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.

ФОНД ОЦЕНОЧНЫХ СРЕДСТВ ПО ДИСЦИПЛИНЕ (МОДУЛЮ) 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.

Фонд оценочных средств по дисциплине (модулю) составлен 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

1. Планируемые результаты обучения, соотнесенные с планируемыми результатами освоения образовательной программы

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.

2. Показатели и критерии оценивания компетенций

Планируемые результаты освоения программы	Уровень овладения	Критерии оценивания (перечень сформированных ИДК/ЗУВ по уровням освоения)
ОПК-7 Способен разрабатывать алгоритмы и программы, пригодные для практического применения	Повышенный	ИДК 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.

			ИДК 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 ИДК 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 ИДК 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.
	Базовый	ИДК 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. ИДК 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.

		ИДК 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 ИДК 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.
	Пороговый	ИДК 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.

		ИДК 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. ИДК 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. ИДК 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.

3. Шкала оценивания

Оценка	Критерии оценивания (перечень сформированных ИДК/ЗУВ)

Отлично	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 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. 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. 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 ОПК-7.2/Зн5 Классы алгоритмических стратегий; стандартные структуры данных (массив, список, дерево, граф, куча и др.); понятия сложности алгоритма. ОПК-7.2/Ум5 Конструировать алгоритмы с приемлемой асимптотикой, выбирать структуры данных под конкретную задачу. 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 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. ОПК-7.3/Ум5 Реализовывать алгоритмы в виде программных модулей, документировать код и соблюдать стандарты оформления 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 ОПК-7.4/Зн5 Методы и стратегии тестирования (юнит-тесты, граничные случаи); приёмы отладки и оптимизации; основные метрики производительности. ОПК-7.4/Ум5 Проводить тестирование, находить ошибки применять оптимизацию узких мест. 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.

Хорошо	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 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. 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. ОПК-7.2 Разрабатывает эффективные алгоритмы для решения задач, а также выбирает оптимальные структуры данных для их реализации ОПК-7.2/Зн5 Классы алгоритмических стратегий; стандартные структуры данных (массив, список, дерево, граф, куча и др.); понятия сложности алгоритма. ОПК-7.2/Ум5 Конструировать алгоритмы с приемлемой асимптотикой, выбирать структуры данных под конкретную задачу. ОПК-7.3 Пишет программный код на одном или нескольких языках программирования, следуя принципам модульности, читаемости, документирования и соответствия стандартам кодирования 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. ОПК-7.3/Ум5 Реализовывать алгоритмы в виде программных модулей, документировать код и соблюдать стандарты оформления 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 ОПК-7.4/Зн5 Методы и стратегии тестирования (юнит-тесты, граничные случаи); приёмы отладки и оптимизации; основные метрики производительности. ОПК-7.4/Ум5 Проводить тестирование, находить ошибки применять оптимизацию узких мест.
Удовлетворительно	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 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. ОПК-7.2 Разрабатывает эффективные алгоритмы для решения задач, а также выбирает оптимальные структуры данных для их реализации ОПК-7.2/Зн5 Классы алгоритмических стратегий; стандартные структуры данных (массив, список, дерево, граф, куча и др.); понятия сложности алгоритма.

	ОПК-7.3 Пишет программный код на одном или нескольких языках программирования, следуя принципам модульности, читаемости, документирования и соответствия стандартам кодирования 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. 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 ОПК-7.4/Зн5 Методы и стратегии тестирования (юнит-тесты, граничные случаи); приёмы отладки и оптимизации; основные метрики производительности.
Неудовле-творительно	Результаты обучения не сформированы на достаточном уровне

4. Контрольные мероприятия по дисциплине

Вид контроля	Форма контроля/Оценочное средство
Текущий контроль	Тестирование Доклад
Промежуточная аттестация	Экзамен

№ п/п	Наименование раздела	Контролируемые ИДК	Вид контроля/ используемые оценочные материалы	
			Текущий	Промежут. аттестация
1	Поянтие алгоритма. Методы оценки алгоритмов.	ОПК-7.1 ОПК-7.3	Тестирование	Экзамен
2	Методы оптимизации алгоритмов	ОПК-7.1 ОПК-7.2 ОПК-7.3	Тестирование Доклад	Экзамен
3	Введение в параллельные алгоритмы	ОПК-7.1 ОПК-7.2 ОПК-7.3	Тестирование	Экзамен
4	Приложение алгоритмов на примере графовых задач	ОПК-7.1 ОПК-7.2 ОПК-7.3 ОПК-7.4	Тестирование	Экзамен

5. 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.

6. 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

7. 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