МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ
РОССИЙСКОЙ ФЕДЕРАЦИИ
федеральное государственное бюджетное образовательное учреждение
высшего образования
МОСКОВСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ
ГЕОДЕЗИИ И КАРТОГРАФИИ
(МИИГАиК)

Факультет геоинформатики и информационной безопасности
Кафедра информатики и геоинформационных технологий

**ДОКУМЕНТ ПОДПИСАН
ЭЛЕКТРОННОЙ ПОДПИСЬЮ**

Сертификат: 5BFEE1D286E9C95A0C0D2C611E434BD5
Владелец: Степанченко Алексей Леонидович
Действителен: с 23.09.2026 по 17.12.2027

«УТВЕРЖДАЮ»
Проректор
А. Л. Степанченко
15 мая 2026 г.

РАБОЧАЯ ПРОГРАММА ДИСЦИПЛИНЫ (МОДУЛЯ) ПРОГРАММИРОВАНИЕ И ТЕХНОЛОГИИ РАЗРАБОТКИ ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ «Б1.О.06.01 АЛГОРИТМИЗАЦИЯ И ПРОГРАММИРОВАНИЕ»

Уровень высшего образования: бакалавриат

Направление подготовки: 09.03.03 Прикладная информатика

Направленность (профиль) подготовки: Разработка программного обеспечения для обработки и анализа пространственных данных

Квалификация (степень) выпускника: бакалавр

Форма обучения: очная

Год набора (приема на обучение): 2026

Срок получения образования: 4 года

Объем: в зачетных единицах: 3 з.е.
в академических часах: 108 ак.ч.

Москва, 2026

Разработчики:
Преподаватель кафедры Информатики и геоинформационных технологий Лебедев Е. Д.

Рабочая программа дисциплины (модуля) составлена в соответствии с требованиями ФГОС ВО по направлению подготовки 09.03.03 Прикладная информатика, утвержденного приказом Минобрнауки от 19.09.2017 № 922

Согласование и утверждение

№	Подразделение	Ответственное лицо	ФИО	Виза	Дата, протокол (при наличии)
1	факультет Геоинформатики и информационной безопасности	Председатель ученого совета факультета	Матерухин А. В.	Рассмотрено	05.11.2025, № 12-25

Содержание

1. Цель и задачи освоения дисциплины (модуля)

2. Планируемые результаты обучения по дисциплине (модулю), соотнесенные с планируемыми результатами освоения образовательной программы

3. Место дисциплины в структуре ОП

4. Объем дисциплины и виды учебной работы

5. Содержание дисциплины
   5.1. Разделы, темы дисциплины и виды занятий
   5.2. Содержание разделов, тем дисциплины и формы текущего контроля

6. Оценочные материалы текущего контроля

7. Оценочные материалы промежуточной аттестации

8. Порядок проведения промежуточной аттестации

9. Материально-техническое и учебно-методическое обеспечение дисциплины
   9.1. Перечень основной и дополнительной учебной литературы
   9.2. Профессиональные базы данных и ресурсы «Интернет», к которым обеспечивается доступ обучающихся
   9.3. Программное обеспечение и информационно-справочные системы, используемые при осуществлении образовательного процесса по дисциплине
   9.4. Оборудование и технические средства обучения

10. Методические указания по освоению дисциплины (модуля)

1. Цель и задачи освоения дисциплины (модуля)

Цель освоения дисциплины - является формирование у обучающихся знаний о принципах построения и анализа алгоритмов, а также развитие практических умений их реализации на языках программирования высокого уровня. . Освоение курса направлено на развитие способности выбирать и реализовывать алгоритмические методы решения прикладных задач информатики с учётом требований к производительности и надёжности программных решений.

Задачи изучения дисциплины:

- Освоение базовых принципов построения алгоритмов и написания кода;
- Разработка действующих программных решений на C++ и Python в соответствии с заданными алгоритмами;
- Изучение принципов модульного программирования: разделение кода на функции и их интеграция в единую структуру.;
- Освоение основных алгоритмических стратегий и принципов построения эффективных решений..

2. Планируемые результаты обучения по дисциплине (модулю), соотнесенные с планируемыми результатами освоения образовательной программы

Компетенции, индикаторы и результаты обучения

ОПК-7 Способен разрабатывать алгоритмы и программы, пригодные для практического применения

ОПК-7.1 Анализирует поставленную задачу и требования к программному продукту, определяет необходимые алгоритмические и программные решения, а также оценивает их применимость для практического использования

Знать:

ОПК-7.1/Зн5 Принципы формализации вычислительных задач, понятие входных и выходных данных, критерии корректности и эффективности алгоритма, а также основные подходы к постановке и декомпозиции задач в программировании.

Уметь:

ОПК-7.1/Ум5 Проводить анализ исходных данных и условий задачи, формулировать требования к алгоритму и программной реализации, определять ограничения и оценивать применимость выбранного решения.

Владеть:

ОПК-7.1/Нв5 Навыками анализа поставленной задачи, включающим формализацию исходных данных в виде алгоритмической модели задачи, выбор рациональных методов решения и обоснование препдпочительной алгоритмической стратегии.

ОПК-7.2 Разрабатывает эффективные алгоритмы для решения задач, а также выбирает оптимальные структуры данных для их реализации

Знать:

ОПК-7.2/Зн5 Классы алгоритмических стратегий; стандартные структуры данных (массив, список, дерево, граф, куча и др.); понятия сложности алгоритма.

Уметь:

ОПК-7.2/Ум5 Конструировать алгоритмы с приемлемой асимптотикой, выбирать структуры данных под конкретную задачу.

Владеть:

ОПК-7.2/Нв5 Навыками разработки, реализации и оптимизации алгоритмов с использованием подходящих структур данных

ОПК-7.3 Пишет программный код на одном или нескольких языках программирования, следуя принципам модульности, читаемости, документирования и соответствия стандартам кодирования

Знать:

ОПК-7.3/Зн5 Синтаксис и основные конструкции языков C++ и Python, включая типы данных, операторы, функции, циклы, условные выражения и механизмы обработки исключений, а также принципы структурного и модульного программирования, обеспечивающие читаемость, повторное использование и сопровождение программного кода.

Уметь:

ОПК-7.3/Ум5 Реализовывать алгоритмы в виде программных модулей, документировать код и соблюдать стандарты оформления

Владеть:

ОПК-7.3/Нв5 Навыками программирования и реализации алгоритмов на языках C++ и Python с соблюдением принципов читаемости и модульностия

ОПК-7.4 Проводит отладку и тестирование программного кода, выявляет и устраняет ошибки, а также оптимизирует производительность программы для обеспечения её стабильной и эффективной работы в реальных условиях

Знать:

ОПК-7.4/Зн5 Методы и стратегии тестирования (юнит-тесты, граничные случаи); приёмы отладки и оптимизации; основные метрики производительности.

Уметь:

ОПК-7.4/Ум5 Проводить тестирование, находить ошибки применять оптимизацию узких мест.

Владеть:

ОПК-7.4/Нв5 Навыками отладки и оптимизации программного кода, включающим умение использовать инструменты анализа выполнения программы, выявлять и устранять ошибки, а также применять методы оптимизации вычислений и управления памятью для повышения производительности реализованных алгоритмов.

3. Место дисциплины в структуре ОП

Дисциплина (модуль) Б1.О.06.01 «Алгоритмизация и программирование» относится к обязательной части образовательной программы и изучается в семестре(ах): 2.

В процессе изучения дисциплины студент готовится к решению типов задач профессиональной деятельности, предусмотренных ФГОС ВО и образовательной программой.

4. Объем дисциплины (модуля) и виды учебной работы

Период обучения	Общая трудоемкость (часы)	Общая трудоемкость (ЗЕТ)	Лекционные занятия (часы)	Практические занятия (часы)	Самостоятельная работа (часы)	Промежуточная аттестация (часы)

2 семестр	108	3	36	36	9	Экзамен (27)
Всего	108	3	36	36	9	27

5. Содержание дисциплины (модуля)

5.1. Разделы, темы дисциплины и виды занятий

(часы промежуточной аттестации не указываются)

Наименование раздела, темы	Всего	Лекционные занятия	Практические занятия	Самостоятельная работа	Планируемые результаты обучения, соотнесенные с результатами освоения программы
Раздел 1. Поянтие алгоритма. Методы оценки алгоритмов.	**9**	**4**	**4**	**1**	ОПК-7.1 ОПК-7.3
Тема 1.1. Введение в асимптотический анализ	5	2	2	1	
Тема 1.2. Анализ рекурсивных алгоритмов	4	2	2		
Раздел 2. Методы оптимизации алгоритмов	**46**	**20**	**20**	**6**	ОПК-7.1 ОПК-7.2 ОПК-7.3
Тема 2.1. Метод декомпозиции	9	4	4	1	
Тема 2.2. Динамическое программирование	9	4	4	1	
Тема 2.3. Жадные алгоритмы	5	2	2	1	
Тема 2.4. Поиск с возвратом	5	2	2	1	
Тема 2.5. Численные алгоритмы	5	2	2	1	
Тема 2.6. Эвристические алгоритмы	13	6	6	1	
Раздел 3. Введение в параллельные алгоритмы	**18**	**8**	**8**	**2**	ОПК-7.1 ОПК-7.2 ОПК-7.3
Тема 3.1. Векторные инструкции процессора	9	4	4	1	
Тема 3.2. Параллелизация на уровне данных	9	4	4	1	
Раздел 4. Приложение алгоритмов на примере графовых задач	**8**	**4**	**4**		ОПК-7.1 ОПК-7.2 ОПК-7.3 ОПК-7.4
Тема 4.1. Деревья. Алгоритмы индексирования данных.	4	2	2		
Тема 4.2. Задача поиска кратчайшего пути между двумя точками	4	2	2		
Итого	**81**	**36**	**36**	**9**	

5.2. Содержание разделов, тем дисциплин и формы текущего контроля

Раздел 1. Поянтие алгоритма. Методы оценки алгоритмов.

Тема 1.1. Введение в асимптотический анализ

Асимптотический анализ — это метод исследования поведения функций или систем в предельных случаях, например, при бесконечном увеличении входных данных или приближении к бесконечно удаленной точке. Он позволяет аппроксимировать сложные функции, определять временную сложность алгоритмов с помощью асимптотических нотаций (например, O, Ω, Θ), а также анализировать предельные свойства случайных величин и поведение физических систем.

Текущий контроль

Вид (форма) контроля
Тестирование

Тема 1.2. Анализ рекурсивных алгоритмов

Анализ рекурсивных алгоритмов включает оценку их временной и пространственной сложности, что является сложной задачей из-за рекурсивного характера алгоритмов и использования стека вызовов. Для анализа используются методы, такие как метод подстановки, рекуррентные соотношения (или рекуррентные уравнения), а также основная теорема (для разделяй и властвуй), которая позволяет оценивать сложность алгоритмов, разбивающих задачу на подзадачи. Визуализация рекурсивного дерева помогает понять структуру и количество вызовов, а анализ стека вызовов позволяет оценить потребление памяти.

Текущий контроль

Вид (форма) контроля
Тестирование

Раздел 2. Методы оптимизации алгоритмов

Тема 2.1. Метод декомпозиции

Метод декомпозиции в алгоритмах – это стратегия "разделяй и властвуй", где сложная задача разбивается на более мелкие, управляемые и идентичные подзадачи, которые решаются рекурсивно, а затем их решения объединяются для получения общего результата. Этот метод упрощает разработку, облегчает отладку и улучшает организацию кода, а также является основой для создания эффективных алгоритмов в информатике.
Процесс декомпозиции алгоритмов
1. Разбивка задачи на меньшие. Основная сложная задача делится на меньшие подзадачи, которые должны быть более простыми для решения.
2. Каждый из меньших экземпляров задачи решается, часто с помощью рекурсивного вызова того же алгоритма, пока задачи не станут настолько малы, чтобы их можно было решить легко.
3. Решения отдельных подзадач объединяются для получения окончательного решения исходной задачи.
Ряд эффективных алгоритмов построены на основе метода декомпозиции, среди них:
а) Сортировка слиянием (Merge Sort): Разбивает массив на подмассивы, сортирует их, а затем сливает отсортированные подмассивы.
б) Быстрая сортировка (Quick Sort): Выбирает опорный элемент и разбивает массив на две части, после чего рекурсивно сортирует эти части.

Текущий контроль

Вид (форма) контроля
Тестирование

Тема 2.2. Динамическое программирование

Динамическое программирование — это метод решения задач путём разбиения их на более простые, перекрывающиеся подзадачи, решения которых сохраняются и используются повторно для избежания лишних вычислений.

Текущий контроль

Вид (форма) контроля
Тестирование

Тема 2.3. Жадные алгоритмы

Жадные алгоритмы — это алгоритмы оптимизации, которые на каждом шаге делают локально оптимальный выбор, чтобы получить в итоге глобально оптимальное решение. Они отличаются простотой реализации и скоростью работы. Однако, жадные алгоритмы не всегда гарантируют оптимальность и работают только для задач с определенной структурой.

Текущий контроль

Вид (форма) контроля
Тестирование

Тема 2.4. Поиск с возвратом

Алгоритм backtracking (возврат к исходным данным) — это метод решения задачи перебора всех возможных вариантов с последующим выбором оптимального решения. Данный подход широко применяется в области разработки программного обеспечения, особенно в решении задач комбинаторной оптимизации, где требуется найти наилучшее решение из множества возможных комбинаций. Принцип работы алгоритма backtracking основывается на рекурсивном итеративном процессе, который проходит по всем возможным ветвям решения задачи. Основной идеей является перебор всех возможных решений путем последовательного выбора вариантов и проверки их на соответствие заданным условиям.

Текущий контроль

Вид (форма) контроля
Тестирование

Тема 2.5. Численные алгоритмы

Методы приближённого решения математических задач, для которых аналитические подходы неприменимы или неэффективны. Изучение начинается с фундаментальных вопросов: источников и классификации погрешностей (неустранимая, погрешность метода, вычислительная), понятий абсолютной и относительной погрешности, связи с числом верных знаков, представления чисел в компьютере в формате с плавающей точкой, машинной точности и округления. Вводятся ключевые концепции устойчивости и обусловленности задач, рассматриваются примеры неустойчивых вычислений, а также практические аспекты оценки накопленных погрешностей и контроля точности. Градиентный спуск изучается как базовый итерационный метод минимизации дифференцируемой функции: рассматриваются его градиентная форма с выбором шага (постоянный, дробление шага, адаптивный), условия сходимости, влияние обусловленности функции на скорость сходимости.

Текущий контроль

Вид (форма) контроля

Доклад

Тема 2.6. Эвристические алгоритмы

Эвристический алгоритм — алгоритм решения задачи, включающий практический метод, не являющийся гарантированно точным или оптимальным, но достаточный для решения поставленной задачи. Позволяет ускорить решение задачи в тех случаях, когда точное решение не может быть найдено.

Текущий контроль

Вид (форма) контроля
Тестирование

Раздел 3. Введение в параллельные алгоритмы

Тема 3.1. Векторные инструкции процессора

Векторные инструкции процессора – это команды, позволяющие выполнять одну и ту же операцию одновременно над несколькими элементами данных (вектором), что значительно ускоряет вычисления по сравнению со скалярными инструкциями, обрабатывающими по одному элементу за раз. Эта технология реализуется в виде векторных расширений, таких как AVX, в современных процессорах архитектур x86 и ARM.

Текущий контроль

Вид (форма) контроля
Тестирование

Тема 3.2. Параллелизация на уровне данных

Параллелизация на уровне данных — это выполнение одной и той же операции параллельно над различными элементами данных, например, над частями одного массива или коллекции. Этот подход разбивает большие наборы данных на меньшие части, каждая из которых обрабатывается одновременно на отдельных вычислительных узлах или ядрах процессора, что значительно ускоряет выполнение задач, таких как численные моделирования или архивация.

Текущий контроль

Вид (форма) контроля
Тестирование

Раздел 4. Приложение алгоритмов на примере графовых задач

Тема 4.1. Деревья. Алгоритмы индексирования данных.

Деревья являются в некотором смысле простейшим классом графов. Для них выполняются многие интересные утверждения, которые не всегда выполняются для графов в общем случае. Применительно к деревьям многие доказательства и рассуждения оказываются намного проще. Выдвигая какие-то гипотезы при решении задач теории графов, целесообразно сначала их проверять на деревьях. Деревья являются самым распространённым классом графов, применяемых в программировании, причём в самых разных ситуациях.

Текущий контроль

Вид (форма) контроля
Тестирование

Тема 4.2. Задача поиска кратчайшего пути между двумя точками

Задача поиска кратчайшего пути между двумя точками — это задача нахождения пути с минимальной суммой весов (например, расстояния или времени) между двумя вершинами на графе, представленном как узлы и связи. Классическим решением для графов с неотрицательными весами является алгоритм Дейкстры, а для графов, где ищут кратчайший путь между всеми парами вершин, существует алгоритм Флойда-Уоршелла.

Текущий контроль

Вид (форма) контроля
Тестирование

6. Порядок проведения промежуточной аттестации

Порядок проведения промежуточной аттестации регламентируется локальными нормативными актами Университета:

- Положением о формах, периодичности, порядке текущего контроля успеваемости и промежуточной аттестации обучающихся по основным образовательным программам высшего образования – программам бакалавриата, программам специалитета, программам магистратуры в федеральном государственном бюджетном образовательном учреждении высшего образования «Московский государственный университет геодезии и картографии», п.6 «Организация и проведение промежуточной аттестации».
- Положением о порядке осуществления образовательного процесса, проведения текущего контроля успеваемости и промежуточной аттестации обучающихся с применением электронного обучения и дистанционных образовательных технологий в федеральном государственном бюджетном образовательном учреждении высшего образования «Московский государственный университет геодезии и картографии», п.8 «Проведение промежуточной аттестации с применением ЭО и ДОТ».

Промежуточная аттестация: Экзамен, 2 семестр.

7. Оценочные материалы текущего контроля

Раздел 1. Поянтие алгоритма. Методы оценки алгоритмов.
Контролируемые ИДК: ОПК-7.1 ОПК-7.3

Тема 1.1. Введение в асимптотический анализ

Форма контроля/оценочное средство: Тестирование

Вопросы/Задания:

1. Впишите пропущенное слово: Согласно тезису Черча, каждая вычислимая функция ... рекурсивна.

Впишите пропущенное слово: Согласно тезису Черча, каждая вычислимая функция ... рекурсивна.

2. Укажите основные ресурсы, используемые для оценки работы алгоритма

1. Машинное время
2. Оплата труда программистов
3. Оперативная память
4. Долговременная память
5. Пропускная способность сети
6. Стоимость лицензии

Тема 1.2. Анализ рекурсивных алгоритмов

Форма контроля/оценочное средство: Тестирование

Вопросы/Задания:

1. Соотнесите этапы рекурсии с их описанием

Этап Описание
1 Прямой ход А Критическая часть, предотвращающая бесконечность
2 Обратный ход Б Углубление вызовов до базового случая;
3 Условие выхода В Возврат значений по цепочке

2. Выберите все утверждения, которые верно характеризуют дерево рекурсии

1. Отображает систему рекурсивных вызовов функции
2. Используется только для итеративных алгоритмов
3. Не зависит от глубины рекурсии
4. Каждый узел дерева соответствует одному вызову функции
5. листья дерева представляют базовые случаи
6. Не отражает количество операций

Раздел 2. Методы оптимизации алгоритмов

Контролируемые ИДК: ОПК-7.1 ОПК-7.2 ОПК-7.3

Тема 2.1. Метод декомпозиции

Форма контроля/оценочное средство: Тестирование

Вопросы/Задания:

1. Какие из перечисленных алгоритмов реализуют принцип метода «разделяй и властвуй»?

1. Быстрая сортировка (QuickSort)
2. Сортировка слиянием (MergeSort)
3. Линейный поиск
4. Сортировка вставками
5. Алгоритм бинарного поиска

2. Вставьте в нужном порядке слова: В алгоритме Карацубы умножение сводится к ... умножениям меньшего размера, с последующим ... результатов.

1) Трём;
2) Двум;
3) Объединением;
4) Разделением;

Тема 2.2. Динамическое программирование

Форма контроля/оценочное средство: Тестирование

Вопросы/Задания:

1. Какие из перечисленных преимуществ характерны для динамического программирования?

A. Уменьшение временной сложности алгоритма за счёт исключения повторных вычислений.
B. Компактность кода: требует меньше строк кода, чем рекурсия.
C. Возможность решать задачи, которые невозможно разбить на подзадачи.
D. Работа с одним типом данных: работа возможна только с массивами
E. Эффективное использование памяти для хранения промежуточных результатов.

2. Алгоритм вычисления значения функции F(n), где n - натуральное число, задан следующими соотношениями:

F(1) = 1
F(n) = F(n-1)*(n + 1), при n > 1

Чему равно значение функции F(5)?

Алгоритм вычисления значения функции F(n), где n - натуральное число, задан следующими соотношениями:

F(1) = 1
F(n) = F(n-1)*(n + 1), при n > 1
Чему равно значение функции F(5)?

Тема 2.3. Жадные алгоритмы

Форма контроля/оценочное средство: Тестирование

Вопросы/Задания:

1. Какие преимущества характерны для жадных алгоритмов? Выберите один или несколько верных ответов.

A. Простота реализации
B. Гарантированная оптимальность решения
C. Высокая скорость работы на больших объёмах данных
D. Обязательное использование рекурсии
E. Подходит для задач с независимыми шагами

2. Дополните предложение, ответы напишите через запятую: «Жадные алгоритмы основаны на последовательном выборе _____, которое кажется _____ на каждом шаге».

- базового случая;
- локального решения;
- рекурсивным;
- точным;
- наилучшим;

Тема 2.4. Поиск с возвратом

Форма контроля/оценочное средство: Тестирование

Вопросы/Задания:

1. Дополните предложение, записав правильный ответ через запятую: «Поиск с возвратом эффективен при наличии _____, которые позволяют _____ неподходящие ветви дерева решений».

- границ;
- отсекать;
- повторять;
- ограничений;
- результатов;
- объединять;

2. Выберите все задачи, в которых целесообразно применять метод поиска с возвратом.

A. Расстановка восьми ферзей на шахматной доске так, чтобы ни один из них не находился под боем другого
B. Поиск кратчайшего пути в графе с неотрицательными весами
C. Подбор комбинаций элементов, сумма которых равна заданному числу
D. Решение судоку
E. Сортировка массива методом пузырька

Тема 2.5. Численные алгоритмы

Форма контроля/оценочное средство: Доклад

Тема 2.6. Эвристические алгоритмы

Форма контроля/оценочное средство: Тестирование

Вопросы/Задания:

1. Какие характеристики отличают эвристические алгоритмы?

A. Всегда находят точное решение

B. Не зависят от параметров задачи
C. Основаны на приближённых решения
D. Используют интуитивные правила
E. Не гарантируют оптимальности результата

2. Соотнесите тип эвристического алгоритма и его краткую характеристику.

Тип алгоритма Характеристика
1. Генетический алгоритм
2. Поиск с имитацией отжига
3. Муравьиный алгоритм

A. Использует вероятностный выбор для выхода из локальных минимумов
C. Использует коллективное поведение агентов для нахождения пути
B. Имитация эволюционных процессов — скрещивания и мутаций

Раздел 3. Введение в параллельные алгоритмы

Контролируемые ИДК: ОПК-7.1 ОПК-7.2 ОПК-7.3

Тема 3.1. Векторные инструкции процессора

Форма контроля/оценочное средство: Тестирование

Вопросы/Задания:

1. Дополните предложение, ответы напишите через запятую: «. .. инструкции, такие как SSE и AVX, обрабатывают несколько элементов данных ... ».

- Скалярные;
- Векторные;
- Последовательно;
- Одновременно;

2. Выберите утверждения, справедливые для интрисиков.

A. Интрисики позволяют использовать SIMD-инструкции на уровне языка C/C++.
B. Интрисики автоматически оптимизируются компилятором без участия программиста.
C. Использование интрисиков может повысить производительность при параллельной обработке данных.
D. Интрисики являются макросами, не зависящими от архитектуры процессора

Тема 3.2. Параллелизация на уровне данных

Форма контроля/оценочное средство: Тестирование

Вопросы/Задания:

1. Соотнесите фреймворки с описанием:

Фреймворк Описание
1. OpenMP A. Реализует GPU-вычисления
2. PyCUDA B. Обеспечивает параллелизацию на уровне CPU с помощью директив в коде.
3. Numba C. Ускоряет вычисления с помощью JIT-компиляции.

2. Выберите утверждения, справедливые для мьютексов.

A. Обеспечивают взаимное исключение при доступе к разделяемым данным.
B. Позволяют нескольким потокам одновременно владеть одним ресурсом.
C. Поток, заблокировавший мьютекс, должен сам его разблокировать.
D. Используются исключительно для синхронизации потоков в разных процессах.
E. Мьютексы предотвращают состояния гонки (race conditions).

Раздел 4. Приложение алгоритмов на примере графовых задач

Контролируемые ИДК: ОПК-7.1 ОПК-7.2 ОПК-7.3 ОПК-7.4

Тема 4.1. Деревья. Алгоритмы индексирования данных.

Форма контроля/оценочное средство: Тестирование

Вопросы/Задания:

1. Какие утверждения справедливы для бинарного дерева поиска (BST)?

A. Левое поддерево узла содержит элементы, меньшие ключа узла.
B. Правое поддерево узла содержит элементы, большие ключа узла.
C. Элементы в поддеревьях могут располагаться в произвольном порядке.
D. Бинарное дерево поиска обеспечивает логарифмическое время поиска при сбалансированной структуре.
E. В BST невозможно удаление узлов.

2. Сколько узлов в полном бинарном дереве, состоящем из 10 уровней?

Сколько узлов в полном бинарном дереве, состоящем из 10 уровней?

Тема 4.2. Задача поиска кратчайшего пути между двумя точками

Форма контроля/оценочное средство: Тестирование

Вопросы/Задания:

1. Какие алгоритмы решают задачу поиска кратчайшего пути?

A. Алгоритм Дейкстры
B. Алгоритм Беллмана–Форда
C. Алгоритм Кнута–Морриса–Пратта
D. Алгоритм A*
E. Алгоритм Дельта-шага
F. Алгоритм Краскала

2. Какие способы представления графа используются в компьютерных структурах данных?

A. Матрица смежности
B. Список смежности
C. Матрица инцидентности
D. Таблица достижимости
E. Массив ключей

8. Оценочные материалы промежуточной аттестации

2 семестр, Экзамен

Контролируемые ИДК: ОПК-7.1 ОПК-7.2 ОПК-7.3 ОПК-7.4

Вопросы/Задания:

1. Прочитайте задание и впишите недостающий термин

Впишите пропущенное слово: согласно тезису Чёрча, каждая вычислимая функция _____ рекурсивна.

2. Прочитайте задание и запишите развернутый, обоснованный ответ

Алгоритм A имеет трудоёмкость $O(n^2)$, алгоритм B — $O(n \log n)$. Обоснуйте, почему на практике алгоритм A может оказаться быстрее алгоритма B.

3. Установите соответствие между асимптотическими нотациями и их смыслом.

Позиция Вариант
1. $O(f(n))$ А. Одновременная оценка сверху и снизу — точный порядок роста
2. $\Omega(f(n))$ Б. Оценка снизу: алгоритм работает не быстрее указанного порядка
3. $\Theta(f(n))$ В. Строго меньший порядок роста: оценка сверху, которая не достигается
4. $o(f(n))$ Г. Оценка сверху: алгоритм работает не медленнее указанного порядка

4. Прочитайте задание и запишите развернутый, обоснованный ответ

Обоснуйте, почему при оценке рекурсивного алгоритма кроме времени работы необходимо

отдельно оценивать расход памяти.

5. Установите правильную последовательность действий при анализе трудоёмкости рекурсивного алгоритма методом рекуррентных соотношений.

А. Решить соотношение и получить асимптотическую оценку

Б. Записать рекуррентное соотношение для T(n)

В. Определить, на сколько подзадач и какого размера разбивается задача

Г. Задать значение T для базового случая

Д. Оценить стоимость разделения задачи и объединения результатов

6. Установите правильную последовательность этапов работы алгоритма, построенного по стратегии «разделяй и властвуй».

А. Объединение полученных решений в решение исходной задачи

Б. Разделение задачи на несколько подзадач меньшего размера

В. Возврат результата вызывающей функции

Г. Проверка условия базового случая и его непосредственное решение

Д. Рекурсивный вызов алгоритма для каждой подзадачи

7. Прочитайте задание и впишите недостающий термин

Приём сохранения результатов уже вычисленных подзадач для их повторного использования называется _____.

8. Установите соответствие между понятиями динамического программирования и их описаниями.

Позиция Вариант

1. Мемоизация А. Одни и те же подзадачи встречаются в дереве рекурсии многократно

2. Табуляция Б. Нисходящее вычисление: результат кэшируется при первом обращении

3. Оптимальная подструктура В. Оптимальное решение задачи строится из оптимальных решений подзадач

4. Перекрывающиеся подзадачи Г. Восходящее заполнение таблицы в порядке возрастания размера подзадачи

9. Установите соответствие между задачами и критерием жадного выбора, применяемым при их решении.

Позиция Вариант

1. Выбор максимального числа непересекающихся занятий А. Два узла с наименьшей частотой

2. Непрерывная (дробная) задача о рюкзаке Б. Занятие с наименьшим временем окончания

3. Построение кода Хаффмана В. Ребро наименьшего веса, не образующее цикл

4. Построение минимального остовного дерева алгоритмом Краскала Г. Предмет с наибольшей удельной стоимостью

10. Установите правильную последовательность действий, выполняемых на одном шаге алгоритма поиска с возвратом

А. Отмена сделанного шага и возврат к предыдущему состоянию

Б. Проверка частичного решения на соответствие ограничениям задачи

В. Выбор очередного кандидата для расширения частичного решения

Г. Фиксация выбранного кандидата в частичном решении

Д. Рекурсивный переход к следующему уровню перебора

11. Прочитайте задание и впишите недостающий термин

Вектор частных производных функции по всем её переменным, указывающий направление наибыстрейшего возрастания функции, называется _____.

12. Прочитайте задание и запишите развернутый, обоснованный ответ

Обоснуйте, почему при слишком большом значении темпа обучения (learning rate) градиентный спуск может не сойтись.

13. Установите соответствие между эвристическими методами и их характеристиками.

Позиция Вариант

1. Генетический алгоритм А. Имитирует эволюцию: селекция, скрещивание, мутация

2. Метод имитации отжига Б. Вычисляет взвешенную сумму входов и применяет функцию

активации
3. Искусственный нейрон В. Использует феромонные метки и коллективное поведение агентов
4. Муравьиный алгоритм Г. Допускает ухудшающие переходы с вероятностью, убывающей вместе с температурой

14. Установите правильную последовательность этапов одного поколения генетического алгоритма.

А. Скрещивание отобранных родительских особей
Б. Формирование начальной популяции
В. Мутация полученных потомков с заданной вероятностью
Г. Вычисление приспособленности каждой особи
Д. Селекция родителей по значению приспособленности

15. Прочитайте задание и запишите развернутый, обоснованный ответ

Обоснуйте, почему метод имитации отжига допускает переход в состояние с худшим значением целевой функции.

16. Прочитайте задание и впишите недостающий термин

Принцип обработки нескольких элементов данных одной инструкцией обозначается аббревиатурой _____.

17. Установите соответствие между программными средствами параллельных вычислений и их назначением.

Позиция Вариант
1. OpenMP А. Реализует GPU-вычисления
2. PyCUDA Б. Обеспечивает параллелизацию на уровне CPU с помощью директив в коде
3. Numba В. Ускоряет вычисления с помощью JIT-компиляции

18. Прочитайте задание и запишите развернутый, обоснованный ответ

Программа распараллелена на восемь ядер, однако измеренное ускорение составило лишь три раза. Обоснуйте возможную причину.

19. Прочитайте задание и впишите недостающий термин

Ситуация, при которой результат работы программы зависит от порядка выполнения потоков, называется состоянием _____.

20. Установите соответствие между средствами синхронизации и их назначением.

Позиция Вариант
1. Мьютекс А. Счётчик, допускающий одновременное владение ресурсом не более чем N потоками
2. Семафор Б. Обеспечивает неделимость элементарной операции без применения блокировок
3. Атомарная переменная В. Позволяет потоку ожидать наступления события, временно освободив блокировку
4. Условная переменная Г. Обеспечивает монопольный доступ к критической секции одному потоку

21. Установите правильную последовательность этапов распараллеливания обработки массива на уровне данных.

А. Объединение частичных результатов (редукция)
Б. Разбиение набора данных на независимые блоки
В. Проверка отсутствия зависимостей между элементами данных
Г. Запуск обработки блоков на отдельных потоках или ядрах
Д. Ожидание завершения всех потоков (синхронизация)

22. Установите соответствие между древовидными структурами данных и их свойствами.

Позиция Вариант
1. Двоичная куча А. Сбалансированное дерево большой степени ветвления, применяемое в индексах баз данных
2. AVL-дерево Б. Хранит строки, разделяя их общие префиксы по рёбрам
3. B-дерево В. Полное двоичное дерево, в котором ключ родителя не меньше ключей потомков

4. Префиксное дерево (бор) Г. Двоичное дерево поиска с разностью высот поддеревьев не более единицы

23. Прочитайте задание и запишите развернутый, обоснованный ответ

Бинарное дерево поиска обеспечивает поиск за O(log n). Обоснуйте, почему на практике применяют самобалансирующиеся деревья (AVL, красно-чёрные).

24. Установите правильную последовательность действий при обходе дерева в ширину.

А. Извлечение узла из начала очереди и его обработка
Б. Помещение корня дерева в очередь
В. Повторение шагов, пока очередь не пуста
Г. Добавление всех потомков извлечённого узла в конец очереди
Д. Завершение обхода при опустевшей очереди

25. Прочитайте задание и впишите недостающий термин

Алгоритм поиска кратчайшего пути, использующий эвристическую оценку оставшегося расстояния до цели, называется алгоритмом _____.

26. Прочитайте задание и запишите развернутый, обоснованный ответ

Обоснуйте, почему алгоритм Дейкстры неприменим к графу, содержащему рёбра отрицательного веса.

27. Установите правильную последовательность шагов алгоритма Дейкстры.

А. Извлечение из очереди вершины с наименьшей меткой
Б. Присвоение стартовой вершине метки 0, а остальным — бесконечности
В. Повторение шагов, пока очередь не опустеет
Г. Пометка извлечённой вершины как обработанной
Д. Релаксация рёбер, исходящих из извлечённой вершины

28. Прочитайте задание и впишите недостающий термин

Способ представления графа в виде двумерного массива, элемент которого хранит признак наличия или вес ребра между парой вершин, называется матрицей

29. Установите соответствие между алгоритмами на графах и решаемой ими задачей.

Позиция Вариант
1. Алгоритм Дейкстры А. Кратчайшие пути от одной вершины при наличии рёбер отрицательного веса
2. Алгоритм Беллмана — Форда Б. Кратчайший путь до заданной цели с использованием эвристической оценки оставшегося расстояния
3. Алгоритм Флойда — Уоршелла В. Кратчайшие пути от одной вершины в графе с неотрицательными весами рёбер
4. Алгоритм A* Г. Кратчайшие пути между всеми парами вершин графа

30. Установите правильную последовательность действий при обходе графа в глубину.

А. Переход к первой непосещённой смежной вершине
Б. Пометка текущей вершины как посещённой
В. Возврат к предыдущей вершине при отсутствии непосещённых соседей
Г. Выбор стартовой вершины обхода
Д. Завершение обхода при возврате в стартовую вершину

9. Материально-техническое и учебно-методическое обеспечение дисциплины

9.1. Перечень основной и дополнительной учебной литературы

Основная литература

1. Страуструп Б. Язык программирования C++ для профессионалов: учебник / Б. Страуструп. - Язык программирования C++ для профессионалов - Москва: Интернет-Университет Информационных Технологий (ИНТУИТ), Ай Пи Ар Медиа, 2025. - 670 c. - 978-5-4497-0922-6. - Текст: электронный // IPR SMART: [сайт]. - URL: https://www.iprbookshop.ru/146412.html - Режим доступа: по подписке

2. Тарасов В. Н. Численные методы. Теория, алгоритмы, программы: учебное пособие / В. Н. Тарасов, Н. Ф. Бахарева. - Численные методы. Теория, алгоритмы, программы - Самара: Поволжский государственный университет телекоммуникаций и информатики, 2017. - 266 c. - 5-7410-0451-2. - Текст: электронный // IPR SMART: [сайт]. - URL: https://www.iprbookshop.ru/71903.html - Режим доступа: по подписке

3. Ягьяева Л. Т. Теория алгоритмов и программ: учебное пособие / Л. Т. Ягьяева, М. Ю. Валеев. - Казань: КНИТУ, 2019. - 116 с. - 978-5-7882-2737-5. - Текст: электронный // Издательство Лань: [сайт]. - URL: https://e.lanbook.com/book/196213 - Режим доступа: по подписке

4. Алексеев А. А. Основы параллельного программирования с использованием Visual Studio 2010: учебное пособие / А. А. Алексеев. - Основы параллельного программирования с использованием Visual Studio 2010 - Москва, Саратов: Интернет-Университет Информационных Технологий (ИНТУИТ), Ай Пи Ар Медиа, 2020. - 330 c. - 978-5-4497-0341-5. - Текст: электронный // IPR SMART: [сайт]. - URL: https://www.iprbookshop.ru/89456.html - Режим доступа: по подписке

5. Шелудько В. М. Основы программирования на языке высокого уровня Python: учебное пособие / В. М. Шелудько. - Основы программирования на языке высокого уровня Python - Ростов-на-Дону, Таганрог: Издательство Южного федерального университета, 2017. - 146 c. - 978-5-9275-2649-9. - Текст: электронный // IPR SMART: [сайт]. - URL: https://www.iprbookshop.ru/87461.html - Режим доступа: по подписке

Дополнительная литература

1. Павлов Л. А. Структуры и алгоритмы обработки данных: учебник для вузов / Л. А. Павлов, Н. В. Первова. - 3-е изд., стер. - Санкт-Петербург: Лань, 2021. - 256 с. - 978-5-8114-7259-8. - Текст: электронный // Издательство Лань: [сайт]. - URL: https://e.lanbook.com/book/156929 - Режим доступа: по подписке

2. Костюкова Н. И. Комбинаторные алгоритмы для программистов: учебное пособие / Н. И. Костюкова. - Комбинаторные алгоритмы для программистов - Москва: Интернет-Университет Информационных Технологий (ИНТУИТ), Ай Пи Ар Медиа, 2024. - 216 c. - 978-5-4497-2406-9. - Текст: электронный // IPR SMART: [сайт]. - URL: https://www.iprbookshop.ru/133945.html - Режим доступа: по подписке

3. Моисеенкова Т. В. Дискретная математика в примерах и задачах: учебное пособие / Т. В. Моисеенкова. - Красноярск: СФУ, 2018. - 132 с. - 978-5-7638-3967-8. - Текст: электронный // Издательство Лань: [сайт]. - URL: https://e.lanbook.com/book/157583 - Режим доступа: по подписке

4. Левин М. П. Параллельное программирование с использованием OpenMP: учебное пособие / М. П. Левин. - Параллельное программирование с использованием OpenMP - Москва: Интернет-Университет Информационных Технологий (ИНТУИТ), Ай Пи Ар Медиа, 2024. - 133 c. - 978-5-4497-3330-6. - Текст: электронный // IPR SMART: [сайт]. - URL: https://www.iprbookshop.ru/142293.html - Режим доступа: по подписке

5. Иванов Б. Н. Дискретная математика. Алгоритмы и программы. Расширенный курс: учебное пособие для вузов / Б. Н. Иванов. - 2-е изд., стер. - Санкт-Петербург: Лань, 2024. - 668 с. - 978-5-507-49205-3. - Текст: электронный // Издательство Лань: [сайт]. - URL: https://e.lanbook.com/book/382373 - Режим доступа: по подписке

6. Сузи Р. А. Язык программирования Python: учебное пособие / Р. А. Сузи. - Язык программирования Python - Москва: Интернет-Университет Информационных Технологий (ИНТУИТ), Ай Пи Ар Медиа, 2024. - 350 c. - 978-5-4497-3351-1. - Текст: электронный // IPR SMART: [сайт]. - URL: https://www.iprbookshop.ru/142310.html - Режим доступа: по подписке

7. Абрамян М. Э. Введение в стандартную библиотеку шаблонов C++. Описание, примеры использования, учебные задачи: учебник / М. Э. Абрамян. - Введение в стандартную библиотеку шаблонов C++. Описание, примеры использования, учебные задачи - Ростов-на-Дону, Таганрог: Издательство Южного федерального университета, 2017. - 178 c. - 978-5-9275-2374-0. - Текст: электронный // IPR SMART: [сайт]. - URL: https://www.iprbookshop.ru/87401.html - Режим доступа: по подписке

8. Параллельные вычисления на GPU. Архитектура и программная модель CUDA: учебное пособие / А. В. Боресков, А. А. Харламов, Н. Д. Марковский [и др.] - Параллельные вычисления на GPU. Архитектура и программная модель CUDA - Москва: Московский государственный университет имени М.В. Ломоносова, 2015. - 336 c. - 978-5-19-011058-6. - Текст: электронный // IPR SMART: [сайт]. - URL: https://www.iprbookshop.ru/54647.html - Режим доступа: по подписке

9.2. Профессиональные базы данных и ресурсы «Интернет», к которым обеспечивается доступ обучающихся

Профессиональные базы данных

Не используются.

Ресурсы «Интернет»

1. https://ratcatcher.ru/alg/ - Интерактивные конспекты по курсы "Алгоритмизация и программирование"
2. https://recursion.vercel.app/ - Инструмент построения дерева рекурсии
3. https://programforyou.ru/block-diagram-redactor - Редактор блок-схем
4. https://www.online-cpp.com/ - Онлайн-компилятор языка программирования C++
5. https://lms.miigaik.ru/ - СДО МИИГАиК
6. https://programforyou.ru/graph-redactor - Визуализация графа. Построение кратчайшего маршрута
алгоритмом Дейкстры
7. https://www.iprbookshop.ru/ - Электронно-библиотечная система IPR SMART
8. https://ratcatcher.ru/media/alg/practice/prac_2/Sim.html - Интерактивная иллюстрация метода Симпсона
9. https://ratcatcher.ru/media/alg/practice/prac_5/Массив.html - Интерактивный демонстратор работы алгоритмов сортировки
10. https://ratcatcher.ru/media/alg/practice/prac_9/A_звезда.html - Интерактивная визуализация Алгоритма A*

9.3. Программное обеспечение и информационно-справочные системы, используемые при осуществлении образовательного процесса по дисциплине

Перечень программного обеспечения
(обновление производится по мере появления новых версий программы)

1. Microsoft Visual Studio;

Перечень информационно-справочных систем
(обновление выполняется еженедельно)

Не используется.

9.4. Оборудование и технические средства обучения

Компьютерный класс

Маркерная доска - 0 шт.

Моноблок - 16 шт.
Проектор - 1 шт.

10. Методические указания по освоению дисциплины (модуля)

С целью успешного освоения данного предмета студенту необходимо выполнять домашние работы по указанным преподавателям темам и представленным на занятиях материалам. Для более успешного выполнения работы студенту предлагается использование учебно-методических материалов, соответствующих указанной теме.

Самостоятельная работа студентов описывается и регулируется методическими рекомендациями по дисциплине.

Оценка итоговых результатов обучения складывается из решения одной задачи с её защитой и ответов на два теоретических вопроса из приведённого ниже перечня.

Экзаменационное задание включает:

Задача — решается, оформляется в соответствии с требованиями, приведёнными ниже, и защищается. Защита состоит в изложении принятого решения, обосновании выбранной алгоритмической стратегии, выводе асимптотической оценки и ответах на вопросы по тексту программы.
Теоретический вопрос № 1
Теоретический вопрос № 2 .

Требования к оформлению решения
1. Постановка задачи и описание входных и выходных данных.
2. Словесное описание алгоритма; для задач блоков 2–6 — обоснование выбранной алгоритмической стратегии.
3. Асимптотическая оценка времени работы и объёма памяти с выводом.
4. Листинг программы с комментариями.
5. Результаты тестирования: приведённые в условии примеры, а также не менее двух собственных тестов, включая граничный случай.
6. Для задач блока 7 дополнительно: значения выбранных параметров метода, число итераций и таблица либо график изменения целевой функции.
7. Ответы на вопросы, приведённые в пункте «Дополнительно» соответствующей задачи.

ЗАДАЧИ
Блок 1. Анализ алгоритмов и рекурсия
Тема 1.1 «Введение в асимптотический анализ», тема 1.2 «Анализ рекурсивных алгоритмов». Задачи блока требуют не только получить ответ, но и обосновать асимптотическую оценку.
Задача 1. Счётчик работы сортировки
Тимофей утверждает, что сортировка пузырьком «почти всегда работает быстро». Чтобы проверить это, нужно измерить фактический объём работы. Реализуйте сортировку пузырьком и подсчитайте два показателя: сколько сравнений и сколько обменов элементов она выполнила. Досрочный выход при отсутствии обменов на очередном проходе использовать НЕ нужно — проходы выполняются все.
Формат ввода. В первой строке число n — размер массива ($1 \le n \le 1000$). Во второй строке n целых чисел.
Формат вывода. Два числа через пробел: количество сравнений и количество обменов.
Пример — ввод:
5
5 1 4 2 8
Пример — вывод:
10 4
Дополнительно. Объясните, почему число сравнений не зависит от исходного порядка

элементов, а число обменов равно количеству инверсий в массиве. Укажите асимптотику алгоритма.

Задача 2. Ханойские башни

Алла нашла в кабинете информатики головоломку: три стержня и n дисков разного диаметра, надетых на первый стержень в порядке убывания диаметра. За один ход разрешается переносить только верхний диск с одного стержня на другой, причём класть больший диск на меньший нельзя. Требуется перенести всю пирамиду на третий стержень.

Формат ввода. Единственное число n — количество дисков ($1 \le n \le 20$).

Формат вывода. В первой строке минимальное количество ходов. Далее, если $n \le 10$, вывести все ходы, по одному в строке, в формате «откуда куда» (номера стержней от 1 до 3).

Пример — ввод:

3

Пример — вывод:

7
1 3
1 2
3 2
1 3
2 1
2 3
1 3

Дополнительно. Запишите рекуррентное соотношение для числа ходов и решите его. Почему задача принципиально не может быть решена быстрее?

Задача 3. Цена наивной рекурсии

Гоша реализовал вычисление чисел Фибоначчи «в лоб»: $fib(n) = fib(n-1) + fib(n-2)$, базовые случаи $fib(0) = 0$ и $fib(1) = 1$. Программа работает, но при n около 40 надолго задумывается. Помогите Гоше понять причину: подсчитайте, сколько всего вызовов функции fib выполняется при вычислении fib(n), включая самый первый вызов.

Формат ввода. Единственное число n ($0 \le n \le 30$).

Формат вывода. Одно число — общее количество вызовов функции.

Пример 1 — ввод:

5

Пример 1 — вывод:

15

Пример 2 — ввод:

0

Пример 2 — вывод:

1

Дополнительно. Постройте дерево рекурсии для n = 5. Выразите число вызовов через числа Фибоначчи и укажите асимптотику. Предложите способ снизить её до O(n), не отказываясь от рекурсии.

Блок 2. Разделяй и властвуй

Тема 2.1 «Метод декомпозиции». Во всех задачах блока требуется именно рекурсивное разбиение задачи на подзадачи; решения «одним проходом» не засчитываются.

Задача 4. Мажоритарный элемент

На выборах старосты группы каждый студент назвал одну фамилию. Староста считается избранным, если за него подано строго больше половины голосов. Определите, избран ли кто-нибудь, применив метод «разделяй и властвуй»: массив делится пополам, мажоритарный элемент ищется в каждой половине, после чего кандидаты проверяются подсчётом на всём массиве.

Формат ввода. В первой строке число n ($1 \le n \le 100\,000$). Во второй строке n целых чисел — номера кандидатов.

Формат вывода. Номер избранного кандидата или слово «нет», если такого нет.

Пример 1 — ввод:

7

2 2 1 2 3 2 2
Пример 1 — вывод:
2
Пример 2 — ввод:
4
1 2 3 4
Пример 2 — вывод:
нет
Дополнительно. Запишите рекуррентное соотношение $T(n) = 2 \cdot T(n/2) + O(n)$ и получите оценку по основной теореме.
Задача 5. k-я порядковая статистика
Рита ведёт таблицу результатов соревнований и хочет быстро узнавать, какое место занимает k-й по величине результат, не сортируя таблицу целиком. Найдите k-й по возрастанию элемент массива методом быстрого выбора (quickselect): массив разбивается по опорному элементу, после чего рекурсивный спуск выполняется только в ту часть, где находится искомый индекс.
Формат ввода. В первой строке числа n и k ($1 \le k \le n \le 100\,000$). Во второй строке n целых чисел.
Формат вывода. Одно число — k-й по возрастанию элемент массива.
Пример — ввод:
6 3
7 10 4 3 20 15
Пример — вывод:
7
Дополнительно. Чем оценка среднего случая отличается от оценки худшего случая? Как выбор опорного элемента влияет на асимптотику?
Задача 6. Слияние k отсортированных списков
В деканат поступило k отсортированных по возрастанию списков оценок из разных групп. Нужно объединить их в один отсортированный список. Примените декомпозицию: списки разбиваются на две половины, каждая половина сливается рекурсивно, затем два результата объединяются процедурой слияния двух отсортированных последовательностей.
Формат ввода. В первой строке число k ($1 \le k \le 100$). Далее k строк, в каждой сначала длина списка, затем его элементы в порядке неубывания. Суммарное количество элементов не превосходит 100 000.
Формат вывода. Все элементы в порядке неубывания через пробел.
Пример — ввод:
3
3 1 4 7
2 2 5
4 0 3 6 8
Пример — вывод:
0 1 2 3 4 5 6 7 8
Дополнительно. Сравните асимптотику этого подхода с последовательным слиянием списков по одному («накопительным» способом) и объясните разницу.
Задача 7. Числа Фибоначчи через степень матрицы
Гоше нужно вычислять числа Фибоначчи для очень больших n, и линейный алгоритм его уже не устраивает. Известно, что возведение матрицы [[1, 1], [1, 0]] в степень n даёт матрицу, в которой левый нижний элемент равен fib(n). Реализуйте умножение матриц 2×2 и их возведение в степень методом «разделяй и властвуй»: чётная степень сводится к квадрату половинной степени.
Формат ввода. Единственное число n ($0 \le n \le 1\,000\,000$). Ответ выводить по модулю 1 000 000 007.
Формат вывода. Одно число — fib(n) по модулю 1 000 000 007.
Пример 1 — ввод:
10

Пример 1 — вывод:
55
Пример 2 — ввод:
1
Пример 2 — вывод:
1
Дополнительно. Укажите асимптотику алгоритма и число умножений матриц. Чем эта задача отличается от быстрого возведения числа в степень?
Блок 3. Динамическое программирование
Тема 2.2 «Динамическое программирование». В отчёте по каждой задаче нужно явно указать: что является состоянием, каково рекуррентное соотношение перехода, каковы базовые значения и в каком порядке заполняется таблица.
Задача 8. Наибольшая возрастающая подпоследовательность
Алла записывает высоты островов архипелага в том порядке, в каком их посещает. Она хочет выбрать как можно более длинную последовательность посещённых островов (не обязательно подряд идущих), в которой высота строго возрастает. Найдите длину такой последовательности.
Формат ввода. В первой строке число n ($1 \le n \le 5000$). Во второй строке n целых чисел — высоты островов.
Формат вывода. Одно число — длина наибольшей строго возрастающей подпоследовательности.
Пример — ввод:
8
10 9 2 5 3 7 101 18
Пример — вывод:
4
Дополнительно. Приведите решение за $O(n^2)$. Опишите (без реализации) идею ускорения до $O(n \log n)$.
Задача 9. Расстояние редактирования
Система проверки домашних работ должна находить опечатки: для этого нужно уметь измерять, насколько одно слово отличается от другого. Расстоянием редактирования (расстоянием Левенштейна) называется минимальное число операций вставки, удаления и замены одного символа, переводящих первую строку во вторую. Вычислите это расстояние.
Формат ввода. Две строки латинскими буквами, длиной не более 2000 символов каждая.
Формат вывода. Одно число — расстояние редактирования.
Пример 1 — ввод:
horse
ros
Пример 1 — вывод:
3
Пример 2 — ввод:
abc
abc
Пример 2 — вывод:
0
Дополнительно. Опишите таблицу состояний и три варианта перехода. Как сократить память до $O(\min(n, m))$?
Задача 10. Наибольшая общая подпоследовательность
Тимофей сравнивает две версии текста лабораторной работы и хочет понять, какая часть текста осталась неизменной. Найдите длину наибольшей общей подпоследовательности двух строк — последовательности символов, входящей в обе строки в том же относительном порядке, но не обязательно подряд.
Формат ввода. Две строки латинскими буквами, длиной не более 2000 символов каждая.
Формат вывода. Одно число — длина наибольшей общей подпоследовательности.
Пример — ввод:

ABCBDAB
BDCABA
Пример — вывод:
4
Дополнительно. Чем эта задача отличается от задачи о наибольшей общей подстроке? Как по заполненной таблице восстановить саму подпоследовательность?
Задача 11. Число способов размена
Автомат по продаже кофе принимает монеты нескольких номиналов, причём монет каждого номинала неограниченно много. Требуется узнать, сколькими различными способами можно набрать заданную сумму. Способы, отличающиеся только порядком монет, считаются одинаковыми.
Формат ввода. В первой строке числа S и m — сумма ($1 \le S \le 10\,000$) и количество номиналов ($1 \le m \le 50$). Во второй строке m различных номиналов.
Формат вывода. Одно число — количество способов набрать сумму S.
Пример — ввод:
5 3
1 2 5
Пример — вывод:
4
Дополнительно. Объясните, почему при внешнем цикле по номиналам порядок монет не учитывается, а при внешнем цикле по сумме — учитывается. Это ключевое различие двух похожих задач.
Задача 12. Самая длинная палиндромная подстрока
В олимпиадной задаче по лингвистике требуется найти в слове самый длинный фрагмент, читающийся одинаково слева направо и справа налево. Фрагмент должен состоять из идущих подряд символов. Если таких фрагментов несколько, выведите любой из них.
Формат ввода. Одна строка латинскими буквами длиной не более 1000 символов.
Формат вывода. Самая длинная палиндромная подстрока.
Пример 1 — ввод:
babad
Пример 1 — вывод:
bab
Пример 2 — ввод:
cbbd
Пример 2 — вывод:
bb
Дополнительно. Приведите решение динамическим программированием за $O(n^2)$ с состоянием «является ли подстрока с i по j палиндромом». Опишите альтернативный подход «расширение от центра».
Блок 4. Жадные алгоритмы
Тема 2.3 «Жадные алгоритмы». Для каждой задачи требуется сформулировать критерий жадного выбора и обосновать, почему локально оптимальный выбор приводит к глобальному оптимуму.
Задача 13. Код Хаффмана
Для сжатия сообщения строится префиксный код: часто встречающимся символам назначаются короткие двоичные коды, редким — длинные. По заданным частотам символов постройте код Хаффмана и вычислите длину закодированного сообщения в битах, то есть сумму произведений частоты каждого символа на длину его кода.
Формат ввода. В первой строке число m — количество различных символов ($2 \le m \le 1000$). Во второй строке m целых положительных чисел — частоты символов.
Формат вывода. Одно число — длина закодированного сообщения в битах.
Пример — ввод:
6
5 9 12 13 16 45
Пример — вывод:

224
Дополнительно. Почему на каждом шаге объединяются именно два наименьших по частоте узла? Какая структура данных обеспечивает выбор минимума за O(log m)?
Задача 14. Платформы вокзала
На вокзале известно расписание прибытия и отправления поездов в течение суток. Пока поезд стоит на вокзале, он занимает платформу целиком. Определите минимальное число платформ, которого достаточно, чтобы обслужить все поезда без задержек. Поезд, отправляющийся в тот же момент, в который прибывает другой, освободить платформу не успевает.
Формат ввода. В первой строке число n ($1 \le n \le 100\,000$). В следующих n строках по два числа — время прибытия и время отправления в формате ЧЧММ.
Формат вывода. Одно число — минимальное необходимое количество платформ.
Пример — ввод:
6
900 910
940 1200
950 1120
1100 1130
1500 1900
1800 2000
Пример — вывод:
3
Дополнительно. Опишите приём с раздельной сортировкой моментов прибытия и отправления. Чем задача отличается от задачи о выборе максимального числа непересекающихся занятий?
Задача 15. Заправки на трассе
Автомобиль выезжает из точки 0 и должен добраться до точки D. На полном баке он проезжает ровно L километров. Вдоль трассы расположены заправки на известных расстояниях от старта. Определите минимальное число заправок, необходимое для того, чтобы доехать до пункта назначения, либо установите, что доехать невозможно. В начале пути бак полон.
Формат ввода. В первой строке числа D и L. Во второй строке число k — количество заправок. В третьей строке k возрастающих расстояний от старта до заправок.
Формат вывода. Минимальное число заправок или −1, если доехать невозможно.
Пример 1 — ввод:
950 400
4
200 375 550 750
Пример 1 — вывод:
2
Пример 2 — ввод:
500 200
1
100
Пример 2 — вывод:
-1
Дополнительно. Обоснуйте жадный критерий «ехать до самой дальней достижимой заправки». Почему выбор ближайшей заправки не оптимален?
Задача 16. Склеивание верёвок
В мастерской лежит несколько кусков верёвки. За одну операцию можно связать два куска, при этом стоимость операции равна суммарной длине связываемых кусков, а на их месте появляется один кусок этой длины. Требуется связать все куски в один так, чтобы суммарная стоимость всех операций была минимальной.
Формат ввода. В первой строке число n ($1 \le n \le 100\,000$). Во второй строке n длин кусков.
Формат вывода. Одно число — минимальная суммарная стоимость.
Пример 1 — ввод:

4
4 3 2 6
Пример 1 — вывод:
29
Пример 2 — ввод:
1
7
Пример 2 — вывод:
0
Дополнительно. Объясните, почему задача сводится к построению дерева Хаффмана. Что произойдёт, если на каждом шаге связывать два самых длинных куска?
Блок 5. Поиск с возвратом
Тема 2.4 «Поиск с возвратом». Во всех задачах блока требуется явно указать, на каком условии происходит отсечение ветви перебора.
Задача 17. Ход конём
Шахматный конь стоит в левом верхнем углу доски размером n × n. Требуется обойти всю доску, побывав в каждой клетке ровно один раз. Найдите любой такой маршрут методом поиска с возвратом либо установите, что маршрута не существует.
Формат ввода. Единственное число n ($1 \le n \le 8$).
Формат вывода. Матрица n × n, в клетках которой записаны номера ходов от 1 до n^2, либо слово «нет», если обход невозможен.
Пример — ввод:
3
Пример — вывод:
нет
Дополнительно. Реализуйте эвристику Варнсдорфа (первым выбирается ход в клетку с наименьшим числом продолжений) и сравните время работы с обычным перебором при n = 8. Это прямая иллюстрация того, как эвристика ускоряет точный метод.
Задача 18. Правильные скобочные последовательности
Рита готовит задания для кружка и хочет получить список всех правильных скобочных последовательностей из n пар круглых скобок. Последовательность правильна, если каждой открывающей скобке соответствует закрывающая и они корректно вложены. Выведите все такие последовательности в лексикографическом порядке.
Формат ввода. Единственное число n ($1 \le n \le 10$).
Формат вывода. В первой строке количество последовательностей. Далее сами последовательности, по одной в строке, в лексикографическом порядке.
Пример — ввод:
3
Пример — вывод:
5
((()))
(()())
(())()
()(())
()()()
Дополнительно. Какие два условия позволяют отсечь заведомо неверную ветвь, не достраивая её до конца? Количеству последовательностей соответствуют числа Каталана — проверьте это для n от 1 до 5.
Задача 19. Путь в лабиринте
Лабиринт задан прямоугольной картой, где символ «.» обозначает свободную клетку, а «#» — стену. Робот стартует в левой верхней клетке и должен попасть в правую нижнюю, перемещаясь по свободным клеткам в четырёх направлениях и не посещая одну клетку дважды. Подсчитайте, сколькими различными путями это можно сделать.
Формат ввода. В первой строке числа h и w — высота и ширина карты ($1 \le h, w \le 8$). Далее h строк по w символов.

Формат вывода. Одно число — количество различных путей.
Пример 1 — ввод:
2 2
..
..
Пример 1 — вывод:
2
Пример 2 — ввод:
2 2
.#
..
Пример 2 — вывод:
1
Дополнительно. Почему для этой задачи неприменимо динамическое программирование, работающее для случая движения только вправо и вниз? Какую роль играет пометка клетки как посещённой и её снятие при возврате?
Блок 6. Деревья и графовые алгоритмы
Раздел 4 «Приложение алгоритмов на примере графовых задач», темы 4.1–4.2. В отчёте указывается выбранный способ представления графа и обоснование этого выбора.
Задача 20. Топологическая сортировка
Учебный план задан списком дисциплин и отношений «дисциплина A должна быть изучена раньше дисциплины B». Постройте порядок изучения дисциплин, согласованный со всеми ограничениями, либо установите, что план противоречив (содержит цикл). Если корректных порядков несколько, выведите любой.
Формат ввода. В первой строке числа n и m — количество дисциплин ($1 \le n \le 100\ 000$) и количество ограничений. В следующих m строках пары номеров a и b, означающие, что a изучается раньше b.
Формат вывода. Последовательность номеров дисциплин или слово «цикл».
Пример — ввод:
4 3
1 2
1 3
3 4
Пример — вывод:
1 3 4 2
Дополнительно. Приведите два способа: обход в глубину с выводом в порядке завершения вершин и алгоритм Кана со степенями захода. Как каждый из них обнаруживает цикл?
Задача 21. Компоненты связности
Компьютерная сеть задана списком каналов связи между узлами. Определите, на сколько изолированных фрагментов распадается сеть, и выведите размер наибольшего из них.
Формат ввода. В первой строке числа n и m — количество узлов и каналов. В следующих m строках пары номеров узлов, соединённых каналом. Каналы двусторонние.
Формат вывода. Два числа: количество компонент связности и размер наибольшей компоненты.
Пример — ввод:
6 3
1 2
2 3
5 6
Пример — вывод:
3 3
Дополнительно. Сравните реализацию через обход в глубину и через систему непересекающихся множеств (DSU). Укажите асимптотику каждой.
Задача 22. Кратчайший маршрут с восстановлением пути
Карта города задана как взвешенный граф: перекрёстки — вершины, улицы — рёбра с

неотрицательным временем проезда. Найдите минимальное время в пути от перекрёстка s до перекрёстка t и сам маршрут.
Формат ввода. В первой строке числа n и m. Во второй строке номера s и t. В следующих m строках тройки a, b, w — концы улицы и время проезда (двустороннее движение).
Формат вывода. В первой строке минимальное время или −1, если маршрут не существует. Во второй строке последовательность вершин маршрута.
Пример — ввод:
5 6
1 5
1 2 10
1 3 3
3 2 4
2 4 2
3 4 8
4 5 5
Пример — вывод:
14
1 3 2 4 5
Дополнительно. Почему для восстановления маршрута достаточно хранить массив предков? Что произойдёт с алгоритмом Дейкстры при появлении ребра отрицательного веса и какой алгоритм следует применить в этом случае?
Задача 23. Минимальное остовное дерево
Требуется соединить все населённые пункты линиями связи так, чтобы из любого пункта можно было связаться с любым другим, а суммарная стоимость прокладки линий была минимальной. Известны стоимости прокладки линии между каждой парой пунктов, для которых это технически возможно.
Формат ввода. В первой строке числа n и m. В следующих m строках тройки a, b, w — пара пунктов и стоимость линии между ними.
Формат вывода. Суммарная стоимость минимального остовного дерева или слово «нет», если соединить все пункты невозможно.
Пример — ввод:
4 5
1 2 1
2 3 2
3 4 3
1 4 10
1 3 4
Пример — вывод:
6
Дополнительно. Реализуйте алгоритм Краскала с системой непересекающихся множеств. Обоснуйте корректность жадного выбора минимального ребра (свойство разреза).
Блок 7. Градиентный спуск и эвристические алгоритмы
Темы 2.5 «Градиентный спуск» и 2.6 «Эвристические алгоритмы». Задачи этого блока не имеют единственно верного ответа: результат зависит от начального приближения и параметров метода. Поэтому в отчёте обязательно приводятся выбранные параметры, число итераций и график (или таблица) изменения целевой функции. Ответ считается верным, если он отличается от эталонного не более чем на указанную в задаче точность.
Задача 24. Градиентный спуск для функции одной переменной
Реализуйте метод градиентного спуска для поиска минимума функции $f(x) = x^2 - 4x + 7$. Производная вычисляется аналитически. На каждой итерации текущая точка смещается по формуле $x \leftarrow x - \eta \cdot f'(x)$. Итерации прекращаются, когда $|f'(x)|$ становится меньше заданной точности ε либо достигнут предел в 100 000 итераций.
Формат ввода. В одной строке три вещественных числа: начальное приближение x_0, темп обучения η и точность ε.
Формат вывода. Две строки: найденная точка минимума и значение функции в ней, каждое с

четырьмя знаками после запятой. В третьей строке — число выполненных итераций.
Пример — ввод:
10 0.1 0.0001
Пример — вывод:
2.0000
3.0000
<число итераций>
Дополнительно. Проведите эксперимент: выполните программу при $\eta = 0{,}01$, $\eta = 0{,}1$, $\eta = 0{,}9$ и $\eta = 1{,}1$. Объясните, почему при $\eta = 1{,}1$ процесс расходится, и определите экспериментально пороговое значение шага.
Задача 25. Численная оценка градиента
Аналитическая производная целевой функции известна далеко не всегда. Реализуйте вычисление градиента функции двух переменных методом центральных конечных разностей: $\partial f/\partial x \approx (f(x + h, y) - f(x - h, y)) / (2h)$, аналогично для второй переменной. Примените его к функции $f(x, y) = x^2 + 3y^2$ и выполните несколько шагов градиентного спуска.
Формат ввода. В первой строке координаты точки x и y. Во второй строке шаг дифференцирования h.
Формат вывода. Две компоненты градиента в заданной точке с четырьмя знаками после запятой.
Пример — ввод:
1 2
0.0001
Пример — вывод:
2.0000 12.0000
Дополнительно. Сравните результат с аналитическим градиентом (2x, 6y). Исследуйте, как погрешность зависит от h: при слишком большом h растёт погрешность метода, при слишком малом — погрешность округления IEEE 754. Найдите оптимальное значение h экспериментально.
Задача 26. Линейная регрессия градиентным спуском
Дан набор экспериментальных точек. Требуется подобрать коэффициенты прямой $y = k \cdot x + b$, минимизирующие среднеквадратичную ошибку. Реализуйте градиентный спуск по двум параметрам: частные производные ошибки по k и b вычисляются аналитически, оба параметра обновляются одновременно на каждой итерации.
Формат ввода. В первой строке число точек n. В следующих n строках пары координат x и y. В последней строке темп обучения η и число итераций.
Формат вывода. Два числа — найденные коэффициенты k и b с точностью до трёх знаков после запятой.
Пример — ввод:
4
1 2
2 4
3 6
4 8
0.01 10000
Пример — вывод:
2.000 0.000
Дополнительно. Сравните полученный результат с точным решением по формулам метода наименьших квадратов. Исследуйте влияние нормировки входных данных на скорость сходимости.
Задача 27. Минимизация методом имитации отжига
Функция $f(x) = x^4 - 3x^3 + 2$ имеет на отрезке $[-2; 4]$ несколько экстремумов, и обычный градиентный спуск из неудачного начального приближения застревает в локальном минимуме. Реализуйте метод имитации отжига: на каждой итерации из окрестности текущей точки выбирается случайная соседняя; если она лучше — принимается безусловно, если хуже — принимается с вероятностью $\exp(-\Delta E / T)$; температура снижается по закону $T \leftarrow \alpha \cdot T$.

Формат ввода. В одной строке: начальная температура T_0, коэффициент охлаждения α ($0 < \alpha < 1$), количество итераций и радиус окрестности.
Формат вывода. Найденная точка минимума и значение функции в ней с точностью до трёх знаков после запятой.
Пример — ввод:
100 0.99 10000 0.5
Пример — вывод:
2.250 -6.543
Дополнительно. Ответ считается верным при отклонении значения функции не более чем на 0,01. Запустите программу 20 раз и постройте распределение результатов. Исследуйте, что происходит при $\alpha = 0{,}5$ (быстрое охлаждение) и при $\alpha = 0{,}999$ (медленное охлаждение).
Задача 28. Генетический алгоритм: максимизация функции
Требуется найти максимум функции $f(x) = x \cdot \sin(10\pi \cdot x) + 2$ на отрезке $[-1; 2]$ генетическим алгоритмом. Функция имеет большое число локальных максимумов, поэтому переборные и градиентные методы неэффективны. Закодируйте x двоичной строкой фиксированной длины, реализуйте турнирную селекцию, одноточечный кроссинговер иточечную мутацию, сохраняя лучшую особь поколения (элитизм).
Формат ввода. В одной строке: размер популяции, число поколений, вероятность кроссинговера и вероятность мутации.
Формат вывода. Найденное значение x и значение функции f(x) с точностью до трёх знаков после запятой.
Пример — ввод:
100 200 0.8 0.05
Пример — вывод:
1.851 3.850
Дополнительно. Ответ считается верным при значении функции не менее 3,80. Постройте график изменения лучшей и средней приспособленности по поколениям. Исследуйте эффект преждевременной сходимости: выполните запуск при вероятности мутации 0 и объясните результат.
Задача 29. Обучение персептрона логической функции
Реализуйте искусственный нейрон с двумя входами, двумя весами и смещением, использующий пороговую функцию активации. Обучите его логической функции по правилу персептрона: при ошибке веса корректируются как $w_i \leftarrow w_i + \eta \cdot (d - y) \cdot x_i$, смещение — как $b \leftarrow b + \eta \cdot (d - y)$, где d — требуемый выход, y — фактический. Обучение продолжается до эпохи без единой ошибки.
Формат ввода. В первой строке название функции: AND или OR. Во второй строке темп обучения η.
Формат вывода. В первой строке число эпох, потребовавшихся для обучения. Во второй строке найденные веса w_1, w_2 и смещение b. Далее четыре строки — выход нейрона на всех наборах входов 00, 01, 10, 11.
Пример — ввод:
AND
0.1
Пример — вывод:
<число эпох>
<w1> <w2> <b>
0
0
0
1
Дополнительно. Изобразите на плоскости четыре входных набора и найденную разделяющую прямую $w_1x_1 + w_2x_2 + b = 0$. Объясните геометрический смысл смещения b.
Задача 30. Линейная разделимость и задача XOR
Повторите обучение персептрона из предыдущей задачи для функции «исключающее ИЛИ» (XOR), ограничив обучение 10 000 эпохами. Зафиксируйте, что обучение не сходится: на

каждой эпохе остаётся хотя бы одна ошибка. Затем постройте сеть из двух нейронов скрытого слоя и одного выходного нейрона, задав их веса вручную так, чтобы сеть правильно реализовала XOR.
Формат ввода. В первой строке темп обучения η. Во второй строке предельное число эпох.
Формат вывода. В первой строке слово «не обучается» и число выполненных эпох. Далее четыре строки — выходы двухслойной сети на наборах 00, 01, 10, 11.
Пример — ввод:
0.1
10000
Пример — вывод:
не обучается 10000
0
1
1
0
Дополнительно. Объясните причину: наборы XOR линейно неразделимы, а одиночный нейрон строит только одну разделяющую гиперплоскость. Дополнительно выведите минимальное число ошибок за эпоху, достигнутое в ходе обучения, и объясните, почему этот показатель меняется от запуска к запуску и почему он не может стать нулевым. Укажите, какой алгоритм применяется для автоматического обучения
многослойной сети и с каким методом оптимизации он связан.

Перечень теоретических вопросов

рекурсии.
Что понимается под абстрактной моделью вычислений? Охарактеризуйте вычислительные ресурсы алгоритма и опишите асимптотический анализ с использованием нотации «О-большое».
Как вычисляется значение рекурсивной функции? Опишите анализ рекуррентных соотношений и сформулируйте основную теорему (мастер-теорему).
Опишите представление целых и вещественных чисел в памяти компьютера. Какие проблемы вычислений порождает стандарт IEEE 754? Приведите примеры применения побитовых операторов.
Раскройте основы параллелизации на уровне процессора. Что такое векторные операции и интринсики?
Дайте определения понятиям «процесс» и «поток». Опишите параллелизацию на уровне данных. В чём состоит проблема разделения ресурсов? Проиллюстрируйте её задачей об обедающих философах и перечислите средства синхронизации.
Охарактеризуйте класс алгоритмов численных приближений. Раскройте идею численного интегрирования. Какой ресурс параллелизации в нём заложен?
Раскройте метод «разделяй и властвуй». Проиллюстрируйте его бинарным возведением в степень и алгоритмом Карацубы. Какой ресурс параллелизации даёт декомпозиция задачи?
Как оценивается эффективность параллельных вычислений? Постройте модель вычислений в виде графа «операции — операнды» и перечислите показатели эффективности параллельного алгоритма.
Сформулируйте закон Амдала и закон Густавсона — Барсиса. В чём состоят практические следствия закона Амдала и каковы границы его применимости?
Опишите особенности реализации параллельных алгоритмов в Python. Что такое Global Interpreter Lock и как он влияет на многопоточность? Перечислите готовые решения для многопоточных приложений.
Проведите сравнительный асимптотический анализ сортировки пузырьком, поразрядной сортировки и быстрой сортировки. Какой ресурс параллелизации заложен в быстрой сортировке?
Охарактеризуйте метод полного перебора и укажите его недостатки. Какими способами перебор оптимизируют? Опишите алгоритм поиска с возвратом.

Дайте определение графа и приведите классификацию видов графов. Опишите способы представления графов в памяти. Приведите примеры графовых задач и поясните, почему граф служит инструментом научного анализа.
Дайте определение дерева как структуры данных. Опишите способы обхода деревьев. Охарактеризуйте двоичную кучу и поиск в двоичном дереве поиска. Раскройте идею индексирования данных.
Опишите способы представления графов в памяти, включая специализированные форматы хранения. Сравните поиск в глубину и поиск в ширину.
Опишите специализированные форматы хранения графов. Каким образом распараллеливается поиск в ширину?
Раскройте метод динамического программирования на примере задачи о кратчайшем пути. Опишите алгоритм Дейкстры. Какие приёмы его оптимизации существуют?
Опишите алгоритм Дейкстры. Какой ресурс параллелизации в нём заложен и что ограничивает его использование?
Опишите алгоритм дельта-шагания для задачи о кратчайшем пути. Чем он выгоднее алгоритма Дейкстры при параллельном исполнении?
Охарактеризуйте класс жадных алгоритмов. Раскройте задачу о рюкзаке. Приведите пример жадной оптимизации на примере алгоритма A*.
Охарактеризуйте класс эвристических алгоритмов. Опишите алгоритм имитации отжига и приведите примеры его применения.
Опишите генетический алгоритм: представление решения, операторы селекции, скрещивания и мутации, критерии останова. Приведите примеры его применения.
Раскройте формализацию понятия алгоритма средствами λ-исчисления Чёрча. Сформулируйте тезис Чёрча. Что такое алгоритмически неразрешимая задача? Приведите пример.
Перечислите средства синхронизации и опишите их реализацию: атомарные переменные, мьютексы, семафоры.
Опишите метод градиентного спуска: понятие градиента и антиградиента, формулу шага, роль темпа обучения (learning rate) и критерии остановки итераций.
Охарактеризуйте разновидности градиентного спуска — пакетный, стохастический, мини-пакетный — и приёмы ускорения сходимости. В чём состоят проблемы локальных минимумов, седловых точек и плато?
Опишите модель искусственного нейрона: входы и синаптические веса, взвешенную сумму и смещение, функции активации. Как связано обучение нейрона с методом градиентного спуска?

Описание возможностей изучения дисциплины лицами с ОВЗ и инвалидами

При необходимости обучения студентов-инвалидов и лиц с ограниченными возможностями здоровья аудиторные занятия могут быть заменены или дополнены изучением полнотекстовых лекций, презентаций, видео- и аудиоматериалов в электронной информационно-образовательной среде (ЭИОС) университета. Индивидуальные задания подбираются в адаптированных к ограничениям здоровья формах (письменно или устно, в форме презентаций). Выбор методов обучения зависит от их доступности для инвалидов и лиц с ограниченными возможностями здоровья.

Рабочая программа дисциплины для инвалидов и лиц с ограниченными возможностями здоровья разрабатывается индивидуально с учетом особенностей психофизического развития, индивидуальных возможностей и состояния здоровья таких обучающихся.