устойчивым алгоритмом сортировки (равные элементы сохраняют свое взаимное расположение относительно друг друга).
- Укажите, какие алгоритмы работают за логарифмическое время – $O(\log(n))$, если речь идет о наиболее эффективных реализациях:
 - поиск элемента в отсортированном массиве;
 - поиск элемента в сбалансированном двоичном дереве поиска;
 - поиск минимального расстояния между двумя вершинами графа;
 - запись значения в массив по индексу.
- Выберите верные определения двоичного дерева поиска (ДДП):

- ДДП – это двоичное дерево, где каждый элемент – число.
 - ДДП – это двоичное дерево, в котором все узлы, за исключением листьев, содержат два дочерних узла.
 - ДДП – это двоичное дерево, для каждого узла которого справедливо утверждение: значение левого потомка меньше значения узла, а значение правого потомка больше значения узла.
 - ДДП – это двоичное дерево, для каждого узла которого справедливо утверждение: все значения в левом для узла поддереве меньше значения узла, а все значения в правом для узла поддереве больше значения узла.
5. Выберите все верные утверждения для связного графа:
- Связный граф содержит одну компоненту связности.
 - Связный граф не может быть взвешенным.
 - Степень любой вершины связного графа не может быть меньше 2-х.
 - Связный граф обязательно содержит как минимум один цикл.
6. Выберите верные утверждения для двусвязного (двунаправленного) списка:
- Двусвязный список в два раза длиннее при одном и том же количестве элементов. Перебирать элементы (осуществлять итерацию по элементам) в двусвязном списке можно как в прямом (от первого элемента к последнему), так и в обратном порядке.
 - Только двусвязный список может быть использован для реализации очереди.
 - Только двусвязный список может быть использован для реализации стека.
7. Алгоритм состоит из 2-х последовательно выполняемых частей. Вычислительная сложность первой части алгоритма – $O(n^2)$, второй – $O(n)$. Какова вычислительная сложность всего алгоритма?
- $O(n)$
 - $O(n^2)$
 - $O(n^3)$
 - Для определения вычислительной сложности всего алгоритма недостаточно данных.
8. Выберите все верные утверждения для красно-черных деревьев (КЧД):
- В узлах КЧД могут быть только целые числа.
 - КЧД являются деревьями поиска.
 - Поиск элемента в КЧД с n узлами имеет сложность $O(n)$.
 - Значение в корневом узле дерева, после его добавления, никогда не меняется.
9. Выберите все верные утверждения для алгоритма Дейкстры:
- Алгоритм находит кратчайшие пути от одной из вершин графа до всех остальных.
 - Алгоритм находит кратчайшие пути между все парами вершин графа.
 - Алгоритм может применяться как в ненаправленных, так и в направленных графах.
 - Алгоритм может работать с графами с ребрами с отрицательным весом, если кол-во вершин в графе четное.
10. Выберите все верные утверждения для хеш-таблиц:
- Хеш-таблицы – один из эффективных вариантов реализации словарей данных (ассоциативных массивов).
 - Для того, чтобы какой-ли тип данных можно было использовать в качестве ключа в хеш-таблицах достаточно, чтобы для этого типа была определена операция сравнения.
 - В качестве ключей в хеш-таблицах можно использовать строки.
 - Запись элемента в хеш-таблицу по ключу в среднем требует $O(\log(n))$ времени.

1. Для взвешенного графа, приведенного на картинке ниже, указать вес минимального остовного дерева?



Ответ: 41

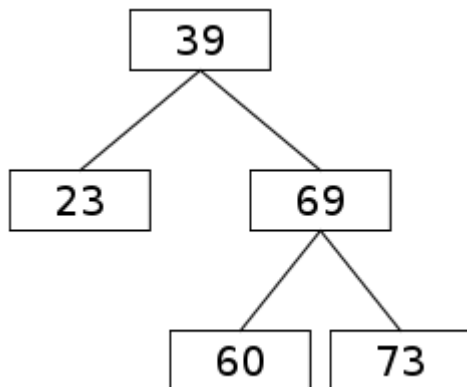
2. В наивное (примитивное, без балансировки) двоичное дерево поиска, которое вначале пустое, последовательно вставляют следующие значения:

5 9 1 3 2 4 6 7 8

Какой высоты дерево в итоге получится?

Ответ: 5

3. В AVL-дерево, показанное на рисунке ниже, добавляется значение 65. Какое значение при этом окажется в вершине дерева после балансировки?



Ответ: 60

4. Выполняется следующий фрагмент кода:

```
int[] a = { 1, 3, 4, 7, 4, 3, 2, 4, 3, 4, 2, 4, 3, 3, 1 };
Map<Integer, Integer> map = new HashMap<>();
for (int v: a) {
    int n = map.getOrDefault(v, 0);
    map.put(v, n + 1);
}
```

Какое максимальное значение (обратите внимание, значение, не ключ) будет содержаться в словаре map после выполнения этого фрагмента?

Ответ: 5

5. Неориентированный взвешенный граф, состоящий из вершин, пронумерованных от 0 до 7, задан в виде следующей матрицы смежности (номера строк и столбцов нумеруются также с 0-ля и соответствуют номерам вершин, пустые ячейки – отсутствие ребра между соответствующими вершинами):

| | | | | | | | |
|----|---|----|----|----|----|----|----|
| | | | 14 | 17 | 10 | | |
| | | | | 3 | | | 4 |
| | | | | | 5 | | 17 |
| 14 | | | | 8 | | | |
| 17 | 3 | | 8 | | | 14 | |
| 10 | | 5 | | | | 8 | 11 |
| | | | | 14 | 8 | | 3 |
| | 4 | 17 | | | 11 | 3 | |

Каково кратчайшее расстояние между вершинами 0 и 1?

Ответ: **20**

Вопросы развернутые (5)

1. Описать алгоритма Дейкстры (решаемая задача, используемые структуры данных, выполняемые действия, математическая сложность алгоритма). Проиллюстрировать работу данного алгоритма по шагам на каком-либо показательном примере (граф для примера придумать).

Критерии оценивания ответа

| Критерии оценивания | Шкала оценок |
|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|---------------------------------------|
| Обучающийся приводит абсолютно правильное описание алгоритма и действий, которые выполняются на каждом шаге. Для каждого шага верно приведены пересчитываемые значения в используемых алгоритмом структурах данных. | Отлично (90-100 баллов) |
| Обучающийся приводит правильное описание алгоритма и действий, которые выполняются на каждом шаге, но в объяснениях или в иллюстрации алгоритма есть мелкие неточности или опущены какие-либо важные детали. | Хорошо (70-80 баллов) |
| В описании алгоритма содержатся ошибки, но с целом суть алгоритма показана правильно. | Удовлетворительно (50-70 баллов) |
| Суть алгоритма описана неправильно или же в иллюстрации алгоритма по шагам на | Неудовлетворительно (менее 50 баллов) |

| | |
|----------------------------------------------------------------------------|--|
| конкретном примере обучающийся демонстрирует непонимание работы алгоритма. | |
|----------------------------------------------------------------------------|--|

2. Описать схематично код на любом языке программирования, который на основе связанных списков реализует структуру данных «Стек» (включая основные операции).

Критерии оценивания ответа

| Критерии оценивания | Шкала оценок |
|------------------------------------------------------------------------------------------------------------------------------------------------------------------------|---------------------------------------|
| Обучающийся приводит абсолютно правильную реализацию необходимых структур данных и основных операций. | Отлично (90-100 баллов) |
| Обучающийся приводит абсолютно правильную реализацию необходимых структур данных и основных операций, но в реализации есть мелкие неточности. | Хорошо (70-80 баллов) |
| В приведенном коде содержатся ошибки, но в целом идея реализации стека на основе связанного списка отражена верно. | Удовлетворительно (50-70 баллов) |
| Код не соответствует задаче или содержит грубые ошибки, демонстрирующие, что обучающийся не понимает принцип работы стека, реализованного на основе связанного списка. | Неудовлетворительно (менее 50 баллов) |

3. Описать схематично код на любом языке программирования, который на основе связанных списков реализует структуру данных «Очередь» (включая основные операции).

Критерии оценивания ответа

| Критерии оценивания | Шкала оценок |
|------------------------------------------------------------------------------------------------------------|-------------------------|
| Обучающийся приводит абсолютно правильную реализацию необходимых структур данных и основных операций. | Отлично (90-100 баллов) |
| Обучающийся приводит абсолютно правильную реализацию необходимых структур данных и основных операций, но в | Хорошо (70-80 баллов) |

| | |
|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------|------------------------------------------|
| реализации есть мелкие неточности. | |
| В приведенном коде содержатся ошибки, но в целом идея реализации очереди на основе связанного списка отражена верно. | Удовлетворительно (50-70 баллов) |
| Код не соответствует задаче или содержит грубые ошибки, демонстрирующие, что обучающийся не понимает принцип работы очереди, реализованной на основе связанного списка. | Неудовлетворительно (менее 50 баллов) |

4. Описать принцип реализации словаря на основе хеш-таблицы. Проиллюстрировать общую структуру хеш-таблицы и принцип работы.

Критерии оценивания ответа

| Критерии оценивания | Шкала оценок |
|-----------------------------------------------------------------------------------------------------------------------|------------------------------------------|
| Обучающийся приводит абсолютно правильное описание идеи и принципов работы хеш-таблиц. | Отлично (90-100 баллов) |
| Обучающийся приводит описание идеи и принципов работы хеш-таблиц, но некоторые важные аспекты реализации не показаны. | Хорошо (70-80 баллов) |
| Обучающийся приводит в целом правильное описание идеи и принципов работы хеш-таблиц, но в деталях содержатся ошибки. | Удовлетворительно (50-70 баллов) |
| Принципы работы хеш-таблиц описаны совершенно неправильно или вообще не описаны. | Неудовлетворительно (менее 50 баллов) |

5. Пояснить, в чем заключается суть асимптотической оценки сложности алгоритмов с помощью обозначения $O(f(n))$. Привести примеры.

Критерии оценивания ответа

| Критерии оценивания | Шкала оценок |
|--------------------------------|------------------|
| Обучающийся приводит абсолютно | Отлично (90-100) |

| | |
|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|---------------------------------------|
| правильное определение/объяснение для асимптотической оценки сложности и правильные примеры. | баллов) |
| Обучающийся приводит в целом правильное определение/объяснение для асимптотической оценки сложности, но с небольшими неточностями. | Хорошо (70-80 баллов) |
| По ответу обучающегося понятно, что в целом он правильно понимает суть оценки сложности алгоритмов, но дать строгое определение/объяснение не может, в примерах могут содержать ошибки. | Удовлетворительно (50-70 баллов) |
| По ответу обучающегося можно сделать вывод, что суть оценки вычислительной сложности алгоритмов он не понимает. | Неудовлетворительно (менее 50 баллов) |