КАЗАНСКИЙ КООПЕРАТИВНЫЙ ИНСТИТУТ (ФИЛИАЛ)
АВТОНОМНОЙ НЕКОММЕРЧЕСКОЙ ОБРАЗОВАТЕЛЬНОЙ
ОРГАНИЗАЦИИ ВЫСШЕГО ОБРАЗОВАНИЯ
ЦЕНТРОСОЮЗА РОССИЙСКОЙ ФЕДЕРАЦИИ
«РОССИЙСКИЙ УНИВЕРСИТЕТ КООПЕРАЦИИ»

ОЦЕНОЧНЫЕ МАТЕРИАЛЫ

МЕТОДЫ ПРОГРАММИРОВАНИЯ

Специальность: 10.05.01 Компьютерная безопасность
Специализация: «Разработка защищенного программного обеспечения»
Формы обучения: очная
Объем дисциплины:
в зачетных единицах: 5 з.е.
в академических часах: 180 ак.ч.

Форма промежуточной аттестации: экзамен

2025

Оценочные материалы по дисциплине Методы программирования

обсуждены и рекомендованы к утверждению решением кафедры естественных дисциплин, сервиса и туризма от 25 марта 2025 г., протокол № 6

одобрены Научно-методическим советом Казанского кооперативного института (филиала) от 2 апреля 2025 г., протокол № 3

СОДЕРЖАНИЕ

Целью оценочных материалов по дисциплине (модулю) (далее –ОМ) является комплексная оценка результатов обучения по дисциплине Методы программирования и установление уровня сформированности компетенций обучающегося.

ОМ предназначен для проведения текущего контроля успеваемости и промежуточной аттестации.

ОМ включает оценочные средства для проведения текущего контроля успеваемости и промежуточной аттестации.

Оценочные материалы разработаны в соответствии с рабочей программой дисциплины Методы программирования.

1. Паспорт оценочных материалов по дисциплине «Методы программирования»

Формируемые компетенции (код и наименование компетенции)	Индикаторы достижения компетенций	Планируемые результаты обучения	Контролируемые разделы, темы дисциплины	Наименование оценочного средства[1]	
				Текущий контроль	Промежуточная аттестация
1	2	3	4	5	6
ОПК-7 Способен создавать программы на языках высокого и низкого уровня, применять методы и инструментальные средства программирования для решения профессиональных задач, осуществлять обоснованный выбор инструментария программирования и способов организации программ	ОПК-7.3 Демонстрирует умение создавать программы на языках низкого уровня, при менять методы и инструментальные средства реверсинжиниринга для решения профессиональных задач	Знать: Типовые компьютерные методы и современное программное обеспечение для решения задач профессиональной деятельности на языке программирования высокого уровня С++. Уметь: Установить и использовать программное обеспечение для решения задач профессиональной деятельности, в частности Microsoft Visual Studio. Применять: методы компьютерного проектирования, разработки структур данных с использованием основных алгоритмических и программных решений в области информационно-коммуникационных технологий на языке высокого уровня С++. Владеть: Проектирование и использование структур данных при реализации алгоритмических и	Тема 1. Линейные динамические информационные структуры Тема 2. Стек. Очередь. Дек. Тема 3. Моделирование стека средствами языка С++. Тема 4. Моделирование очереди средствами языка С++. Тема 5. Кольцевой буфер Тема 6. Реализация двух однотипных стеков Тема 7. Однонаправленные списки. Операция вставки и удаления элемента	Реферат (тема 1-12), Тест (тема 1-12)	Тест (В1-14)

Формируемые компетенции (код и наименование компетенции)	Индикаторы достижения компетенций	Планируемые результаты обучения	Контролируемые разделы, темы дисциплины	Наименование оценочного средства[1]	
				Текущий контроль	Промежуточная аттестация
1	2	3	4	5	6
		программных решений на языке программирования высокого уровня С++.	Тема 8. Двунаправленные списки. Построение, операция удаления и вставки элемента Тема 9. Деревья. Дерево двоичного поиска. Обходы ДДП Тема 10. Обработка дерева двоичного поиска. Тема 11. Система управления вводом-выводом. Тема 12. Общие операции над файлами.		
	ОПК- 7.4 Демонстрирует умение создавать программы на языках высокого уровня, при менять методы визуального	Знать: типовые методы и приемы поиска, анализа и синтеза информации необходимой для решения профессиональных алгоритмических задач. Уметь: выполнять поиск, анализ и синтез научно-практической информации, полученной из	Тема 13. Ориентированные графы. Представления ориентированных графов. Тема 14. Задача нахождения кратчайшего пути.	Реферат (тема 13-24), Тест (тема 13-24)	Тест (В15-25)

Формируемые компетенции (код и наименование компетенции)	Индикаторы достижения компетенций	Планируемые результаты обучения	Контролируемые разделы, темы дисциплины	Наименование оценочного средства[1]	
				Текущий контроль	Промежуточная аттестация
1	2	3	4	5	6
	объектного моделирования в области разработки программ и соответствующие инструментальные средства в процессе решения профессиональных задач	достоверных источников. Применять результат поиска для решения профессиональных задач. Владеть: применение методов системного подхода для решения профессиональных задач, в частности в области разработки программного обеспечения.	Тема 15. Нахождение кратчайших путей между парами вершин. Алгоритм Флойда. Тема 16. Поиск центра ориентированного графа. Тема 17. Обход ориентированных графов. Поиск в глубину. Тема 18. Алгоритм нахождения сильно связных компонент. Тема 19. Представление неориентированных графов. Остовные деревья минимальной стоимости. Тема 20. Алгоритм Прима. Алгоритм Крускала.		

Формируемые компетенции (код и наименование компетенции)	Индикаторы достижения компетенций	Планируемые результаты обучения	Контролируемые разделы, темы дисциплины	Наименование оценочного средства[1]	
				Текущий контроль	Промежуточная аттестация
1	2	3	4	5	6
			Алгоритм Прима. Алгоритм Крускала. Реализация на C++. Тема 21. Контейнерный класс vector. Тема 22. Контейнерный класс list. Тема 23. Ассоциативные контейнеры set и map. Тема 24. Общая схема компиляции.		

2. Оценочные материалы по дисциплине «Методы программирования»

2.1 Оценочные материалы для проведения текущего контроля успеваемости

Тематика рефератов

Тема 1. Линейные динамические информационные структуры

1. Определение и классификация линейных динамических информационных структур.
2. Алгоритмы работы с линейными динамическими структурами данных: основные подходы и методы.
3. Сравнительный анализ линейных и нелинейных динамических структур данных.
4. Применение стеков и очередей в программировании: примеры и задачи.
5. Динамические массивы: преимущества и недостатки по сравнению со статическими.
6. Связанные списки: типы, структура и области применения.
7. Реализация и использование двусвязных списков в программировании.
8. Алгоритмы сортировки и их связь с линейными динамическими структурами.
9. Использование линейных динамических структур в базах данных.
10. Проблемы управления памятью при работе с динамическими структурами данных.
11. Применение линейных динамических структур в графических интерфейсах пользователя.
12. Стек вызовов: как работает и почему важен для программирования.
13. Очереди с приоритетом: концепция и применение в алгоритмах.
14. Линейные динамические структуры в языке Python: примеры и библиотеки.
15. Будущее линейных динамических структур: тенденции и новые технологии в области программирования.

Тема 2. Стек. Очередь. Дек.

1. Определение и основные характеристики стека: структура и принципы работы.
2. Очередь: принципы работы, типы и области применения.
3. Дек (двусторонняя очередь): особенности и преимущества по сравнению с обычными очередями.
4. Алгоритмы работы со стеком: реализация и примеры использования.
5. Очереди с приоритетом: как они работают и где применяются.
6. Сравнительный анализ стека, очереди и дека: когда использовать каждую структуру данных.
7. Применение стека в рекурсивных алгоритмах и решении задач.

8. Очереди в многопоточном программировании: управление задачами и ресурсами.

9. Динамические структуры данных: реализация стека, очереди и дека на языке Python.

10. Использование стека для обратной польской нотации (ОПН) в вычислениях.

11. Очереди в сетевых протоколах: как они обеспечивают передачу данных.

12. Дек как универсальная структура данных: примеры использования в алгоритмах.

13. Ошибки и проблемы при использовании стеков и очередей в программировании.

14. Реализация структур данных на основе массивов и связанных списков: плюсы и минусы.

15. Будущее стеков, очередей и деков в контексте современных технологий и языков программирования.

Тема 3. Моделирование стека средствами языка C++.

1. Основы структуры данных "Стек": определение и основные операции.

2. Реализация стека на основе массива в C++: алгоритмы и примеры.

3. Стек на основе связного списка: преимущества и недостатки по сравнению с массивом.

4. Использование стандартной библиотеки C++ для работы со стеком: контейнер std:stack.

5. Реализация динамического стека: управление памятью и обработка ошибок.

6. Алгоритмы, использующие стек: от простых до сложных задач.

7. Проверка сбалансированности скобок с помощью стека в C++.

8. Обратная польская нотация (ОПН): реализация вычислений с использованием стека.

9. Моделирование стека в многопоточном приложении на C++: синхронизация и безопасность.

10. Тестирование и отладка стека: методы и инструменты для C++.

11. Параметризованные шаблоны в C++ для создания универсального стека.

12. Сравнение производительности различных реализаций стека на C++.

13. Применение стека в алгоритмах обхода графов: реализация на C++.

14. Стек как средство реализации рекурсии в C++.

15. Будущее стека в контексте современных возможностей языка C++: концепции и улучшения.

Тема 4. Моделирование очереди средствами языка C++.

1. Основы структуры данных "Очередь": определение и основные операции.

2. Реализация очереди на основе массива в C++: алгоритмы и примеры.

3. Очередь на основе связного списка: преимущества и недостатки по сравнению с массивом.

4. Использование стандартной библиотеки C++ для работы с очередью: контейнер std:queue.

5. Динамическая очередь: управление памятью и обработка ошибок в C++.

6. Алгоритмы, использующие очередь: от простых до сложных задач.

7. Моделирование очереди с приоритетом: реализация и применение в C++.

8. Применение очереди в алгоритмах обхода графов: реализация на C++.

9. Очередь как средство реализации алгоритма "Ширина" (BFS) в графах.

10. Тестирование и отладка очереди: методы и инструменты для C++.

11. Параметризованные шаблоны в C++ для создания универсальной очереди.

12. Сравнение производительности различных реализаций очереди на C++.

13. Многопоточные приложения: синхронизация доступа к очереди в C++.

14. Реализация кольцевой очереди на C++: алгоритмы и примеры использования.

15. Будущее очереди в контексте современных возможностей языка C++: концепции и улучшения.

Тема 5. Кольцевой буфер

1. Определение и основные характеристики кольцевого буфера: принципы работы.

2. Структура данных "Кольцевой буфер": реализация на языке C++.

3. Сравнение кольцевого буфера с традиционными очередями: преимущества и недостатки.

4. Применение кольцевого буфера в системах реального времени: примеры и кейсы.

5. Алгоритмы работы с кольцевым буфером: вставка, удаление и чтение данных.

6. Кольцевой буфер в многопоточных приложениях: синхронизация и управление доступом.

7. Реализация кольцевого буфера для хранения потоковых данных: примеры на C++.

8. Оптимизация кольцевого буфера: использование памяти и производительность.

9. Использование кольцевого буфера в сетевых приложениях: обработка пакетов данных.

10. Кольцевой буфер в аудио- и видеопотоках: реализация и проблемы задержек.

11. Тестирование и отладка кольцевого буфера: методы и инструменты для C++.

12. Разработка универсального кольцевого буфера с использованием шаблонов C++.

13. Сравнение различных реализаций кольцевого буфера: на основе массивов и связных списков.

14. Кольцевой буфер как основа для реализации других структур данных: стек, очередь, дек.

15. Будущее кольцевых буферов: новые технологии и подходы в программировании.

Тема 6. Реализация двух однотипных стеков

1. Основные принципы работы стеков: сравнение с другими структурами данных.

2. Реализация двух стеков с использованием массива: алгоритмы и сложности.

3. Динамическая реализация стеков: использование связанных списков для двух стеков.

4. Сравнение производительности: массивы против связанных списков для реализации стеков.

5. Алгоритмы для реализации двух стеков в одном массиве: подходы и примеры.

6. Применение двух стеков в алгоритмах: решение задач на основе стека.

7. Проблемы и решения при реализации двух стеков: управление памятью и производительность.

8. Реализация двух стеков в многопоточных приложениях: синхронизация и безопасность.

9. Использование шаблонов C++ для создания универсальных стеков: реализация двух однотипных стеков.

10. Кейс-стадии: примеры использования двух стеков в реальных приложениях.

11. Тестирование и отладка реализации двух стеков: методы и инструменты.

12. Оптимизация памяти при реализации двух стеков: стратегии и лучшие практики.

13. Два стека для реализации очереди: алгоритмы и примеры кода.

14. Визуализация работы двух стеков: графические инструменты и методы.

15. Будущее стековых структур данных: новые подходы и технологии в программировании.

Тема 7. Однонаправленные списки. Операция вставки и удаления элемента

1. Основные принципы работы однонаправленных списков: структура и преимущества.

2. Сравнение однонаправленных списков с массивами: когда и почему использовать.

3. Алгоритм вставки элемента в однонаправленный список: пошаговое руководство.

4. Удаление элемента из однонаправленного списка: алгоритмы и примеры.

5. Вставка и удаление в однонаправленном списке: анализ временной сложности.

6. Реализация однонаправленного списка на языке программирования Python: примеры кода.

7. Динамическое выделение памяти для однонаправленных списков: управление памятью и оптимизация.

8. Проблемы при вставке и удалении элементов из однонаправленного списка: распространенные ошибки.

9. Различные подходы к реализации операций вставки и удаления в однонаправленных списках.

10. Использование однонаправленных списков в реальных приложениях: кейс-стадии и примеры.

11. Визуализация операций вставки и удаления в однонаправленных списках: инструменты и методы.

12. Сравнительный анализ производительности операций вставки и удаления в различных структурах данных.

13. Расширенные операции над однонаправленными списками: реверсирование, сортировка и объединение.

14. Однонаправленные списки в многопоточных приложениях: проблемы синхронизации при вставке и удалении.

15. Будущее однонаправленных списков: новые технологии и подходы в обработке данных.

Тема 8. Двунаправленные списки. Построение, операция удаления и вставки элемента

1. Основные характеристики двунаправленных списков: структура и преимущества по сравнению с однонаправленными.

2. Алгоритм построения двунаправленного списка: пошаговое руководство.

3. Вставка элемента в двунаправленный список: алгоритмы и примеры кода.

4. Удаление элемента из двунаправленного списка: пошаговая инструкция и примеры.

5. Сравнение временной сложности операций вставки и удаления в двунаправленных и однонаправленных списках.

6. Реализация двунаправленного списка на языке C++: практические примеры.

7. Динамическое выделение памяти для двунаправленных списков: управление памятью и оптимизация.

8. Проблемы и ошибки при реализации операций вставки и удаления в двунаправленных списках.

9. Расширенные операции над двунаправленными списками: реверсирование, сортировка и объединение списков.

10. Применение двунаправленных списков в реальных приложениях: кейс-стадии и примеры использования.

11. Визуализация операций вставки и удаления в двунаправленных списках: инструменты для обучения.

12. Сравнительный анализ производительности двунаправленных списков с другими структурами данных (массивы, стеки, очереди).

13. Многопоточное программирование и работа с двунаправленными списками: проблемы синхронизации при вставке и удалении элементов.

14. Будущее двунаправленных списков: новые технологии и подходы в обработке данных.

15. Примеры использования двунаправленных списков в алгоритмах и структурах данных: стек, очередь, графы.

Тема 9. Деревья. Дерево двоичного поиска. Обходы ДДП

1. Основные характеристики деревьев и их применение в информатике.

2. Структура и свойства дерева двоичного поиска: основные понятия и определения.

3. Алгоритмы вставки и удаления узлов в дереве двоичного поиска: пошаговое руководство.

4. Обходы дерева двоичного поиска: ин-order, pre-order и post-order — алгоритмы и примеры.

5. Сравнительный анализ временной сложности операций в дереве двоичного поиска.

6. Реализация дерева двоичного поиска на языке Python: практические примеры.

7. Балансировка дерева двоичного поиска: AVL-деревья и красно-черные деревья.

8. Использование деревьев двоичного поиска в задачах поиска и сортировки данных.

9. Проблемы и ошибки при реализации операций вставки и удаления в дереве двоичного поиска.

10. Обходы дерева: алгоритмы и их применение в практических задачах.

11. Применение дерева двоичного поиска в системах управления базами данных (СУБД).

12. Визуализация работы алгоритмов обхода дерева двоичного поиска: инструменты и методы.

13. Сравнение дерева двоичного поиска с другими структурами данных: хеш-таблицы, массивы, списки.

14. Многопоточное программирование и работа с деревьями двоичного поиска: проблемы синхронизации.

15. Будущее деревьев двоичного поиска: новые технологии и подходы в обработке данных.

Тема 10. Обработка дерева двоичного поиска.

1. Основы обработки деревьев двоичного поиска: структура и свойства.
2. Алгоритмы обхода дерева двоичного поиска: сравнительный анализ и применение.
3. Визуализация обходов дерева двоичного поиска: инструменты и методы.
4. Эффективность различных алгоритмов обхода дерева двоичного поиска: время выполнения и память.
5. Реализация алгоритмов обхода (in-order, pre-order, post-order) на языке Python.
6. Обход дерева двоичного поиска в контексте рекурсивных и итеративных подходов.
7. Проблемы и ошибки при реализации обходов дерева двоичного поиска: как их избежать?
8. Практическое применение обходов дерева двоичного поиска в задачах сортировки и поиска.
9. Обходы дерева двоичного поиска и их использование в алгоритмах сжатия данных.
10. Сравнение обходов дерева двоичного поиска с обходами других структур данных (например, графов).
11. Оптимизация алгоритмов обхода дерева двоичного поиска для больших объемов данных.
12. Роль обходов дерева двоичного поиска в построении выражений и вычислениях в математических системах.
13. Использование многопоточности при обработке деревьев двоичного поиска: вызовы и решения.
14. Обходы дерева двоичного поиска в контексте баз данных: индексация и запросы.
15. Будущее обработки деревьев двоичного поиска: новые подходы и технологии в алгоритмах обработки данных.

Тема 11. Система управления вводом-выводом. Буферизация ввода-вывода

1. Основы систем управления вводом-выводом: архитектура и компоненты.
2. Роль буферизации в системах ввода-вывода: преимущества и недостатки.
3. Типы буферов: статические и динамические буферы ввода-вывода.
4. Алгоритмы управления буферизацией: FIFO, LIFO и другие методы.
5. Влияние буферизации на производительность систем ввода-вывода.
6. Сравнительный анализ различных стратегий буферизации в операционных системах.
7. Буферизация ввода-вывода в контексте многозадачности и параллелизма.

8. Реализация систем управления вводом-выводом в современных операционных системах (Windows, Linux).

9. Программные интерфейсы для работы с буферизацией ввода-вывода: POSIX и другие стандарты.

10. Ошибки и проблемы, возникающие при буферизации ввода-вывода: диагностика и решение.

11. Буферизация ввода-вывода в распределенных системах: вызовы и решения.

12. Использование кэширования как метода оптимизации ввода-вывода: сравнение с буферизацией.

13. Будущее систем управления вводом-выводом: новые технологии и подходы к буферизации.

14. Влияние аппаратного обеспечения на эффективность систем ввода-вывода и буферизацию.

15. Практическое применение буферизации ввода-вывода в реальных приложениях: примеры и кейсы.

Тема 12. Общие операции над файлами

1. Основные операции над файлами: создание, открытие, чтение и запись.

2. Файловые системы: структура и принципы работы с файлами.

3. Различия между текстовыми и бинарными файлами: особенности операций.

4. Работа с файловыми дескрипторами в операционных системах.

5. Методы обработки ошибок при работе с файлами: лучшие практики.

6. Параллельный доступ к файлам: проблемы и решения.

7. Безопасность файловых операций: управление доступом и шифрование.

8. Резервное копирование и восстановление файлов: стратегии и инструменты.

9. Оптимизация операций над файлами: кэширование и буферизация.

10. Использование библиотек для работы с файлами в различных языках программирования (например, Python, Java).

11. Файловые форматы: стандарты и их влияние на операции над файлами.

12. Сравнительный анализ систем управления версиями файлов (например, Git).

13. Влияние аппаратного обеспечения на производительность операций с файлами.

14. Работа с большими файлами: подходы к эффективному чтению и записи данных.

15. Будущее управления файлами: облачные технологии и распределенные файловые системы.

Тема 13. Ориентированные графы. Представления ориентированных графов.

1. Введение в ориентированные графы: определения и основные свойства.

2. Различия между ориентированными и неориентированными графами: анализ и примеры.

3. Матрицы смежности: представление ориентированных графов и их использование.

4. Списки смежности: преимущества и недостатки в представлении ориентированных графов.

5. Обход ориентированного графа: алгоритмы DFS и BFS.

6. Топологическая сортировка ориентированного графа: алгоритмы и применения.

7. Проблема поиска кратчайшего пути в ориентированных графах: алгоритм Дейкстры.

8. Алгоритм Флойда-Уоршелла для нахождения всех пар кратчайших путей в ориентированных графах.

9. Циклы в ориентированных графах: определение, обнаружение и их значение.

10. Сильные компоненты связности в ориентированных графах: алгоритм Косарайю.

11. Применение ориентированных графов в реальных задачах: от сетей до социальных медиа.

12. Алгоритмы для нахождения максимального потока в ориентированных графах: алгоритм Форда-Фалкерсона.

13. Графы с весами: как веса влияют на операции над ориентированными графами.

14. Использование ориентированных графов в машинном обучении и анализе данных.

15. Будущее исследований в области ориентированных графов: новые методы и технологии.

Тема 14. Задача нахождения кратчайшего пути.

1. Введение в задачу нахождения кратчайшего пути: основные определения и концепции.

2. Алгоритм Дейкстры: принципы работы и применение для нахождения кратчайшего пути.

3. Алгоритм Беллмана-Форда: преимущества и недостатки по сравнению с алгоритмом Дейкстры.

4. Алгоритм Флойда-Уоршелла: решение задачи о кратчайших путях для всех пар вершин.

5. Применение алгоритма A*: эвристические методы для нахождения кратчайшего пути.

6. Сравнительный анализ алгоритмов нахождения кратчайшего пути: эффективность и сложность.

7. Нахождение кратчайшего пути в графах с отрицательными весами: алгоритм Беллмана-Форда.

8. Применение кратчайших путей в навигационных системах и геоинформационных системах.

9. Оптимизация маршрутов в транспортных сетях: использование алгоритмов кратчайшего пути.

10. Кратчайшие пути в динамических графах: методы и алгоритмы для изменяющихся данных.

11. Графы с ограничениями: нахождение кратчайшего пути с учетом дополнительных условий.

12. Использование параллельных вычислений для ускорения алгоритмов нахождения кратчайшего пути.

13. Кратчайшие пути в сетях связи: применение в телекоммуникациях и интернет-протоколах.

14. Методы визуализации результатов алгоритмов нахождения кратчайшего пути.

15. Будущее задач нахождения кратчайшего пути: новые подходы и исследования в области графов.

Тема 15. Нахождение кратчайших путей между парами вершин. Алгоритм Флойда.

1. Обзор алгоритма Флойда-Уоршелла: принципы работы и основные характеристики.

2. Сравнение алгоритма Флойда с другими алгоритмами нахождения кратчайших путей: Дейкстра и Беллмана-Форда.

3. Применение алгоритма Флойда в задачах оптимизации маршрутов.

4. Алгоритм Флойда для нахождения кратчайших путей в графах с отрицательными весами.

5. Эффективность алгоритма Флойда: временная и пространственная сложность.

6. Визуализация работы алгоритма Флойда на примере графа.

7. Расширение алгоритма Флойда: применение для нахождения кратчайших путей в динамических графах.

8. Алгоритм Флойда в контексте теории графов: связь с другими концепциями.

9. Применение алгоритма Флойда в реальных задачах: примеры из логистики и транспортных систем.

10. Оптимизация алгоритма Флойда: улучшения и современные подходы.

11. Сравнительный анализ алгоритма Флойда и алгоритма Джонсона для разреженных графов.

12. Алгоритм Флойда в компьютерной сети: применение для нахождения кратчайших маршрутов.

13. Анализ сложности алгоритма Флойда при больших объемах данных: проблемы и решения.

14. Исторический аспект: развитие алгоритма Флойда и его влияние на теорию графов.

15. Будущее алгоритма Флойда: новые исследования и направления в области нахождения кратчайших путей.

Тема 16. Поиск центра ориентированного графа.

1. Определение и свойства центра ориентированного графа: основные понятия.
2. Методы нахождения центра ориентированного графа: алгоритмы и подходы.
3. Сравнение центра ориентированного графа с центром неориентированного графа: ключевые отличия.
4. Применение центра ориентированного графа в сетевых технологиях и теории игр.
5. Алгоритм для нахождения центра ориентированного графа: пошаговый анализ.
6. Влияние структуры графа на поиск центра: анализ различных типов ориентированных графов.
7. Центр ориентированного графа и его применение в задачах маршрутизации.
8. Оптимизация алгоритмов поиска центра ориентированного графа: современные подходы и исследования.
9. Изучение центров ориентированных графов в контексте социальных сетей.
10. Центр ориентированного графа в контексте теории когнитивных карт и моделирования знаний.
11. Сравнительный анализ различных алгоритмов для нахождения центра ориентированного графа: преимущества и недостатки.
12. Применение центра ориентированного графа в логистике и управлении цепочками поставок.
13. Анализ сложности алгоритмов нахождения центра ориентированного графа: временные и пространственные характеристики.
14. Исторический аспект изучения центров графов: от классических определений до современных исследований.
15. Будущее исследований центра ориентированного графа: новые направления и открытые вопросы.

Тема 17. Обход ориентированных графов. Поиск в глубину.

1. Основные принципы обхода ориентированных графов: алгоритмы и их применение.
2. Поиск в глубину (DFS): алгоритм, его реализация и особенности.
3. Сравнение поиска в глубину и поиска в ширину: преимущества и недостатки каждого подхода.
4. Применение поиска в глубину для решения задач на графах: примеры и кейсы.
5. Анализ временной и пространственной сложности алгоритма поиска в глубину.
6. Рекурсивная vs. итеративная реализация поиска в глубину: плюсы и минусы.

7. Поиск в глубину как инструмент для топологической сортировки ориентированных графов.

8. Поиск в глубину и его использование в задачах нахождения компонент связности в ориентированных графах.

9. Модификации алгоритма поиска в глубину: применение в специализированных задачах.

10. Поиск в глубину для нахождения кратчайших путей в ориентированных графах: анализ методов.

11. Применение поиска в глубину в искусственном интеллекте: игры и алгоритмы принятия решений.

12. Обход ориентированных графов в контексте социальных сетей: анализ взаимодействий и связей.

13. Поиск в глубину и его роль в анализе сетевых структур: от компьютерных сетей до биологических систем.

14. Графы с циклами: особенности поиска в глубину и методы их обработки.

15. Будущее исследований обхода ориентированных графов: новые направления и вызовы для алгоритмов.

Тема 18. Алгоритм нахождения сильно связных компонент.

1. Основные принципы алгоритма нахождения сильно связных компонент в ориентированных графах.

2. Алгоритм Косарайю: пошаговый разбор и применение для нахождения сильно связных компонент.

3. Алгоритм Тарьяна: детали реализации и его преимущества по сравнению с другими методами.

4. Сравнение алгоритмов нахождения сильно связных компонент: Косарайю vs. Тарьяна.

5. Применение сильно связных компонент в анализе социальных сетей: выявление групп пользователей.

6. Сильно связные компоненты и их использование в оптимизации маршрутов в транспортных системах.

7. Анализ временной и пространственной сложности алгоритмов нахождения сильно связных компонент.

8. Графы с циклами: особенности нахождения сильно связных компонент и их обработка.

9. Применение алгоритма нахождения сильно связных компонент в теории игр: анализ стратегий.

10. Сильно связные компоненты и их роль в анализе зависимостей в программном обеспечении.

11. Алгоритмы нахождения сильно связных компонент в динамических графах: вызовы и решения.

12. Визуализация сильно связных компонент: методы и инструменты для анализа графов.

13. Изучение свойств сильно связных компонент: теоретические аспекты и практические примеры.

14. Сильно связные компоненты в контексте биологических сетей: анализ взаимодействий между молекулами.

15. Будущее исследований в области нахождения сильно связных компонент: новые направления и технологии.

Тема 19. Представление неориентированных графов. Основные деревья минимальной стоимости.

1. Основные способы представления неориентированных графов: матрицы смежности и списки смежности.

2. Сравнение представлений неориентированных графов: преимущества и недостатки различных методов.

3. Алгоритм Краскала для нахождения основного дерева минимальной стоимости: пошаговый разбор.

4. Алгоритм Прима: реализация и применение для поиска основного дерева минимальной стоимости.

5. Сравнение алгоритмов Краскала и Прима: когда использовать каждый из них?

6. Применение основных деревьев минимальной стоимости в сетевых технологиях: примеры из реальной жизни.

7. Влияние структуры графа на эффективность алгоритмов нахождения основного дерева минимальной стоимости.

8. Алгоритмы для нахождения основных деревьев в динамических графах: вызовы и решения.

9. Методы оптимизации алгоритмов нахождения основных деревьев: улучшение производительности.

10. Графы с весами: как выбрать оптимальные веса для нахождения основного дерева минимальной стоимости?

11. Применение основных деревьев в задачах маршрутизации: от теории к практике.

12. Анализ временной и пространственной сложности алгоритмов нахождения основных деревьев.

13. Генерация случайных графов и их применение для тестирования алгоритмов нахождения основных деревьев.

14. Изучение свойств основных деревьев: теоретические аспекты и практические примеры.

15. Будущее исследований в области основных деревьев минимальной стоимости: новые направления и технологии.

Тема 20. Алгоритм Прима. Алгоритм Крускала.

1. Обзор алгоритмов Прима и Крускала: основные принципы и подходы.

2. Алгоритм Прима: пошаговый анализ и его реализация на примере.

3. Алгоритм Крускала: пошаговый анализ и его реализация на примере.

4. Сравнительный анализ алгоритмов Прима и Крускала: преимущества и недостатки каждого.

5. Применение алгоритма Прима в задачах, связанных с сетями и графами.

6. Применение алгоритма Крускала в задачах, связанных с построением минимальных сетей.

7. Влияние структуры графа на выбор алгоритма для нахождения основного дерева минимальной стоимости.

8. Оптимизация алгоритма Прима: использование различных структур данных для повышения эффективности.

9. Оптимизация алгоритма Крускала: применение методов объединения и поиска для повышения производительности.

10. Алгоритмы Прима и Крускала в динамических графах: вызовы и стратегии решения.

11. Реализация алгоритмов Прима и Крускала на различных языках программирования: примеры и сравнение.

12. Графы с весами: как правильно задавать веса для применения алгоритмов Прима и Крускала?

13. Применение алгоритмов Прима и Крускала в реальных приложениях: примеры из телекоммуникаций и транспортных систем.

14. Анализ временной и пространственной сложности алгоритмов Прима и Крускала: что выбрать в зависимости от задачи?

15. Будущее исследований в области основных деревьев: новые подходы к алгоритмам Прима и Крускала.

Тема 21. Контейнерный класс vector.

1. Обзор контейнерного класса vector: основные характеристики и преимущества.

2. Сравнение контейнерного класса vector с другими контейнерами STL: list, deque и array.

3. Структура данных vector: как она устроена и как работает под капотом.

4. Алгоритмы работы с vector: добавление, удаление и поиск элементов.

5. Эффективность использования vector: временные и пространственные характеристики операций.

6. Инициализация и заполнение контейнера vector: различные подходы и методы.

7. Использование vector с пользовательскими типами данных: шаблоны и перегрузка операторов.

8. Проблемы и решения при работе с большими объемами данных в vector.

9. Перемещение и копирование объектов в vector: семантика перемещения и копирования.

10. Параллельная обработка данных в vector: использование многопоточности для повышения производительности.

11. Контейнер vector в контексте управления памятью: аллокация и деаллокация.

12. Практические примеры использования vector в реальных приложениях: от алгоритмов до графических интерфейсов.

13. Ошибки и подводные камни при работе с vector: типичные проблемы и их решения.

14. Расширенные функции vector: использование методов reserve(), resize() и других.

15. Будущее контейнеров в C++: изменения и улучшения, касающиеся класса vector в новых стандартах языка.

Тема 22. Контейнерный класс list.

1. Обзор контейнерного класса list: основные характеристики и преимущества.

2. Сравнение контейнерного класса list с другими контейнерами STL: vector, deque и array.

3. Структура данных list: как она устроена и как работает под капотом.

4. Алгоритмы работы с list: добавление, удаление и поиск элементов.

5. Эффективность использования list: временные и пространственные характеристики операций.

6. Инициализация и заполнение контейнера list: различные подходы и методы.

7. Использование list с пользовательскими типами данных: шаблоны и перегрузка операторов.

8. Проблемы и решения при работе с большими объемами данных в list.

9. Перемещение и копирование объектов в list: семантика перемещения и копирования.

10. Параллельная обработка данных в list: использование многопоточности для повышения производительности.

11. Контейнер list в контексте управления памятью: аллокация и деаллокация.

12. Практические примеры использования list в реальных приложениях: от алгоритмов до графических интерфейсов.

13. Ошибки и подводные камни при работе с list: типичные проблемы и их решения.

14. Расширенные функции list: использование методов splice(), merge() и других.

15. Будущее контейнеров в C++: изменения и улучшения, касающиеся класса list в новых стандартах языка.

Тема 23. Ассоциативные контейнеры set и map.

1. Обзор ассоциативных контейнеров set и map: основные характеристики и отличия.

2. Структура данных set и map: как они устроены и как работают под капотом.

3. Сравнение производительности set и map с другими контейнерами STL: преимущества и недостатки.

4. Алгоритмы работы с set: добавление, удаление и поиск уникальных элементов.

5. Использование map для хранения пар ключ-значение: как эффективно организовать данные.

6. Пользовательские компараторы в set и map: настройка порядка хранения элементов.

7. Перебор элементов в ассоциативных контейнерах: итераторы и диапазоны.

8. Проблемы с дублированием ключей в map и уникальностью в set: как их избежать.

9. Расширенные функции set и map: методы find(), count(), lower_bound() и другие.

10. Применение ассоциативных контейнеров в алгоритмах: от поиска до сортировки.

11. Эффективное использование памяти в set и map: аллокация, деаллокация и управление ресурсами.

12. Параллельная обработка данных в ассоциативных контейнерах: использование многопоточности.

13. Ошибки и подводные камни при работе с set и map: типичные проблемы и их решения.

14. Использование set и map с пользовательскими типами данных: шаблоны и перегрузка операторов.

15. Будущее ассоциативных контейнеров в C++: изменения и улучшения, касающиеся set и map в новых стандартах языка.

Тема 24. Общая схема компиляции.

1. Этапы компиляции: от исходного кода до исполняемого файла.

2. Лексический анализ: как компилятор разбивает код на токены.

3. Синтаксический анализ: построение дерева разбора и его значение.

4. Семантический анализ: проверка корректности типов и контекста.

5. Оптимизация кода: виды оптимизаций и их влияние на производительность.

6. Генерация промежуточного кода: зачем и как это делается.

7. Компиляция в машинный код: как компилятор преобразует промежуточный код.

8. Сборка и линковка: объединение объектных файлов в исполняемую программу.

9. Ошибки компиляции: типы ошибок и методы их устранения.

10. Инструменты для разработки компиляторов: парсеры, генераторы кода и другие утилиты.

11. Различия между компиляторами и интерпретаторами: преимущества и недостатки каждого подхода.

12. Роль статической и динамической типизации в процессе компиляции.

13. Современные компиляторы: примеры (GCC, Clang, MSVC) и их особенности.

14. Компиляция для различных платформ: кросс-компиляция и ее вызовы.

15. Будущее компиляции: новые подходы и технологии в разработке компиляторов.

Критерии оценки выполнения рефератов по учебной дисциплине

Оценка «отлично» выставляется студенту, если тема реферата раскрыта в полной мере и все требования по написанию реферата соблюдены.

Оценка «хорошо» выставляется студенту, если недостаточно раскрыта тема реферата, но все требования по написанию реферата соблюдены.

Оценка «удовлетворительно» выставляется студенту, если недостаточно раскрыта тема реферата, не все требования по написанию реферата соблюдены, присутствуют ошибки и неточности в работе.

Оценка «неудовлетворительно» выставляется студенту, если отсутствует реферат.

Рекомендации по написанию реферата:

Реферат — письменная работа, выполняемая обучающимся в течение определенного срока.

Реферат — краткое точное изложение сущности какого-либо вопроса, темы на основе одной или нескольких книг, монографий или других первоисточников. Реферат должен содержать основные фактические сведения и выводы по рассматриваемому вопросу.

Реферат отвечает на вопрос — что содержится в данной публикации (публикациях). Реферат — не механический пересказ работы, а изложение ее существа.

Кроме реферирования прочитанной литературы, от обучающегося требуется аргументированное изложение собственных мыслей по рассматриваемому вопросу. Тему реферата предлагает преподаватель или сам обучающийся, в последнем случае она должна быть согласованна с преподавателем.

В реферате нужны развернутые аргументы, рассуждения, сравнения. Материал подается не столько в развитии, сколько в форме констатации или описания.

Содержание реферируемого произведения излагается объективно от имени автора. Если в первичном документе главная мысль сформулирована недостаточно четко, в реферате она должна быть конкретизирована и выделена.

Структура реферата:

1. Титульный лист.

2. После титульного листа на отдельной странице следует оглавление (план, содержание), в котором указаны названия всех разделов (пунктов плана) реферата и номера страниц, указывающие начало этих разделов в тексте

реферата.

3. После оглавления следует введение. Объем введения составляет 1,5-2 страницы.

4. Основная часть реферата может иметь одну или несколько глав, состоящих из 2-3 параграфов (подпунктов, разделов) и предполагает осмысленное и логичное изложение главных положений и идей, содержащихся в изученной литературе. В тексте обязательны ссылки на первоисточники. В том случае если цитируется или используется чья-либо неординарная мысль, идея, вывод, приводится какой-либо цифрой материал, таблицу - обязательно сделайте ссылку на того автора у кого вы взяли данный материал.

5. Заключение содержит главные выводы, и итоги из текста основной части, в нем отмечается, как выполнены задачи и достигнуты ли цели, сформулированные во введении.

6. Приложение может включать графики, таблицы, расчеты.

7. Библиография (список литературы) здесь указывается реально использованная для написания реферата литература. Список составляется согласно правилам библиографического описания.

Этапы работы над рефератом.

Работу над рефератом можно условно подразделить на три этапа:

1. Подготовительный этап, включающий изучение предмета исследования;

2. Изложение результатов изучения в виде связного текста;

3. Устное сообщение по теме реферата.

1. Подготовительный этап работы.

Формулировка темы. Подготовительная работа над рефератом начинается с формулировки темы. Тема в концентрированном виде выражает содержание будущего текста, фиксируя как предмет исследования, так и его ожидаемый результат. Для того чтобы работа над рефератом была успешной, необходимо, чтобы тема заключала в себе проблему, скрытый вопрос (даже если наука уже давно дала ответ на этот вопрос, студент, только знакомящийся с соответствующей областью знаний, будет вынужден искать ответ заново, что даст толчок к развитию проблемного, исследовательского мышления).

Поиск источников. Грамотно сформулированная тема зафиксировала предмет изучения; задача обучающегося — найти информацию, относящуюся к данному предмету и разрешить поставленную проблему. Выполнение этой задачи начинается с поиска источников. На этом этапе необходимо вспомнить, как работать с энциклопедиями и энциклопедическими словарями.

Работа с источниками. Работу с источниками надо начинать с ознакомительного чтения, т. е. просмотреть текст, выделяя его структурные единицы. При ознакомительном чтении закладками отмечаются те страницы, которые требуют более внимательного изучения. В зависимости от результатов ознакомительного чтения выбирается дальнейший способ работы с источником. Если для разрешения поставленной задачи требуется изучение некоторых фрагментов текста, то используется метод выборочного чтения. Если в книге нет подробного оглавления, следует обратить внимание на

предметные и именные указатели.

Избранные фрагменты или весь текст (если он целиком имеет отношение к теме) требуют вдумчивого, неторопливого чтения с «мысленной проработкой» материала. Такое чтение предполагает выделение: 1) главного в тексте; 2) основных аргументов; 3) выводов. Особое внимание следует обратить на то, вытекает тезис из аргументов или нет. Необходимо также проанализировать, какие из утверждений автора носят проблематичный, гипотетический характер и уловить скрытые вопросы.

Понятно, что умение таким образом работать с текстом приходит далеко не сразу. Наилучший способ научиться выделять главное в тексте, улавливать проблематичный характер утверждений, давать оценку авторской позиции — это сравнительное чтение, в ходе которого студент знакомится с различными мнениями по одному и тому же вопросу, сравнивает весомость и доказательность аргументов сторон и делает вывод о наибольшей убедительности той или иной позиции.

Создание конспектов для написания реферата.

Подготовительный этап работы завершается созданием конспектов, фиксирующих основные тезисы и аргументы. Здесь важно вспомнить, что конспекты пишутся на одной стороне листа, с полями и достаточным для исправления и ремарок межстрочным расстоянием (эти правила соблюдаются для удобства редактирования). Если в конспектах приводятся цитаты, то непременно должно быть дано указание на источник (автор, название, выходные данные, № страницы).

По завершении предварительного этапа можно переходить непосредственно к созданию текста реферата.

2. Создание текста.

Общие требования к тексту.

Текст реферата должен подчиняться определенным требованиям: он должен раскрывать тему, обладать связностью и цельностью.

Раскрытие темы предполагает, что в тексте реферата излагается относящийся к теме материал и предлагаются пути решения содержащейся в теме проблемы; связность текста предполагает смысловую соотносительность отдельных компонентов, а цельность - смысловую законченность текста.

С точки зрения связности все тексты делятся на тексты-констатации и тексты-рассуждения. Тексты-констатации содержат результаты ознакомления с предметом и фиксируют устойчивые и несомненные суждения. В текстах-рассуждениях одни мысли извлекаются из других, некоторые ставятся под сомнение, дается им оценка, выдвигаются различные предположения.

План реферата.

Универсальный план реферата – введение, основной текст и заключение.

Требования к введению.

Во введении аргументируется актуальность исследования, - т. е. выявляется практическое и теоретическое значение данного исследования. Далее констатируется, что сделано в данной области предшественниками; перечисляются положения, которые должны быть обоснованы. Введение может

также содержать обзор источников или экспериментальных данных, уточнение исходных понятий и терминов, сведения о методах исследования. Во введении обязательно формулируются цель и задачи реферата.

Объем введения - в среднем около 10% от общего объема реферата.

Основная часть реферата.

Основная часть реферата раскрывает содержание темы. Она наиболее значительна по объему, наиболее значима и ответственна. В ней обосновываются основные тезисы реферата, приводятся развернутые аргументы, предполагаются гипотезы, касающиеся существа обсуждаемого вопроса. Важно проследить, чтобы основная часть не имела форму монолога. Аргументируя собственную позицию, можно и должно анализировать, и оценивать позиции различных исследователей, с чем-то соглашаться, чему-то возражать, кого-то опровергать. Текст основной части делится на главы, параграфы, пункты. План основной части может быть составлен с использованием различных методов группировки материала: классификации (эмпирические исследования), типологии (теоретические исследования), периодизации (исторические исследования).

Заключение.

Заключение — последняя часть научного текста. В ней краткой и сжатой форме излагаются полученные результаты, представляющие собой ответ на главный вопрос исследования. Здесь же могут намечаться и дальнейшие перспективы развития темы. Небольшое по объему сообщение также не может обойтись без заключительной части - пусть это будут две-три фразы. Но в них должен подводиться итог проделанной работы.

Список использованной литературы.

Реферат любого уровня сложности обязательно сопровождается списком используемой литературы. Названия книг в списке располагают по алфавиту с указанием выходных данных использованных книг.

Требования, предъявляемыек оформлению реферата

Объемы рефератов – от 10-18 машинописных страниц. Работа выполняется на одной стороне листа стандартного формата. По обеим сторонам листа оставляются поля размером 35 мм. слева и 15 мм. справа, рекомендуется шрифт 12-14, интервал - 1,5. Все листы реферата должны быть пронумерованы. Каждый вопрос в тексте должен иметь заголовок в точном соответствии с наименованием в плане-оглавлении. При написании и оформлении реферата следует избегать типичных ошибок, например, таких:

– поверхностное изложение основных теоретических вопросов выбранной темы, когда автор не понимает, какие проблемы в тексте являются главными, а какие второстепенными,

– в некоторых случаях проблемы, рассматриваемые в разделах, не раскрывают основных аспектов выбранной для реферата темы,

– дословное переписывание книг, статей, заимствования рефератов из интернета и т. д.

При проверке реферата преподавателем оцениваются:

1. Знания и умения на уровне требований стандарта конкретной

дисциплины: знание фактического материала, усвоение общих представлений, понятий, идей.

2. Характеристика реализации цели и задач исследования (новизна и актуальность поставленных в реферате проблем, правильность формулирования цели, определения задач исследования, правильность выбора методов решения задач и реализации цели; соответствие выводов решаемым задачам, поставленной цели, убедительность выводов).

3. Степень обоснованности аргументов и обобщений (полнота, глубина, всесторонность раскрытия темы, логичность и последовательность изложения материала, корректность аргументации и системы доказательств, характер и достоверность примеров, иллюстративного материала, широта кругозора автора, наличие знаний интегрированного характера, способность к обобщению).

4. Качество и ценность полученных результатов (степень завершенности реферативного исследования, спорность или однозначность выводов).

5. Использование литературных источников.

6. Культура письменного изложения материала.

7. Культура оформления материалов работы.

Комплект типовых практических заданий

Тема 1. Линейные динамические информационные структуры

Определение линейных динамических структур данных (например, списки, стеки и очереди).

Ключевые операции с линейными структурами: добавление, удаление, поиск и обход.

Сравнение статических и динамических структур данных: преимущества и недостатки.

Реализация линейных структур данных на выбранном языке программирования (например, Python, C++, Java).

Примеры практического применения линейных динамических структур данных в реальных задачах (например, обработка данных, управление задачами).

Вопросы для самоконтроля:

1. Что такое линейная динамическая структура данных?

2. В чем разница между стеком и очередью?

3. Каковы основные операции, поддерживаемые линейными структурами данных?

4. Какие преимущества имеют динамические структуры данных по сравнению со статическими?

5. Приведите примеры задач, где использование линейных динамических структур данных является оптимальным решением.

Цель работы – формирование практических навыков работы с линейными динамическими структурами данных и их реализация на выбранном языке программирования.

В работе используется презентация слайдов, выполненная в MS Powerpoint

Критерии оценивания выполнения практического задания:

Оценка «отлично» выставляется, если: работа выполнена в полном объеме с соблюдением необходимой последовательности действий; работа проведена в условиях, обеспечивающих получение правильных результатов и выводов; соблюдены правила техники безопасности; в ответе правильно и аккуратно выполняет все записи, таблицы, рисунки, чертежи, графики, вычисления; правильно выполняет анализ ошибок.

Оценка «хорошо» выставляется, если: работа выполнена правильно с учетом 1-2 мелких погрешностей или 2-3 недочетов, исправленных самостоятельно по требованию преподавателя.

Оценка «удовлетворительно» выставляется, если: работа выполнена правильно не менее чем наполовину, допущены 1-2 погрешности или одна грубая ошибка.

Оценка «неудовлетворительно» выставляется, если: допущены две (и более) грубые ошибки в ходе работы, которые обучающийся не может исправить даже по требованию преподавателя или работа не выполнена полностью.

Тема 2. Стек. Очередь. Дек.

Определение стека, очереди и дека: основные характеристики и принципы работы.

Ключевые операции для каждой структуры данных.

Сравнение стеков, очередей и деков: когда использовать каждую из этих структур данных.

Реализация стека, очереди и дека на выбранном языке программирования

Примеры практического применения стека, очереди и дека в реальных задачах

Вопросы для самоконтроля:

1. Что такое стек и какие основные операции с ним выполняются?
2. В чем разница между очередью и стеком?
3. Каковы основные операции для дека и в каких случаях его использование предпочтительнее?
4. Приведите примеры задач, где использование стека является оптимальным решением.
5. Какова роль очереди в алгоритмах обработки данных?

Цель работы – формирование практических навыков работы со структурами данных стек, очередь и дек, а также их реализация на выбранном языке программирования.

В работе используется презентация слайдов, выполненная в MS Powerpoint.

Критерии оценивания выполнения практического задания:

Оценка «отлично» выставляется, если: работа выполнена в полном объеме с соблюдением необходимой последовательности действий; работа проведена в условиях, обеспечивающих получение правильных результатов и выводов; соблюдены правила техники безопасности; в ответе правильно и аккуратно выполняет все записи, таблицы, рисунки, чертежи, графики, вычисления; правильно выполняет анализ ошибок.

Оценка «хорошо» выставляется, если: работа выполнена правильно с учетом 1-2 мелких погрешностей или 2-3 недочетов, исправленных самостоятельно по требованию преподавателя.

Оценка «удовлетворительно» выставляется, если: работа выполнена правильно не менее чем наполовину, допущены 1-2 погрешности или одна грубая ошибка.

Оценка «неудовлетворительно» выставляется, если: допущены две (и более) грубые ошибки в ходе работы, которые обучающийся не может исправить даже по требованию преподавателя или работа не выполнена полностью.

Тема 3. Моделирование стека средствами языка C++.

Определение стека. Ключевые операции стека: Push, Pop, Peek, isEmpty, size

Реализация стека на C++:

Использовать динамический массив или связный список для хранения элементов стека.

Моделирование обратной польской нотации (RPN) для вычисления арифметических выражений.

Реализация алгоритма обхода графа в глубину (DFS).

Пример использования стека для проверки правильности скобочной последовательности.

Сравнение стеков с другими структурами данных:

Вопросы для самоконтроля:

1. Что такое стек и каковы его основные свойства?
2. Какова разница между операциями push и pop?
3. Как реализовать стек с использованием динамического массива в C++?
4. Приведите пример задачи, где использование стека будет оптимальным решением.

5. Как проверить, является ли последовательность скобок правильной с помощью стека?

Цель работы – развитие практических навыков программирования на C++, а также понимание принципов работы со стеком как структурой данных.

В работе используется база данных выполненная в MS Access/

Критерии оценивания выполнения практического задания:

Оценка «отлично» выставляется, если: работа выполнена в полном объеме с соблюдением необходимой последовательности действий; работа проведена в условиях, обеспечивающих получение правильных результатов и выводов; соблюдены правила техники безопасности; в ответе правильно и аккуратно выполняет все записи, таблицы, рисунки, чертежи, графики, вычисления; правильно выполняет анализ ошибок.

Оценка «хорошо» выставляется, если: работа выполнена правильно с учетом 1-2 мелких погрешностей или 2-3 недочетов, исправленных самостоятельно по требованию преподавателя.

Оценка «удовлетворительно» выставляется, если: работа выполнена правильно не менее чем наполовину, допущены 1-2 погрешности или одна грубая ошибка.

Оценка «неудовлетворительно» выставляется, если: допущены две (и более) грубые ошибки в ходе работы, которые обучающийся не может исправить даже по требованию преподавателя или работа не выполнена полностью.

Тема 4. Моделирование очереди средствами языка C++.

Определение очереди: описать основные характеристики: FIFO (First In, First Out) принцип работы.

Ключевые операции очереди: Enqueue, Dequeue, Front (или Peek), isEmpty, size:

Реализация очереди на C++

Использовать динамический массив или связный список для хранения элементов очереди.

Вопросы для самоконтроля:

1. Что такое очередь и каковы ее основные свойства?
2. Какова разница между операциями enqueue и dequeue?
3. Как реализовать очередь с использованием связного списка в C++?
4. Приведите пример задачи, где использование очереди будет оптимальным решением.
5. Как работает алгоритм обхода графа в ширину с использованием очереди?

Цель работы – развитие практических навыков программирования на C++, а также понимание принципов работы с очередью как структурой данных.

В работе используется база данных выполненная в MS Access/

Критерии оценивания выполнения практического задания:

Оценка «отлично» выставляется, если: работа выполнена в полном объеме с соблюдением необходимой последовательности действий; работа проведена в условиях, обеспечивающих получение правильных результатов и выводов; соблюдены правила техники безопасности; в ответе правильно и аккуратно выполняет все записи, таблицы, рисунки, чертежи, графики, вычисления; правильно выполняет анализ ошибок.

Оценка «хорошо» выставляется, если: работа выполнена правильно с учетом 1-2 мелких погрешностей или 2-3 недочетов, исправленных самостоятельно по требованию преподавателя.

Оценка «удовлетворительно» выставляется, если: работа выполнена правильно не менее чем наполовину, допущены 1-2 погрешности или одна грубая ошибка.

Оценка «неудовлетворительно» выставляется, если: допущены две (и более) грубые ошибки в ходе работы, которые обучающийся не может исправить даже по требованию преподавателя или работа не выполнена полностью.

Тема 5. Кольцевой буфер

Принципы работы защищенных беспроводных сетей**.** Стандарты безопасности для беспроводных сетей (WPA2, WPA3)**.** Методы шифрования и аутентификации в беспроводных сетях**.** Защита от атак на беспроводные сети (например, MITM, DoS)**.** Рекомендации по проектированию и внедрению защищенных беспроводных систем

Вопросы для самоконтроля:

1. Какие основные принципы работы защищенных беспроводных сетей?
2. В чем разница между стандартами WPA2 и WPA3?
3. Какие методы шифрования используются в беспроводных сетях?
4. Что такое аутентификация в контексте беспроводных сетей и какие протоколы для этого применяются?
5. Какие виды атак наиболее распространены в беспроводных сетях и как им противостоять?
6. Каковы рекомендации по проектированию защищенных беспроводных систем на промышленных предприятиях?
7. Что такое VLAN и как он может помочь в защите беспроводных сетей?

Цель работы - получение теоретических и практических навыков настройки защищенных беспроводных сетей, анализа их безопасности и разработки рекомендаций по их улучшению.

В работе используется презентация слайдов, выполненная в MS Powerpoint

Критерии оценивания выполнения практического задания:

Оценка «отлично» выставляется, если: работа выполнена в полном объеме с соблюдением необходимой последовательности действий; работа проведена в условиях, обеспечивающих получение правильных результатов и выводов; соблюдены правила техники безопасности; в ответе правильно и аккуратно выполняет все записи, таблицы, рисунки, чертежи, графики, вычисления; правильно выполняет анализ ошибок.

Оценка «хорошо» выставляется, если: работа выполнена правильно с учетом 1-2 мелких погрешностей или 2-3 недочетов, исправленных самостоятельно по требованию преподавателя.

Оценка «удовлетворительно» выставляется, если: работа выполнена правильно не менее чем наполовину, допущены 1-2 погрешности или одна грубая ошибка.

Оценка «неудовлетворительно» выставляется, если: допущены две (и более) грубые ошибки в ходе работы, которые обучающийся не может исправить даже по требованию преподавателя или работа не выполнена полностью.

Тема 6. Реализация двух однотипных стеков

Описать основные характеристики: LIFO (Last In, First Out) принцип работы.

Ключевые операции стека: Push, Pop, Top (или Peek),size

Реализация двух однотипных стеков на C++:

Обеспечить возможность независимого управления обоими стеками.

Реализация алгоритма обратной польской нотации (RPN) с использованием двух стеков.

Моделирование системы обработки выражений, где один стек используется для операндов, а другой – для операторов.

Вопросы для самоконтроля:

1. Что такое стек и каковы его основные свойства?
2. Какова разница между операциями push и pop?
3. Как реализовать два стека в одном классе на C++?
4. Приведите пример задачи, где использование двух стеков будет оптимальным решением.
5. Как работает алгоритм обратной польской нотации с использованием двух стеков?

Цель работы – развитие практических навыков программирования на C++, а также понимание принципов работы со стеком как структурой данных.

В работе используется презентация слайдов, выполненная в MS Powerpoint

Критерии оценивания выполнения практического задания:

Оценка «отлично» выставляется, если: работа выполнена в полном объеме с соблюдением необходимой последовательности действий; работа

проведена в условиях, обеспечивающих получение правильных результатов и выводов; соблюдены правила техники безопасности; в ответе правильно и аккуратно выполняет все записи, таблицы, рисунки, чертежи, графики, вычисления; правильно выполняет анализ ошибок.

Оценка «хорошо» выставляется, если: работа выполнена правильно с учетом 1-2 мелких погрешностей или 2-3 недочетов, исправленных самостоятельно по требованию преподавателя.

Оценка «удовлетворительно» выставляется, если: работа выполнена правильно не менее чем наполовину, допущены 1-2 погрешности или одна грубая ошибка.

Оценка «неудовлетворительно» выставляется, если: допущены две (и более) грубые ошибки в ходе работы, которые обучающийся не может исправить даже по требованию преподавателя или работа не выполнена полностью.

Тема 7. Однонаправленные списки. Операция вставки и удаления элемента

Описать основные характеристики: последовательность элементов, где каждый элемент (узел) содержит данные и указатель на следующий узел.

Ключевые операции однонаправленного списка

Реализовать методы для вставки и удаления элементов, а также метод для вывода списка на экран.

Реализация простого текстового редактора, где операции вставки и удаления текста будут реализованы с использованием однонаправленного списка.

Моделирование очереди задач, где каждая задача представляется узлом списка.

Вопросы для самоконтроля:

1. Что такое однонаправленный список и каковы его основные свойства?
2. Каковы основные операции вставки и удаления в однонаправленном списке?
3. Как реализовать однонаправленный список на C++?
4. Приведите пример задачи, где использование однонаправленного списка будет оптимальным решением.
5. Каковы преимущества и недостатки однонаправленного списка по сравнению с массивами?

Цель работы – развитие практических навыков программирования на C++, а также понимание принципов работы с однонаправленным списком как структурой данных.

В работе используется презентация слайдов, выполненная в MS Powerpoint

Критерии оценивания выполнения практического задания:

Оценка «отлично» выставляется, если: работа выполнена в полном объеме с соблюдением необходимой последовательности действий; работа проведена в условиях, обеспечивающих получение правильных результатов и выводов; соблюдены правила техники безопасности; в ответе правильно и аккуратно выполняет все записи, таблицы, рисунки, чертежи, графики, вычисления; правильно выполняет анализ ошибок.

Оценка «хорошо» выставляется, если: работа выполнена правильно с учетом 1-2 мелких погрешностей или 2-3 недочетов, исправленных самостоятельно по требованию преподавателя.

Оценка «удовлетворительно» выставляется, если: работа выполнена правильно не менее чем наполовину, допущены 1-2 погрешности или одна грубая ошибка.

Оценка «неудовлетворительно» выставляется, если: допущены две (и более) грубые ошибки в ходе работы, которые обучающийся не может исправить даже по требованию преподавателя или работа не выполнена полностью.

Тема 8. Двунаправленные списки. Построение, операция удаления и вставки элемента

Описать основные характеристики: последовательность элементов, где каждый элемент (узел) содержит данные, указатель на следующий узел и указатель на предыдущий узел.

Реализовать методы для вставки и удаления элементов, а также метод для вывода списка на экран (в прямом и обратном порядке).

Реализация программы для управления историей браузера, где каждая страница представляется узлом списка и можно перемещаться вперед и назад.

Моделирование системы управления задачами, где пользователи могут добавлять, удалять и изменять приоритет задач.

Вопросы для самоконтроля:

1. Что такое двунаправленный список и каковы его основные свойства?
2. Каковы основные операции вставки и удаления в двунаправленном списке?
3. Как реализовать двунаправленный список на C++?
4. Приведите пример задачи, где использование двунаправленного списка будет оптимальным решением.
5. Каковы преимущества и недостатки двунаправленного списка по сравнению с однонаправленными списками?

Цель работы – развитие практических навыков программирования на C++, а также понимание принципов работы с двунаправленным списком как структурой данных.

В работе используется презентация слайдов, выполненная в MS Powerpoint

Критерии оценивания выполнения практического задания:

Оценка «отлично» выставляется, если: работа выполнена в полном объеме с соблюдением необходимой последовательности действий; работа проведена в условиях, обеспечивающих получение правильных результатов и выводов; соблюдены правила техники безопасности; в ответе правильно и аккуратно выполняет все записи, таблицы, рисунки, чертежи, графики, вычисления; правильно выполняет анализ ошибок.

Оценка «хорошо» выставляется, если: работа выполнена правильно с учетом 1-2 мелких погрешностей или 2-3 недочетов, исправленных самостоятельно по требованию преподавателя.

Оценка «удовлетворительно» выставляется, если: работа выполнена правильно не менее чем наполовину, допущены 1-2 погрешности или одна грубая ошибка.

Оценка «неудовлетворительно» выставляется, если: допущены две (и более) грубые ошибки в ходе работы, которые обучающийся не может исправить даже по требованию преподавателя или работа не выполнена полностью.

Тема 9. Деревья. Дерево двоичного поиска. Обходы ДДП

Описать основные компоненты дерева: узлы, корень, листья, уровень узла и высота дерева.

Основные операции с деревом двоичного поиска: вставка элемента, поиск элемента, удаление элемента, обсуждение сложности выполнения этих операций.

Реализация программы для управления коллекцией книг, где каждая книга представляется узлом дерева.

Моделирование системы для хранения и быстрого поиска информации о пользователях.

Вопросы для самоконтроля:

1. Что такое дерево и какие его основные компоненты?
2. Каковы свойства дерева двоичного поиска?
3. Какие операции можно выполнять с деревом двоичного поиска?
4. В чем различия между методами обхода дерева?
5. Каковы преимущества и недостатки дерева двоичного поиска по сравнению с другими структурами данных?

Цель работы – формирование теоретических и практических навыков работы с деревьями, особенно с деревом двоичного поиска.

В работе используется презентация слайдов, выполненная в MS Powerpoint

Критерии оценивания выполнения практического задания:

Оценка «отлично» выставляется, если: работа выполнена в полном объеме с соблюдением необходимой последовательности действий; работа

проведена в условиях, обеспечивающих получение правильных результатов и выводов; соблюдены правила техники безопасности; в ответе правильно и аккуратно выполняет все записи, таблицы, рисунки, чертежи, графики, вычисления; правильно выполняет анализ ошибок.

Оценка «хорошо» выставляется, если: работа выполнена правильно с учетом 1-2 мелких погрешностей или 2-3 недочетов, исправленных самостоятельно по требованию преподавателя.

Оценка «удовлетворительно» выставляется, если: работа выполнена правильно не менее чем наполовину, допущены 1-2 погрешности или одна грубая ошибка.

Оценка «неудовлетворительно» выставляется, если: допущены две (и более) грубые ошибки в ходе работы, которые обучающийся не может исправить даже по требованию преподавателя или работа не выполнена полностью.

Тема 10. Обработка дерева двоичного поиска.

Определить, что такое обработка дерева и зачем она нужна.

Реализация программы для управления базой данных клиентов, где каждый клиент представлен узлом дерева.

Моделирование системы для хранения и быстрого поиска информации о товарах в интернет-магазине.

Сравнить производительность ДДП с другими структурами данных, такими как хеш-таблицы и AVL-деревья.

Вопросы для самоконтроля:

1. Что такое обработка дерева двоичного поиска и почему она важна?
2. Какие основные методы обработки ДДП существуют?
3. Как реализуются операции вставки, поиска и удаления в ДДП?
4. В чем различия между методами обхода дерева?
5. Какие дополнительные операции можно выполнять с деревом двоичного поиска?

Цель работы – углубление знаний и практических навыков работы с деревьями двоичного поиска, а также развитие умений реализовывать различные операции обработки.

В работе используется презентация слайдов, выполненная в MS Powerpoint

Критерии оценивания выполнения практического задания:

Оценка «отлично» выставляется, если: работа выполнена в полном объеме с соблюдением необходимой последовательности действий; работа проведена в условиях, обеспечивающих получение правильных результатов и выводов; соблюдены правила техники безопасности; в ответе правильно и аккуратно выполняет все записи, таблицы, рисунки, чертежи, графики, вычисления; правильно выполняет анализ ошибок.

Оценка «хорошо» выставляется, если: работа выполнена правильно с учетом 1-2 мелких погрешностей или 2-3 недочетов, исправленных самостоятельно по требованию преподавателя.

Оценка «удовлетворительно» выставляется, если: работа выполнена правильно не менее чем наполовину, допущены 1-2 погрешности или одна грубая ошибка.

Оценка «неудовлетворительно» выставляется, если: допущены две (и более) грубые ошибки в ходе работы, которые обучающийся не может исправить даже по требованию преподавателя или работа не выполнена полностью.

Тема 11. Система управления вводом-выводом. Буферизация ввода-вывода

Определить, что такое система управления вводом-выводом и ее роль в операционных системах.

Основные компоненты системы ввода-вывода: описать различные методы буферизации: буферизация на уровне устройства, буферизация на уровне приложения.

Реализация программы для чтения и записи данных в файл с использованием буферизации.

Моделирование системы для обработки данных с различных устройств ввода (например, считывание данных с датчиков).

Сравнить производительность различных подходов к буферизации (например, с использованием памяти и диска).

Вопросы для самоконтроля:

1. Что такое система управления вводом-выводом и какова ее роль в операционной системе?
2. Какие основные компоненты входят в систему ввода-вывода?
3. Каковы преимущества и недостатки буферизации ввода-вывода?
4. Какие алгоритмы управления буферизацией существуют и как они работают?
5. Как можно реализовать систему ввода-вывода в программном обеспечении?

Цель работы – углубление знаний и практических навыков работы с системами управления вводом-выводом, а также развитие умений реализовывать различные методы буферизации.

В работе используется презентация слайдов, выполненная в MS Powerpoint

Критерии оценивания выполнения практического задания:

Оценка «отлично» выставляется, если: работа выполнена в полном объеме с соблюдением необходимой последовательности действий; работа проведена в условиях, обеспечивающих получение правильных результатов и

выводов; соблюдены правила техники безопасности; в ответе правильно и аккуратно выполняет все записи, таблицы, рисунки, чертежи, графики, вычисления; правильно выполняет анализ ошибок.

Оценка «хорошо» выставляется, если: работа выполнена правильно с учетом 1-2 мелких погрешностей или 2-3 недочетов, исправленных самостоятельно по требованию преподавателя.

Оценка «удовлетворительно» выставляется, если: работа выполнена правильно не менее чем наполовину, допущены 1-2 погрешности или одна грубая ошибка.

Оценка «неудовлетворительно» выставляется, если: допущены две (и более) грубые ошибки в ходе работы, которые обучающийся не может исправить даже по требованию преподавателя или работа не выполнена полностью.

Тема 12. Общие операции над файлами

Определить, что такое файл и какова его роль в операционных системах. Обсудить права доступа к файлам и их значение.

Рассмотреть возможные ошибки при выполнении операций с файлами (например, файл не найден, недостаточно прав и т.д.).

Реализовать программу для создания, записи и чтения данных из текстового файла.

Сравнить производительность различных методов чтения и записи данных.

Вопросы для самоконтроля:

1. Что такое файл и какова его роль в операционной системе?
2. Какие основные операции можно выполнять с файлами?
3. Какова структура файловой системы и какие существуют ее типы?
4. Какие возможные ошибки могут возникнуть при работе с файлами и как их обрабатывать?
5. Как реализовать дополнительные операции над файлами, такие как удаление или переименование?

Цель работы – углубление знаний и практических навыков работы с файлами в операционных системах, а также развитие умений реализовывать различные операции над файлами.

В работе используется презентация слайдов, выполненная в MS Powerpoint

Критерии оценивания выполнения практического задания:

Оценка «отлично» выставляется, если: работа выполнена в полном объеме с соблюдением необходимой последовательности действий; работа проведена в условиях, обеспечивающих получение правильных результатов и выводов; соблюдены правила техники безопасности; в ответе правильно и

аккуратно выполняет все записи, таблицы, рисунки, чертежи, графики, вычисления; правильно выполняет анализ ошибок.

Оценка «хорошо» выставляется, если: работа выполнена правильно с учетом 1-2 мелких погрешностей или 2-3 недочетов, исправленных самостоятельно по требованию преподавателя.

Оценка «удовлетворительно» выставляется, если: работа выполнена правильно не менее чем наполовину, допущены 1-2 погрешности или одна грубая ошибка.

Оценка «неудовлетворительно» выставляется, если: допущены две (и более) грубые ошибки в ходе работы, которые обучающийся не может исправить даже по требованию преподавателя или работа не выполнена полностью.

Тема 13. Ориентированные графы. Представления ориентированных графов.

Определение ориентированного графа и его ключевые элементы. Различия между направленными и ненаправленными графами.

Представление ориентированных графов

Примеры применения ориентированных графов в реальных задачах

Алгоритмы, работающие с ориентированными графами

Вопросы для самоконтроля:

1. Что такое ориентированный граф и каковы его основные компоненты?
2. В чем разница между матрицей смежности и списком смежности?
3. Приведите пример задачи, которая может быть решена с помощью ориентированного графа.
4. Какие алгоритмы можно использовать для работы с ориентированными графами?
5. Как определить, является ли ориентированный граф сильно связным?

Цель работы - формирование теоретических и практических навыков в области работы с ориентированными графами, их представлениями и алгоритмами.

В работе используется презентация слайдов, выполненная в MS Powerpoint

Критерии оценивания выполнения практического задания:

Оценка «отлично» выставляется, если: работа выполнена в полном объеме с соблюдением необходимой последовательности действий; работа проведена в условиях, обеспечивающих получение правильных результатов и выводов; соблюдены правила техники безопасности; в ответе правильно и аккуратно выполняет все записи, таблицы, рисунки, чертежи, графики, вычисления; правильно выполняет анализ ошибок.

Оценка «хорошо» выставляется, если: работа выполнена правильно с учетом 1-2 мелких погрешностей или 2-3 недочетов, исправленных самостоятельно по требованию преподавателя.

Оценка «удовлетворительно» выставляется, если: работа выполнена правильно не менее чем наполовину, допущены 1-2 погрешности или одна грубая ошибка.

Оценка «неудовлетворительно» выставляется, если: допущены две (и более) грубые ошибки в ходе работы, которые обучающийся не может исправить даже по требованию преподавателя или работа не выполнена полностью.

Тема 14. Задача нахождения кратчайшего пути.

Определение задачи нахождения кратчайшего пути и ее важность в теории графов.

Основные алгоритмы для решения задачи кратчайшего пути.

Различия между взвешенными и невзвешенными графами.

Применение задачи нахождения кратчайшего пути в реальных сценариях.

Примеры реализации алгоритмов на языке программирования.

Вопросы для самоконтроля:

1. Что такое задача нахождения кратчайшего пути и в чем ее практическая значимость?

2. Какие основные алгоритмы используются для нахождения кратчайшего пути? В чем их отличия?

3. Каковы особенности работы с взвешенными и невзвешенными графами?

4. Приведите примеры реальных приложений, где используется задача нахождения кратчайшего пути.

5. Как можно реализовать алгоритм Дейкстры на языке Python?

Цель работы - формирование теоретических знаний и практических навыков в области решения задачи нахождения кратчайшего пути, понимания алгоритмов и их применения.

В работе используется презентация слайдов, выполненная в MS Powerpoint

Критерии оценивания выполнения практического задания:

Оценка «отлично» выставляется, если: работа выполнена в полном объеме с соблюдением необходимой последовательности действий; работа проведена в условиях, обеспечивающих получение правильных результатов и выводов; соблюдены правила техники безопасности; в ответе правильно и аккуратно выполняет все записи, таблицы, рисунки, чертежи, графики, вычисления; правильно выполняет анализ ошибок.

Оценка «хорошо» выставляется, если: работа выполнена правильно с учетом 1-2 мелких погрешностей или 2-3 недочетов, исправленных самостоятельно по требованию преподавателя.

Оценка «удовлетворительно» выставляется, если: работа выполнена правильно не менее чем наполовину, допущены 1-2 погрешности или одна грубая ошибка.

Оценка «неудовлетворительно» выставляется, если: допущены две (и более) грубые ошибки в ходе работы, которые обучающийся не может исправить даже по требованию преподавателя или работа не выполнена полностью.

Тема 15. Нахождение кратчайших путей между парами вершин. Алгоритм Флойда.

Определение задачи нахождения кратчайших путей между парами вершин и ее значимость в теории графов.

Описание алгоритма Флойда-Уоршелла, его принципы работы и шаги выполнения.

Применение алгоритма Флойда в различных областях.

Сравнение алгоритма Флойда с другими алгоритмами нахождения кратчайших путей

Примеры реализации алгоритма Флойда на языке программирования.

Вопросы для самоконтроля:

1. Что такое задача нахождения кратчайших путей между парами вершин и почему она важна?

2. Как работает алгоритм Флойда-Уоршелла? Опишите основные шаги его выполнения.

3. В каких случаях предпочтительнее использовать алгоритм Флойда по сравнению с другими методами нахождения кратчайших путей?

4. Приведите примеры реальных приложений, где используется алгоритм Флойда.

5. Как можно реализовать алгоритм Флойда на языке Python?

Цель работы - формирование теоретических знаний и практических навыков в области нахождения кратчайших путей между парами вершин, понимания алгоритма Флойда и его применения.

В работе используется презентация слайдов, выполненная в MS Powerpoint

Критерии оценивания выполнения практического задания:

Оценка «отлично» выставляется, если: работа выполнена в полном объеме с соблюдением необходимой последовательности действий; работа проведена в условиях, обеспечивающих получение правильных результатов и выводов; соблюдены правила техники безопасности; в ответе правильно и

аккуратно выполняет все записи, таблицы, рисунки, чертежи, графики, вычисления; правильно выполняет анализ ошибок.

Оценка «хорошо» выставляется, если: работа выполнена правильно с учетом 1-2 мелких погрешностей или 2-3 недочетов, исправленных самостоятельно по требованию преподавателя.

Оценка «удовлетворительно» выставляется, если: работа выполнена правильно не менее чем наполовину, допущены 1-2 погрешности или одна грубая ошибка.

Оценка «неудовлетворительно» выставляется, если: допущены две (и более) грубые ошибки в ходе работы, которые обучающийся не может исправить даже по требованию преподавателя или работа не выполнена полностью.

Тема 16. Поиск центра ориентированного графа.

Определение центра ориентированного графа и его ключевые характеристики.

Описание методов поиска центра ориентированного графа, включая алгоритмы и их принципы работы.

3. Применение центра ориентированных графов в различных областях.

4. Сравнение центра ориентированных графов с другими типами графов.

5. Примеры реализации алгоритмов поиска центра ориентированного графа на языке программирования.

Вопросы для самоконтроля:

1. Что такое центра ориентированный граф и почему он важен в теории графов?

2. Какие методы используются для поиска центра ориентированного графа? Опишите основные шаги выполнения этих методов.

3. В каких случаях предпочтительнее использовать центра ориентированные графы по сравнению с другими типами графов?

4. Приведите примеры реальных приложений, где используются центра ориентированные графы.

5. Как можно реализовать алгоритм поиска центра ориентированного графа на языке Python?

Цель работы - формирование теоретических знаний и практических навыков в области анализа центра ориентированных графов, понимания методов их поиска и применения.

В работе используется презентация слайдов, выполненная в MS Powerpoint

Критерии оценивания выполнения практического задания:

Оценка «отлично» выставляется, если: работа выполнена в полном объеме с соблюдением необходимой последовательности действий; работа проведена в условиях, обеспечивающих получение правильных результатов и выводов; соблюдены правила техники безопасности; в ответе правильно и

аккуратно выполняет все записи, таблицы, рисунки, чертежи, графики, вычисления; правильно выполняет анализ ошибок.

Оценка «хорошо» выставляется, если: работа выполнена правильно с учетом 1-2 мелких погрешностей или 2-3 недочетов, исправленных самостоятельно по требованию преподавателя.

Оценка «удовлетворительно» выставляется, если: работа выполнена правильно не менее чем наполовину, допущены 1-2 погрешности или одна грубая ошибка.

Оценка «неудовлетворительно» выставляется, если: допущены две (и более) грубые ошибки в ходе работы, которые обучающийся не может исправить даже по требованию преподавателя или работа не выполнена полностью.

Тема 17. Обход ориентированных графов. Поиск в глубину.

Определение ориентированного графа и его основные характеристики. Описание алгоритма поиска в глубину (DFS) и его принцип работы.

Применение поиска в глубину в различных задачах.

Сравнение поиска в глубину с другими методами обхода графов.

Примеры реализации алгоритма поиска в глубину на языке программирования.

Вопросы для самоконтроля:

1. Что такое ориентированный граф и чем он отличается от неориентированного?
2. Как работает алгоритм поиска в глубину? Опишите основные шаги его выполнения.
3. В каких случаях предпочтительнее использовать поиск в глубину по сравнению с другими методами обхода графов?
4. Приведите примеры реальных задач, где применяется поиск в глубину.
5. Как можно реализовать алгоритм поиска в глубину на языке Python?

Цель работы - формирование теоретических знаний и практических навыков в области обхода ориентированных графов, понимания алгоритма поиска в глубину и его применения.

В работе используется презентация слайдов, выполненная в MS Powerpoint

Критерии оценивания выполнения практического задания:

Оценка «отлично» выставляется, если: работа выполнена в полном объеме с соблюдением необходимой последовательности действий; работа проведена в условиях, обеспечивающих получение правильных результатов и выводов; соблюдены правила техники безопасности; в ответе правильно и аккуратно выполняет все записи, таблицы, рисунки, чертежи, графики, вычисления; правильно выполняет анализ ошибок.

Оценка «хорошо» выставляется, если: работа выполнена правильно с учетом 1-2 мелких погрешностей или 2-3 недочетов, исправленных самостоятельно по требованию преподавателя.

Оценка «удовлетворительно» выставляется, если: работа выполнена правильно не менее чем наполовину, допущены 1-2 погрешности или одна грубая ошибка.

Оценка «неудовлетворительно» выставляется, если: допущены две (и более) грубые ошибки в ходе работы, которые обучающийся не может исправить даже по требованию преподавателя или работа не выполнена полностью.

Тема 18. Алгоритм нахождения сильно связных компонент.

Определение сильно связных компонент (SCC) в ориентированном графе и их основные характеристики.

Описание алгоритма нахождения сильно связных компонент.

3. Применение алгоритмов нахождения SCC в различных задачах.

4. Сравнение различных алгоритмов нахождения сильно связных компонент.

5. Примеры реализации алгоритма нахождения SCC на языке программирования.

Вопросы для самоконтроля:

1. Что такое сильно связная компонента и как она определяется в ориентированном графе?

2. Как работают алгоритмы Косарайю и Тарьяна? Опишите основные шаги их выполнения.

3. В каких случаях предпочтительнее использовать один алгоритм нахождения SCC по сравнению с другим?

4. Приведите примеры реальных задач, где применяется нахождение сильно связных компонент.

5. Как можно реализовать алгоритм нахождения сильно связных компонент на языке Python?

Цель работы - формирование теоретических знаний и практических навыков в области нахождения сильно связных компонент в ориентированных графах, понимания алгоритмов их нахождения и их применения.

В работе используется презентация слайдов, выполненная в MS Powerpoint

Критерии оценивания выполнения практического задания:

Оценка «отлично» выставляется, если: работа выполнена в полном объеме с соблюдением необходимой последовательности действий; работа проведена в условиях, обеспечивающих получение правильных результатов и выводов; соблюдены правила техники безопасности; в ответе правильно и

аккуратно выполняет все записи, таблицы, рисунки, чертежи, графики, вычисления; правильно выполняет анализ ошибок.

Оценка «хорошо» выставляется, если: работа выполнена правильно с учетом 1-2 мелких погрешностей или 2-3 недочетов, исправленных самостоятельно по требованию преподавателя.

Оценка «удовлетворительно» выставляется, если: работа выполнена правильно не менее чем наполовину, допущены 1-2 погрешности или одна грубая ошибка.

Оценка «неудовлетворительно» выставляется, если: допущены две (и более) грубые ошибки в ходе работы, которые обучающийся не может исправить даже по требованию преподавателя или работа не выполнена полностью.

Тема 19. Представление неориентированных графов. Основные деревья минимальной стоимости.

Определение неориентированного графа и его основные характеристики.

Способы представления неориентированных графов. Понятие основного дерева и его свойства.

Алгоритмы нахождения основного дерева минимальной стоимости.

Применение основных деревьев в различных задачах (например, проектирование сетей, оптимизация маршрутов).

Сравнение различных алгоритмов нахождения основного дерева минимальной стоимости.

Примеры реализации алгоритмов нахождения основного дерева на языке программирования.

Вопросы для самоконтроля:

1. Что такое неориентированный граф и каковы его основные характеристики?
2. Каковы преимущества и недостатки различных способов представления неориентированных графов?
3. Что такое основное дерево и каковы его основные свойства?
4. Как работают алгоритмы Краскала и Прима? Опишите основные шаги их выполнения.
5. В каких случаях предпочтительнее использовать один алгоритм нахождения основного дерева по сравнению с другим?

Цель работы - формирование теоретических знаний и практических навыков в области представления неориентированных графов, понимания концепции основных деревьев минимальной стоимости и алгоритмов их нахождения.

В работе используется презентация слайдов, выполненная в MS Powerpoint

Критерии оценивания выполнения практического задания:

Оценка «отлично» выставляется, если: работа выполнена в полном объеме с соблюдением необходимой последовательности действий; работа проведена в условиях, обеспечивающих получение правильных результатов и выводов; соблюдены правила техники безопасности; в ответе правильно и аккуратно выполняет все записи, таблицы, рисунки, чертежи, графики, вычисления; правильно выполняет анализ ошибок.

Оценка «хорошо» выставляется, если: работа выполнена правильно с учетом 1-2 мелких погрешностей или 2-3 недочетов, исправленных самостоятельно по требованию преподавателя.

Оценка «удовлетворительно» выставляется, если: работа выполнена правильно не менее чем наполовину, допущены 1-2 погрешности или одна грубая ошибка.

Оценка «неудовлетворительно» выставляется, если: допущены две (и более) грубые ошибки в ходе работы, которые обучающийся не может исправить даже по требованию преподавателя или работа не выполнена полностью.

Тема 20. Алгоритм Прима. Алгоритм Крускала.

Определение основного дерева и его важность в теории графов.

Подробное описание алгоритма Прима. Подробное описание алгоритма Краскала.

Сравнение алгоритмов Прима и Краскала.

Применение алгоритмов Прима и Краскала в реальных задачах.

Примеры реализации обоих алгоритмов на языке программирования.

Вопросы для самоконтроля:

1. Что такое основное дерево и почему оно важно в теории графов?
2. Каковы основные отличия между алгоритмами Прима и Краскала?
3. Опишите основные шаги работы алгоритма Прима.
4. Опишите основные шаги работы алгоритма Краскала.
5. В каких случаях один из алгоритмов может работать быстрее или эффективнее другого?

Цель работы - формирование теоретических знаний и практических навыков в области нахождения основных деревьев с использованием алгоритмов Прима и Краскала.

В работе используется презентация слайдов, выполненная в MS Powerpoint

Критерии оценивания выполнения практического задания:

Оценка «отлично» выставляется, если: работа выполнена в полном объеме с соблюдением необходимой последовательности действий; работа проведена в условиях, обеспечивающих получение правильных результатов и выводов; соблюдены правила техники безопасности; в ответе правильно и

аккуратно выполняет все записи, таблицы, рисунки, чертежи, графики, вычисления; правильно выполняет анализ ошибок.

Оценка «хорошо» выставляется, если: работа выполнена правильно с учетом 1-2 мелких погрешностей или 2-3 недочетов, исправленных самостоятельно по требованию преподавателя.

Оценка «удовлетворительно» выставляется, если: работа выполнена правильно не менее чем наполовину, допущены 1-2 погрешности или одна грубая ошибка.

Оценка «неудовлетворительно» выставляется, если: допущены две (и более) грубые ошибки в ходе работы, которые обучающийся не может исправить даже по требованию преподавателя или работа не выполнена полностью.

Тема 21. Контейнерный класс vector

Определение контейнерного класса vector и его роль в стандартной библиотеке C++.

Основные характеристики и преимущества использования vector. Описание структуры контейнера vectorю

Сравнение vector с другими контейнерами STL

Применение контейнера vector в реальных задачах

Примеры реализации операций с vector на языке C++.

Вопросы для самоконтроля:

1. Что такое контейнерный класс vector и какие его ключевые особенности?

2. Как происходит управление памятью в контейнере vector?

3. Какие основные методы предоставляет класс vector для работы с элементами?

4. В чем разница между vector и другими контейнерами STL, такими как list и deque?

5. Приведите примеры задач, где использование vector является предпочтительным.

Цель работы - формирование теоретических знаний и практических навыков работы с контейнерным классом vector в C++.

В работе используется презентация слайдов, выполненная в MS Powerpoint

Критерии оценивания выполнения практического задания:

Оценка «отлично» выставляется, если: работа выполнена в полном объеме с соблюдением необходимой последовательности действий; работа проведена в условиях, обеспечивающих получение правильных результатов и выводов; соблюдены правила техники безопасности; в ответе правильно и аккуратно выполняет все записи, таблицы, рисунки, чертежи, графики, вычисления; правильно выполняет анализ ошибок.

Оценка «хорошо» выставляется, если: работа выполнена правильно с учетом 1-2 мелких погрешностей или 2-3 недочетов, исправленных самостоятельно по требованию преподавателя.

Оценка «удовлетворительно» выставляется, если: работа выполнена правильно не менее чем наполовину, допущены 1-2 погрешности или одна грубая ошибка.

Оценка «неудовлетворительно» выставляется, если: допущены две (и более) грубые ошибки в ходе работы, которые обучающийся не может исправить даже по требованию преподавателя или работа не выполнена полностью.

Тема 22. Контейнерный класс list

Определение контейнерного класса list и его роль в стандартной библиотеке C++. Основные характеристики и преимущества использования list.

Описание структуры контейнера list. Основные операции с list.

Сравнение list с другими контейнерами STL (например, vector, deque).

Применение контейнера list в реальных задачах.

Примеры реализации операций с list на языке C++.

Вопросы для самоконтроля:

1. Что такое контейнерный класс list и какие его ключевые особенности?
2. Какова структура данных, лежащая в основе контейнера list?
3. Какие основные методы предоставляет класс list для работы с элементами?
4. В чем разница между list и другими контейнерами STL, такими как vector и deque?
5. Приведите примеры задач, где использование list является предпочтительным.

Цель работы - формирование теоретических знаний и практических навыков работы с контейнерным классом list в C++.

В работе используется презентация слайдов, выполненная в MS Powerpoint

Критерии оценивания выполнения практического задания:

Оценка «отлично» выставляется, если: работа выполнена в полном объеме с соблюдением необходимой последовательности действий; работа проведена в условиях, обеспечивающих получение правильных результатов и выводов; соблюдены правила техники безопасности; в ответе правильно и аккуратно выполняет все записи, таблицы, рисунки, чертежи, графики, вычисления; правильно выполняет анализ ошибок.

Оценка «хорошо» выставляется, если: работа выполнена правильно с учетом 1-2 мелких погрешностей или 2-3 недочетов, исправленных самостоятельно по требованию преподавателя.

Оценка «удовлетворительно» выставляется, если: работа выполнена правильно не менее чем наполовину, допущены 1-2 погрешности или одна грубая ошибка.

Оценка «неудовлетворительно» выставляется, если: допущены две (и более) грубые ошибки в ходе работы, которые обучающийся не может исправить даже по требованию преподавателя или работа не выполнена полностью.

Тема 23. Ассоциативные контейнеры set и map.

Определение ассоциативных контейнеров set и map и их роль в стандартной библиотеке C++.

Основные характеристики и преимущества использования set и map.

Основные операции с set и map. Примеры использования set и map

Сравнение set и map с другими контейнерами STL

Применение ассоциативных контейнеров в реальных задачах.

Примеры реализации операций с set и map на языке C++.

Вопросы для самоконтроля:

1. Что такое ассоциативные контейнеры set и map, и какие их ключевые особенности?

2. Какова структура данных, лежащая в основе контейнера set?

3. Какова структура данных, лежащая в основе контейнера map?

4. Какие основные методы предоставляет класс set для работы с элементами?

5. Какие основные методы предоставляет класс map для работы с парами ключ-значение?

Цель работы - формирование теоретических знаний и практических навыков работы с ассоциативными контейнерами set и map в C++.

В работе используется презентация слайдов, выполненная в MS Powerpoint

Критерии оценивания выполнения практического задания:

Оценка «отлично» выставляется, если: работа выполнена в полном объеме с соблюдением необходимой последовательности действий; работа проведена в условиях, обеспечивающих получение правильных результатов и выводов; соблюдены правила техники безопасности; в ответе правильно и аккуратно выполняет все записи, таблицы, рисунки, чертежи, графики, вычисления; правильно выполняет анализ ошибок.

Оценка «хорошо» выставляется, если: работа выполнена правильно с учетом 1-2 мелких погрешностей или 2-3 недочетов, исправленных самостоятельно по требованию преподавателя.

Оценка «удовлетворительно» выставляется, если: работа выполнена правильно не менее чем наполовину, допущены 1-2 погрешности или одна грубая ошибка.

Оценка «неудовлетворительно» выставляется, если: допущены две (и более) грубые ошибки в ходе работы, которые обучающийся не может исправить даже по требованию преподавателя или работа не выполнена полностью.

Тема 24. Общая схема компиляции.

Определение компиляции и её роль в процессе разработки программного обеспечения.

Основные этапы компиляции. Подробное описание каждого этапа. Описание промежуточных представлений.

Роль компилятора и его компоненты. Примеры компиляции на языке C++.

Сравнение различных компиляторов и их особенностей.

Применение знаний о компиляции в реальных задачах.

Вопросы для самоконтроля:

1. Что такое компиляция и какую роль она играет в разработке программного обеспечения?
2. Какие основные этапы включает в себя процесс компиляции?
3. Каковы задачи препроцессора и какие директивы он обрабатывает?
4. Как происходит преобразование исходного кода в объектный код?
5. Что такое линковка и какие задачи она решает?

Цель работы - формирование теоретических знаний и практических навыков понимания процесса компиляции программ на языке C++.

В работе используется презентация слайдов, выполненная в MS Powerpoint

Критерии оценивания выполнения практического задания:

Оценка «отлично» выставляется, если: работа выполнена в полном объеме с соблюдением необходимой последовательности действий; работа проведена в условиях, обеспечивающих получение правильных результатов и выводов; соблюдены правила техники безопасности; в ответе правильно и аккуратно выполняет все записи, таблицы, рисунки, чертежи, графики, вычисления; правильно выполняет анализ ошибок.

Оценка «хорошо» выставляется, если: работа выполнена правильно с учетом 1-2 мелких погрешностей или 2-3 недочетов, исправленных самостоятельно по требованию преподавателя.

Оценка «удовлетворительно» выставляется, если: работа выполнена правильно не менее чем наполовину, допущены 1-2 погрешности или одна грубая ошибка.

Оценка «неудовлетворительно» выставляется, если: допущены две (и более) грубые ошибки в ходе работы, которые обучающийся не может исправить даже по требованию преподавателя или работа не выполнена полностью.

Типовые тесты

Тест №1

Тема 1. Линейные динамические информационные структуры

1. Какой из следующих типов структур данных относится к линейным динамическим структурам?

A) Массив

B) Связанный список +

C) Стек

D) Дерево

2. Какой из следующих методов используется для добавления элемента в конец связанного списка?

A) InsertAtBeginning

B) Append +

C) Remove

D) Search

3. Какова основная причина использования динамических структур данных вместо статических?

A) Простота реализации

B) Более низкая скорость доступа

C) Эффективное использование памяти +

D) Упрощение алгоритмов

4. Что происходит при удалении элемента из связанного списка?

A) Элемент просто помечается как удаленный

B) Указатели на соседние элементы обновляются +

C) Все элементы сдвигаются влево

D) Список становится пустым

5. Какой из следующих вариантов является основным преимуществом стека?

A) Доступ к элементам в произвольном порядке

B) Легкость добавления и удаления элементов +

C) Хранение элементов в отсортированном порядке

D) Поддержка многопоточности

6. Какое время занимает операция доступа к элементу в связанном списке, если известен его индекс?

A) O(1)

B) O(n) +

C) O(log n)

D) O(n^2)

7. Какой из следующих методов позволяет пройти по всем элементам связанного списка?

A) Search

B) Traverse +

C) Insert

D) Delete

8. Какой тип данных используется для реализации динамического

массива?

A) Стек

B) Очередь

C) Связанный список

D) Вектор +

9. Какой из следующих методов является неэффективным при работе с динамическим массивом?

A) Добавление элемента в конец массива (при наличии свободного места)

B) Удаление элемента из середины массива +

C) Доступ к элементу по индексу

D) Итерация по элементам массива

10. Какой из следующих терминов описывает структуру данных, в которой элементы добавляются и удаляются по принципу "последний пришел — первый вышел" (LIFO)?

A) Очередь

B) Массив

C) Стек +

D) Связанный список

Критерии оценивания тестовых заданий для текущего контроля

Максимальное количество баллов за решение одного теста составляет 5 баллов, которые переводятся в академические оценки следующим образом:

- оценка «отлично» (5 баллов) выставляется при условии правильного ответа студента не менее чем на 85% контрольных заданий

- оценка «хорошо» (4 балла) выставляется при условии правильного ответа студента не менее чем на 70% контрольных заданий

- оценка «удовлетворительно» (3 балла) выставляется при условии правильного ответа студента не менее чем на 51% контрольных заданий

- оценка «неудовлетворительно» (1-2 балла) выставляется при условии правильного ответа студента менее чем на 50% контрольных заданий

Тест №2

Тема 2. Стек. Очередь. Дек.

1. Какой из следующих типов структур данных работает по принципу "последний пришел — первый вышел" (LIFO)?

А) Очередь

Б) Стек +

В) Дек

Г) Массив

2. Какой из следующих типов структур данных работает по принципу "первый пришел — первый вышел" (FIFO)?

А) Стек

Б) Очередь +

В) Связанный список

Г) Дек

3. Какой метод используется для добавления элемента в стек?
А) Enqueue
Б) Push +
В) Insert
Г) Append
4. Какой метод используется для удаления элемента из очереди?
А) Pop
Б) Dequeue +
В) Remove
Г) Exit
5. Какое время занимает операция добавления элемента в конец очереди?
А) O(n)
Б) O(1) +
В) O(log n)
Г) O(n^2)
6. Какой из следующих методов не является частью интерфейса стека?
А) Push
Б) Pop
В) Peek
Г) Dequeue +
7. Какой тип структуры данных позволяет добавлять и удалять элементы с обеих сторон?
А) Стек
Б) Очередь
В) Дек +
Г) Массив
8. Какой метод используется для получения верхнего элемента стека без его удаления?
А) Pop
Б) Peek +
В) Top
Г) Get
9. Какое свойство деков позволяет использовать их как стек и очередь одновременно?
А) Доступ к элементам по индексу
Б) Возможность добавления и удаления с обеих сторон +
В) Ограничение по размеру
Г) Сортировка элементов
10. Какой из следующих методов используется для добавления элемента в очередь?
А) Pop
Б) Push
В) Enqueue +

Г) Remove

Критерии оценивания тестовых заданий для текущего контроля

Максимальное количество баллов за решение одного теста составляет 5 баллов, которые переводятся в академические оценки следующим образом:

- оценка «отлично» (5 баллов) выставляется при условии правильного ответа студента не менее чем на 85% контрольных заданий
- оценка «хорошо» (4 балла) выставляется при условии правильного ответа студента не менее чем на 70% контрольных заданий
- оценка «удовлетворительно» (3 балла) выставляется при условии правильного ответа студента не менее чем на 51% контрольных заданий
- оценка «неудовлетворительно» (1-2 балла) выставляется при условии правильного ответа студента менее чем на 50% контрольных заданий

Тест №3

Тема 3. Моделирование стека средствами языка С++.

1. Какой стандартный контейнер в С++ можно использовать для реализации стека?

А) vector
Б) list
В) deque +
Г) map

2. Какой метод используется для добавления элемента в стек в С++?

А) push_back
Б) insert
В) push +
Г) append

3. Какой метод используется для удаления верхнего элемента из стека в С++?

А) remove
Б) pop +
В) erase
Г) delete

4. Какой метод позволяет получить верхний элемент стека без его удаления?

А) top +
Б) peek
В) front
Г) get

5. Какой заголовочный файл необходимо подключить для использования стандартного стека в С++?

А) <vector>
Б) <deque>
В) <stack> +
Г) <list>

6. Какой тип данных обычно используется для реализации стека в C++?

А) int

Б) char

В) любой тип (шаблоны) +

Г) только указатели

7. Какое время занимает операция добавления элемента в стек?

А) O(n)

Б) O(1) +

В) O(log n)

Г) O(n^2)

8. Как можно проверить, пуст ли стек в C++?

А) isEmpty()

Б) empty() +

В) size() == 0

Г) count() == 0

9. Какой из следующих методов возвращает количество элементов в стеке?

А) size() +

Б) length()

В) count()

Г) capacity()

10. Какой из следующих операторов может быть использован для создания стека с использованием STL в C++?

А) stack<int> myStack; +

Б) stack myStack<int>;

В) stack{int} myStack;

Г) stack<int> myStack();

Критерии оценивания тестовых заданий для текущего контроля

Максимальное количество баллов за решение одного теста составляет 5 баллов, которые переводятся в академические оценки следующим образом:

- оценка «отлично» (5 баллов) выставляется при условии правильного ответа студента не менее чем на 85% контрольных заданий

- оценка «хорошо» (4 балла) выставляется при условии правильного ответа студента не менее чем на 70% контрольных заданий

- оценка «удовлетворительно» (3 балла) выставляется при условии правильного ответа студента не менее чем на 51% контрольных заданий

- оценка «неудовлетворительно» (1-2 балла) выставляется при условии правильного ответа студента менее чем на 50% контрольных заданий

Тест №4

Тема 4. Моделирование очереди средствами языка C++.

1. Какой стандартный контейнер в C++ можно использовать для реализации очереди?

А) vector

Б) list
В) deque +
Г) map

2. Какой метод используется для добавления элемента в очередь в C++?
А) push_back
Б) enqueue +
В) insert
Г) append

3. Какой метод используется для удаления элемента из начала очереди в C++?
А) remove
Б) dequeue +
В) erase
Г) delete

4. Какой метод позволяет получить элемент в начале очереди без его удаления?
А) front +
Б) peek
В) top
Г) get

5. Какой заголовочный файл необходимо подключить для использования стандартной очереди в C++?
А) <vector>
Б) <deque>
В) <queue> +
Г) <list>

6. Какой тип данных обычно используется для реализации очереди в C++?
А) int
Б) char
В) любой тип (шаблоны) +
Г) только указатели

7. Какое время занимает операция добавления элемента в очередь?
А) O(n)
Б) O(1) +
В) O(log n)
Г) O(n^2)

8. Как можно проверить, пуста ли очередь в C++?
А) isEmpty()
Б) empty() +
В) size() == 0
Г) count() == 0

9. Какой из следующих методов возвращает количество элементов в очереди?
А) size() +
Б) length()

В) count()
Г) capacity()

10. Какой из следующих операторов может быть использован для создания очереди с использованием STL в C++?

А) queue<int> myQueue; +
Б) queue myQueue<int>;
В) queue{int} myQueue;
Г) queue<int> myQueue()

Критерии оценивания тестовых заданий для текущего контроля

Максимальное количество баллов за решение одного теста составляет 5 баллов, которые переводятся в академические оценки следующим образом:

- оценка «отлично» (5 баллов) выставляется при условии правильного ответа студента не менее чем на 85% контрольных заданий
- оценка «хорошо» (4 балла) выставляется при условии правильного ответа студента не менее чем на 70% контрольных заданий
- оценка «удовлетворительно» (3 балла) выставляется при условии правильного ответа студента не менее чем на 51% контрольных заданий
- оценка «неудовлетворительно» (1-2 балла) выставляется при условии правильного ответа студента менее чем на 50% контрольных заданий

Тест №5

Тема 5. Кольцевой буфер

1. Что такое кольцевой буфер?

А) Структура данных, которая позволяет динамически изменять размер
Б) Статическая структура данных с фиксированным размером +
В) Структура данных, реализующая стек
Г) Односвязный список

2. Какое основное преимущество кольцевого буфера?

А) Простота реализации
Б) Эффективное использование памяти +
В) Быстрая сортировка
Г) Поддержка произвольного доступа

3. Какой метод используется для добавления элемента в кольцевой буфер?

А) push_back
Б) enqueue
В) insert
Г) write +

4. Какой метод используется для удаления элемента из кольцевого буфера?

А) pop
Б) dequeue
В) read +

Г) remove

5. Как можно проверить, полон ли кольцевой буфер?

А) isFull() +

Б) full()

В) size() == capacity()

Г) count() == max_size()

6. Как можно проверить, пуст ли кольцевой буфер?

А) isEmpty() +

Б) empty()

В) size() == 0

Г) count() == 0

7. Какое время занимает операция добавления элемента в кольцевой буфер?

А) O(n)

Б) O(1) +

В) O(log n)

Г) O(n^2)

8. Какова сложность доступа к элементам в кольцевом буфере?

А) O(1) +

Б) O(n)

В) O(log n)

Г) O(n^2)

9. Какой из следующих операторов может быть использован для создания кольцевого буфера на основе массива?

А) int buffer[size]; +

Б) array<int> buffer[size];

В) vector<int> buffer(size);

Г) list<int> buffer(size);

10. Как реализовать кольцевой буфер в C++?

А) Использовать массив и два указателя (начало и конец) +

Б) Использовать только динамическую память

В) Использовать только статический массив без указателей

Г) Использовать STL контейнеры без дополнительных структур

Критерии оценивания тестовых заданий для текущего контроля

Максимальное количество баллов за решение одного теста составляет 5 баллов, которые переводятся в академические оценки следующим образом:

- оценка «отлично» (5 баллов) выставляется при условии правильного ответа студента не менее чем на 85% контрольных заданий

- оценка «хорошо» (4 балла) выставляется при условии правильного ответа студента не менее чем на 70% контрольных заданий

- оценка «удовлетворительно» (3 балла) выставляется при условии правильного ответа студента не менее чем на 51% контрольных заданий

- оценка «неудовлетворительно» (1-2 балла) выставляется при условии правильного ответа студента менее чем на 50% контрольных заданий

Тест №6

Тема 6. Реализация двух однотипных стеков

1. Что такое стек?

А) Структура данных, работающая по принципу FIFO (First In First Out).

Б) Специальный тип переменных для хранения чисел.

В) Структура данных, реализующая принцип LIFO (Last In First Out). +

Г) Коллекция элементов, упорядоченная по значению ключа.

2. Какое основное ограничение массива при реализации стека?

А) Невозможность хранить элементы разных типов.

Б) Фиксированный размер памяти, выделяемый заранее. +

В) Нельзя добавлять новые элементы.

Г) Массив не поддерживает операции push и pop.

3. Какой метод используется для добавления элемента в стек?

А) shift()

Б) unshift()

В) push() +

Г) enqueue()

4. Метод для удаления верхнего элемента из стека называется...

А) remove()

Б) dequeue()

В) pop() +

Г) delete()

5. Для чего чаще всего используют два однотипных стека вместе?

А) Увеличение производительности системы.

Б) Уменьшение размера программы.

В) Реализация очереди или преобразование постфиксной записи выражений. +

Г) Оптимизация операций чтения.

6. Если реализовать очередь через два стека, какой порядок действий необходим для извлечения первого элемента?

А) Удалять верхний элемент из первого стека.

Б) Переложить элементы из первого стека во второй, затем удалить верхний элемент второго стека. +

В) Переместить первый элемент из второго стека обратно в первый стек.

Г) Очистить оба стека и вставить новый элемент.

7. Какова сложность алгоритма push и pop при реализации одиночного стека массивом?

А) $O(n^2)$

Б) $O(\log n)$

В) $O(1)$ +

Г) $O(n)$

8. Что произойдет, если попытаться достать элемент из пустого стека?

А) Возвратится значение null.

Б) Возможна ошибка или исключение. +

В) Ничего не произойдёт.

Г) Элемент автоматически добавляется в стек.

9. Зачем иногда применяют двойную реализацию стеков?

А) Повышения скорости обработки запросов.

Б) Обеспечения дополнительной функциональности или структуры данных (например, очередь). +

В) Экономии оперативной памяти.

Г) Удобства написания программного кода.

10. Какие ограничения существуют при реализации стека с использованием связанного списка?

А) Сложность реализации методов push и pop.

Б) Нехватка памяти при большом количестве элементов.

В) Дополнительные затраты памяти на хранение ссылок между элементами. +

Г) Нет ограничений.

Критерии оценивания тестовых заданий для текущего контроля

Максимальное количество баллов за решение одного теста составляет 5 баллов, которые переводятся в академические оценки следующим образом:

- оценка «отлично» (5 баллов) выставляется при условии правильного ответа студента не менее чем на 85% контрольных заданий

- оценка «хорошо» (4 балла) выставляется при условии правильного ответа студента не менее чем на 70% контрольных заданий

- оценка «удовлетворительно» (3 балла) выставляется при условии правильного ответа студента не менее чем на 51% контрольных заданий

- оценка «неудовлетворительно» (1-2 балла) выставляется при условии правильного ответа студента менее чем на 50% контрольных заданий

Тест № 7

Тема 7. Однонаправленные списки. Операция вставки и удаления элемента

1. Чем характеризуется однонаправленный список?

А) Каждый узел хранит ссылку на предыдущий элемент.

Б) Все узлы связаны друг с другом двусторонними ссылками.

В) Узлы содержат ссылку только на следующий элемент. +

Г) Список имеет фиксированную длину.

2. Что представляет собой операция вставки нового узла в начало однонаправленного списка?

А) Добавление нового узла перед первым элементом без изменения ссылок.

Б) Создание нового узла и изменение ссылки текущего первого элемента таким образом, чтобы новый узел стал началом списка. +

В) Вставка нового узла после последнего элемента списка.

Г) Изменение значений всех узлов списка.

3. Как выполняется удаление первого элемента из однонаправленного списка?

А) Просто удаляется ссылка на последний элемент.

Б) Меняется ссылка заголовочного узла, указывая теперь на второй элемент списка. +

В) Происходит циклический сдвиг всех элементов списка назад.

Г) Требуется повторная инициализация всей структуры.

4. Где проще всего добавить новый элемент в однонаправленном списке?

А) После произвольного элемента.

Б) Перед последним элементом.

В) В начало списка. +

Г) Между двумя любыми узлами.

5. Какая основная проблема возникает при удалении промежуточного элемента из однонаправленного списка?

А) Нужно изменить значения всех последующих элементов.

Б) Необходимо предварительно найти предшествующий элемент, чтобы скорректировать его ссылку. +

В) Список становится двунаправленным.

Г) Отсутствует доступ к предыдущему элементу списка.

6. Что означает термин «головной узел» в однонаправленных списках?

А) Последний узел списка.

Б) Любой узел, содержащий нулевую ссылку.

В) Первый узел списка, откуда начинается обход. +

Г) Узел с наибольшим значением.

7. Как выглядит процедура вставки элемента в конец однонаправленного списка?

А) Создается новый узел, и головная ссылка обновляется на этот узел.

Б) Последовательный проход по всему списку для нахождения последнего узла, затем обновление его ссылки на новый узел. +

В) Новый узел помещается сразу после головного узла.

Г) Новая структура создается заново.

8. В каком случае потребуется перебор всех элементов однонаправленного списка?

А) При удалении последнего элемента.

Б) При поиске конкретного элемента или вставке нового в конце списка. +

В) При создании нового списка.

Г) Никогда.

9. Какую операцию над списком легче всего выполнить в однонаправленном списке?

А) Поиск минимального элемента.

Б) Проверка наличия заданного элемента.

В) Вставку нового элемента в начало списка. +

Г) Определение длины списка.

10. Почему однонаправленные списки называют также односвязанными?

А) Потому что они имеют фиксированное количество элементов.

Б) Из-за возможности проходить список только вперед и назад одновременно.

В) Так как каждый узел хранит ровно одну ссылку на следующий элемент. +

Г) Это название обусловлено отсутствием необходимости удалять элементы.

Критерии оценивания тестовых заданий для текущего контроля

Максимальное количество баллов за решение одного теста составляет 5 баллов, которые переводятся в академические оценки следующим образом:

- оценка «отлично» (5 баллов) выставляется при условии правильного ответа студента не менее чем на 85% контрольных заданий
- оценка «хорошо» (4 балла) выставляется при условии правильного ответа студента не менее чем на 70% контрольных заданий
- оценка «удовлетворительно» (3 балла) выставляется при условии правильного ответа студента не менее чем на 51% контрольных заданий
- оценка «неудовлетворительно» (1-2 балла) выставляется при условии правильного ответа студента менее чем на 50% контрольных заданий

Тест № 8

Тема 8. Двунаправленные списки. Построение, операция удаления и вставки элемента

1. Что отличает двунаправленный список от однонаправленного?

А) Возможность хранить больше данных в каждом узле.

Б) Наличие у каждого узла ссылок на предыдущий и последующий элементы. +

В) Только первая запись хранит дополнительную информацию.

Г) Отсутствие головной ссылки.

2. Как осуществляется вставка нового элемента в середину двунаправленного списка?

А) Меняются ссылки соседних узлов, указывающие на новый элемент.

Б) Новому узлу присваиваются ссылки на соседние узлы, а ссылки соседей меняются соответствующим образом. +

В) Создаются копии предыдущих и последующих узлов.

Г) Выполняется полный обход списка для обновления ссылок.

3. Как производится удаление промежуточного элемента из двунаправленного списка?

А) Корректируется лишь ссылка предыдущего элемента.

Б) Предшествующий и последующий узлы меняют ссылки друг на друга, исключая удаленный элемент. +

В) Удаляется весь список начиная с выбранного элемента.

Г) Автоматически освобождается память, занимаемая данным элементом.

4. В чём преимущество двунаправленных списков перед

однонаправленными?

А) Они занимают меньше места в памяти.

Б) Можно эффективно перемещаться как вперёд, так и назад по списку. +

В) Быстрее выполняются операции сортировки.

Г) Проще поддерживать уникальный индекс каждого элемента.

5. Как изменяется головная ссылка при удалении первого элемента двунаправленного списка?

А) Головная ссылка остаётся неизменной.

Б) Она переустанавливается на следующий элемент, предыдущий элемент которого очищается. +

В) Голова ссылается на хвост списка.

Г) Голова теряет связь с остальным списком.

6. Какая особенность позволяет легко осуществлять быструю вставку элемента в любом месте двунаправленного списка?

А) Использование дополнительного вспомогательного массива.

Б) Присутствие обратных ссылок, позволяющих быстро добраться до нужного узла. +

В) Одновременное наличие прямого и обратного порядков индексации.

Г) Неограниченное расширение памяти под структуру данных.

7. Сколько связей надо обновить при удалении среднего элемента из двунаправленного списка?

А) Одна.

Б) Три.

В) Две. +

Г) Четыре.

8. Когда удобнее всего применять двунаправленные списки?

А) Когда важна экономия памяти.

Б) Когда часто требуются операции просмотра элементов в обоих направлениях. +

В) При необходимости быстрой проверки уникальности элементов.

Г) При построении больших иерархических структур.

9. Какой способ вставки нового элемента в двунаправленный список считается наиболее эффективным?

А) Всегда добавлять новый элемент в начало списка.

Б) Найти подходящую позицию и выполнить соответствующую коррекцию ссылок. +

В) Постоянно сохранять отсортированность списка при каждой вставке.

Г) Периодически перестраивать весь список.

10. В чём заключается главное отличие процедуры удаления элемента из двунаправленного списка по сравнению с однонаправленным?

А) Процесс требует значительно большего количества шагов.

Б) Легче определить и исправить связи с соседними элементами благодаря наличию обратной ссылки. +

В) Используется совершенно иной алгоритм управления памятью.

Г) Никакого отличия нет.

Критерии оценивания тестовых заданий для текущего контроля

Максимальное количество баллов за решение одного теста составляет 5 баллов, которые переводятся в академические оценки следующим образом:

- оценка «отлично» (5 баллов) выставляется при условии правильного ответа студента не менее чем на 85% контрольных заданий

- оценка «хорошо» (4 балла) выставляется при условии правильного ответа студента не менее чем на 70% контрольных заданий

- оценка «удовлетворительно» (3 балла) выставляется при условии правильного ответа студента не менее чем на 51% контрольных заданий

- оценка «неудовлетворительно» (1-2 балла) выставляется при условии правильного ответа студента менее чем на 50% контрольных заданий

Тест №9

Тема 9. Деревья. Дерево двоичного поиска. Обходы ДДП

1. Что такое дерево двоичного поиска (ДДП)?

А) Это структура данных, в которой каждое левое дочернее значение меньше родительского, а правое больше родителя +

Б) Это упорядоченный линейный список

В) Это граф, состоящий из узлов и рёбер

Г) Это сбалансированное бинарное дерево

2. Какой обход дерева двоичного поиска называется прямым (preorder)?

А) Сначала обрабатывается корень, потом левое поддерево, затем правое поддерево +

Б) Сначала обрабатываются листья, затем внутренние узлы

В) Сначала обрабатывается левое поддерево, затем корень, затем правое поддерево

Г) Сначала обрабатывается правое поддерево, затем корень, затем левое поддерево

3. При каком обходе ДДП элементы выводятся в порядке возрастания?

А) Прямой обход (preorder)

Б) Инфиксный обход (inorder) +

В) Постфиксный обход (postorder)

Г) Случайный порядок

4. Какие свойства выполняются в дереве двоичного поиска?

А) Левое поддерево любого узла содержит значения меньшие, чем узел, а правое — большие +

Б) Правое поддерево любого узла содержит значения меньшие, чем узел, а левое — большие

В) Все значения дерева расположены случайным образом

Г) Любое дерево является полным бинарным деревом

5. Для чего применяется постфиксный обход (postorder)?

А) Чтобы удалить дерево начиная с листьев +

Б) Чтобы вывести элементы в убывающем порядке

В) Чтобы обработать сначала корни

Г) Чтобы определить высоту дерева

6. Обход inorder обеспечивает вывод ключей в каком порядке?

А) По возрастанию +

Б) По убыванию

В) От большего к меньшему

Г) От правого листа к корню

7. Почему важно поддерживать балансировку дерева двоичного поиска?

А) Балансировка улучшает производительность операций поиска, вставки и удаления+

Б) Без балансировки невозможно производить обход дерева

В) Балансировка уменьшает потребление памяти

Г) Балансировка предотвращает дублирование данных

8. Какова сложность алгоритма поиска в идеально сбалансированном дереве двоичного поиска?

А) O(log n) +

Б) O(n log n)

В) $O(n^2)$

Г) O(1)

9. Что произойдёт, если вставлять ключи в дерево двоичного поиска последовательно по порядку увеличения?

А) Получится почти полное бинарное дерево

Б) Получится практически цепочка (вырожденное дерево) +

В) Элементы будут равномерно распределены по уровням

Г) Ничего особенного не произойдёт

10В чём заключается идея рекурсивного обхода дерева?

А) Посещение всех возможных путей графа

Б) Повторение одной и той же процедуры обработки над каждым узлом, используя её для обработки детей +

В) Подсчет суммы всех элементов дерева

Г) Нахождение минимального пути между двумя узлами

Критерии оценивания тестовых заданий для текущего контроля

Максимальное количество баллов за решение одного теста составляет 5 баллов, которые переводятся в академические оценки следующим образом:

- оценка «отлично» (5 баллов) выставляется при условии правильного ответа студента не менее чем на 85% контрольных заданий

- оценка «хорошо» (4 балла) выставляется при условии правильного ответа студента не менее чем на 70% контрольных заданий

- оценка «удовлетворительно» (3 балла) выставляется при условии правильного ответа студента не менее чем на 51% контрольных заданий

- оценка «неудовлетворительно» (1-2 балла) выставляется при условии правильного ответа студента менее чем на 50% контрольных заданий

Тест № 10

Тема 10. Обработка дерева двоичного поиска.

1. Что значит "бинарное дерево"?

А) Древовидная структура, в которой каждая вершина связана только с одной родительской вершиной.

Б) Структура данных, организованная линейно.

В) Граф, состоящий исключительно из изолированных точек.

Г) Дерево, где каждая вершина имеет максимум две ветви: левую и правую. +

2. Что характерно для дерева двоичного поиска (Binary Search Tree, BST)?

А) Значения в левом поддереве больше, чем в правом.

Б) Правые ветви содержат отрицательные значения.

В) Каждое дерево должно содержать чётное количество вершин.

Г) В левой части каждого узла хранятся меньшие значения, а в правой — большие. +

3. Каково назначение метода insert в дереве двоичного поиска?

А) Поиск наименьшего значения в дереве.

Б) Удаление указанного узла из дерева.

В) Переопределение порядка сортировки дерева.

Г) Вставка нового узла в правильное положение согласно правилам BST. +

4. Какова цель процедуры balance в дереве двоичного поиска?

А) Установка специального атрибута балансировки каждому узлу

.Б) Обнуление существующих данных в дереве.

В) Вычисление глубины каждого узла.

Г) Приведение дерева к сбалансированной форме для повышения эффективности операций. +

5. Какой алгоритм применяется для преобразования обычного бинарного дерева в сбалансированное?

А) Insertion Sort

Б) Bubble Sort

В) Merge Sort

Г) AVL или Red-Black деревья +

6. Что обеспечивает оптимальный путь поиска в дереве двоичного поиска?

А) Применение хеш-функций для быстрого сопоставления значений.

Б) Непосредственное сравнение всех узлов подряд.

В) Регулярное перемешивание узлов случайным образом.

Г) Рекурсивный спуск по ветвям дерева в зависимости от сравниваемых значений. +

7. Как называется ситуация, когда глубина левого и правого поддеревьев отличается незначительно?

А) Несбалансированная структура

Б) Асимметрия графа

В) Хвостовая оптимизация

Г) Балансировка дерева +

8. Какой метод используется для поиска минимальной величины в дереве

двоичного поиска?

А) Обход сверху-вниз по дереву в случайном порядке.

Б) Сравнение всех возможных путей к листовым узлам.

В) Пересчёт среднего арифметического значений всех узлов.

Г) Обход дерева вниз по левым ветвям пока не достигнется крайняя левая точка. +

9. Как правильно провести процедуру удаления узла в дереве двоичного поиска?

А) Просто стереть узел и сохранить структуру без изменений.

Б) Переписать все остальные узлы вверх.

В) Оставить пустое пространство там, где находился удалённый узел.

Г) Найти подходящего преемника или предшественника и заменить ими удалённый узел. +

10. Каковы основные типы обхода дерева двоичного поиска?

А) Pre-traversal, post-traversal, sideways-traversal

Б) Ascending-only, descending-only, random-access

В) Sequential-search, binary-search, hash-based search

Г) Preorder, inorder, postorder +

Критерии оценивания тестовых заданий для текущего контроля

Максимальное количество баллов за решение одного теста составляет 5 баллов, которые переводятся в академические оценки следующим образом:

- оценка «отлично» (5 баллов) выставляется при условии правильного ответа студента не менее чем на 85% контрольных заданий

- оценка «хорошо» (4 балла) выставляется при условии правильного ответа студента не менее чем на 70% контрольных заданий

- оценка «удовлетворительно» (3 балла) выставляется при условии правильного ответа студента не менее чем на 51% контрольных заданий

- оценка «неудовлетворительно» (1-2 балла) выставляется при условии правильного ответа студента менее чем на 50% контрольных заданий

Тест №11.

Тема 11. Система управления вводом-выводом. Буферизация ввода-вывода

1. Что такое система управления вводом-выводом?

А) Программа для обработки данных

Б) Аппаратное обеспечение для хранения данных

В) Комплекс программных и аппаратных средств для управления обменом данных между устройствами и процессором +

Г) Операционная система

2. Какова основная функция буфера в системе ввода-вывода?

А) Увеличение скорости обработки данных

Б) Хранение данных только для вывода

В) Временное хранение данных для сглаживания разницы в скорости между устройствами +

Г) Сжатие данных перед их отправкой

3. Какой из следующих типов буферов обычно используется в системах ввода-вывода?

А) Динамический буфер

Б) Фиксированный буфер +

В) Статический буфер

Г) Все перечисленные

4. Какой из следующих факторов влияет на эффективность системы ввода-вывода?

А) Размер оперативной памяти

Б) Скорость процессора

В) Размер и управление буфером +

Г) Все перечисленные факторы

5. Что происходит, если буфер переполнен?

А) Данные автоматически удаляются

Б) Система начинает работать быстрее

В) Данные могут быть потеряны или возникнут задержки в обработке +

Г) Ничего не происходит

6. Какой тип буферизации позволяет одновременно записывать и читать данные?

А) Однонаправленная буферизация

Б) Двунаправленная буферизация +

В) Непрерывная буферизация

Г) Блокирующая буферизация

7. Какой из следующих методов используется для управления буферизацией?

А) FIFO (first in, first out) +

Б) LIFO (last in, first out)

В) Принцип случайного доступа

Г) Все перечисленные

8. Какой из следующих компонентов не является частью системы управления вводом-выводом?

А) Дисплей

Б) Жесткий диск

В) Оперативная память +

Г) Клавиатура

9. Какой из следующих методов буферизации позволяет уменьшить время ожидания при вводе-выводе?

А) Статическая буферизация

Б) Динамическая буферизация

В) Синхронная буферизация

Г) Асинхронная буферизация +

10. Какова основная цель системы управления вводом-выводом?

А) Увеличение объема памяти

Б) Повышение производительности обработки данных

В) Обеспечение эффективного обмена данными между устройствами и процессором+

Г) Сжатие данных для экономии места

Критерии оценивания тестовых заданий для текущего контроля

Максимальное количество баллов за решение одного теста составляет 5 баллов, которые переводятся в академические оценки следующим образом:

- оценка «отлично» (5 баллов) выставляется при условии правильного ответа студента не менее чем на 85% контрольных заданий

- оценка «хорошо» (4 балла) выставляется при условии правильного ответа студента не менее чем на 70% контрольных заданий

- оценка «удовлетворительно» (3 балла) выставляется при условии правильного ответа студента не менее чем на 51% контрольных заданий

- оценка «неудовлетворительно» (1-2 балла) выставляется при условии правильного ответа студента менее чем на 50% контрольных заданий

Тест №12

Тема 12. Общие операции над файлами

1. Что такое система управления вводом-выводом?

А) Программа для обработки данных

Б) Аппаратное обеспечение для хранения данных

В) Комплекс программных и аппаратных средств для управления обменом данных между устройствами и процессором +

Г) Операционная система

2. Какова основная функция буфера в системе ввода-вывода?

А) Увеличение скорости обработки данных

Б) Хранение данных только для вывода

В) Временное хранение данных для сглаживания разницы в скорости между устройствами +

Г) Сжатие данных перед их отправкой

3. Какой из следующих типов буферов обычно используется в системах ввода-вывода?

А) Динамический буфер

Б) Фиксированный буфер +

В) Статический буфер

Г) Все перечисленные

4. Какой из следующих факторов влияет на эффективность системы ввода-вывода?

А) Размер оперативной памяти

Б) Скорость процессора

В) Размер и управление буфером +

Г) Все перечисленные факторы

5. Что происходит, если буфер переполнен?

А) Данные автоматически удаляются

Б) Система начинает работать быстрее

В) Данные могут быть потеряны или возникнут задержки в обработке +

Г) Ничего не происходит

6. Какой тип буферизации позволяет одновременно записывать и читать данные?

А) Однонаправленная буферизация

Б) Двунаправленная буферизация +

В) Непрерывная буферизация

Г) Блокирующая буферизация

7. Какой из следующих методов используется для управления буферизацией?

А) FIFO (first in, first out) +

Б) LIFO (last in, first out)

В) Принцип случайного доступа

Г) Все перечисленные

8. Какой из следующих компонентов не является частью системы управления вводом-выводом?

А) Дисплей

Б) Жесткий диск

В) Оперативная память +

Г) Клавиатура

9. Какой из следующих методов буферизации позволяет уменьшить время ожидания при вводе-выводе?

А) Статическая буферизация

Б) Динамическая буферизация

В) Синхронная буферизация

Г) Асинхронная буферизация +

10. Какова основная цель системы управления вводом-выводом?

А) Увеличение объема памяти

Б) Повышение производительности обработки данных

В) Обеспечение эффективного обмена данными между устройствами и процессором+

Г) Сжатие данных для экономии места

Критерии оценивания тестовых заданий для текущего контроля

Максимальное количество баллов за решение одного теста составляет 5 баллов, которые переводятся в академические оценки следующим образом:

- оценка «отлично» (5 баллов) выставляется при условии правильного ответа студента не менее чем на 85% контрольных заданий

- оценка «хорошо» (4 балла) выставляется при условии правильного ответа студента не менее чем на 70% контрольных заданий

- оценка «удовлетворительно» (3 балла) выставляется при условии правильного ответа студента не менее чем на 51% контрольных заданий

- оценка «неудовлетворительно» (1-2 балла) выставляется при условии

правильного ответа студента менее чем на 50% контрольных заданий

Тест №13

Тема 13. Ориентированные графы. Представления ориентированных графов.

1. Что такое файл в контексте операционных систем?

А) Программа для обработки данных

Б) Единица хранения информации на носителе +

В) Устройство вывода данных

Г) Объект, используемый для сетевой передачи данных

2. Какой из следующих методов используется для открытия файла?

А) open() +

Б) read()

В) write()

Г) close()

3. Какое действие выполняет функция read()?

А) Считывает данные из файла +

Б) Записывает данные в файл

В) Закрывает файл

Г) Открывает файл для записи

4. Что происходит, если попытаться записать данные в файл, который открыт только для чтения?

А) Произойдет ошибка +

Б) Данные будут записаны

В) Файл автоматически откроется для записи

Г) Ничего не произойдет

5. Какой из следующих режимов открытия файла позволяет добавлять данные в конец файла?

А) r

Б) w

В) a +

Г) r+

6. Что делает функция close()?

А) Закрывает открытый файл и освобождает ресурсы +

Б) Открывает новый файл

В) Считывает данные из файла

Г) Записывает данные в файл

7. Какой из следующих форматов обычно используется для текстовых файлов?

А) .exe

Б) .bin

В) .txt +

Г) .jpg

8. Какой метод позволяет перемещать указатель файла?

А) read()

Б) write()
В) seek() +
Г) close()
9. Какой из следующих методов используется для удаления файла?
А) delete()
Б) clear()
В) remove() +
Г) erase()
10. Какая операция позволяет скопировать содержимое одного файла в другой?
А) copy() +
Б) move()
В) write()
Г) read()

Критерии оценивания тестовых заданий для текущего контроля

Максимальное количество баллов за решение одного теста составляет 5 баллов, которые переводятся в академические оценки следующим образом:

- оценка «отлично» (5 баллов) выставляется при условии правильного ответа студента не менее чем на 85% контрольных заданий
- оценка «хорошо» (4 балла) выставляется при условии правильного ответа студента не менее чем на 70% контрольных заданий
- оценка «удовлетворительно» (3 балла) выставляется при условии правильного ответа студента не менее чем на 51% контрольных заданий
- оценка «неудовлетворительно» (1-2 балла) выставляется при условии правильного ответа студента менее чем на 50% контрольных заданий

Тест№14

Тема 14. Задача нахождения кратчайшего пути.

1. Что такое задача нахождения кратчайшего пути?
А) Задача, связанная с нахождением самого длинного пути в графе
Б) Задача нахождения пути с минимальной суммарной длиной или весом между двумя вершинами в графе +
В) Задача, которая не имеет решения
Г) Задача нахождения всех возможных путей в графе
2. Какой из следующих алгоритмов обычно используется для нахождения кратчайшего пути в графе с неотрицательными весами?
А) Алгоритм Дейкстры +
Б) Алгоритм Флойда-Уоршелла
В) Алгоритм Краскала
Г) Алгоритм Беллмана-Форда
3. Какой алгоритм используется для нахождения кратчайшего пути в графах с отрицательными весами?
А) Алгоритм Дейкстры
Б) Алгоритм Беллмана-Форда +

В) Алгоритм Краскала

Г) Алгоритм Флойда-Уоршелла

4. Какой из следующих методов представления графа подходит для алгоритма Дейкстры?

А) Список смежности

Б) Матрица смежности +

В) Линейный список

Г) Дерево

5. Что делает алгоритм Флойда-Уоршелла?

А) Находит кратчайший путь только между двумя вершинами

Б) Находит кратчайшие пути между всеми парами вершин в графе +

В) Находит только циклы в графе

Г) Находит максимальный путь в графе

6. Какой из следующих критериев используется в алгоритме Дейкстры для выбора следующей вершины?

А) Случайный выбор

Б) Вершина с минимальным значением стоимости пути +

В) Вершина с максимальным значением веса

Г) Вершина, которая была недавно посещена

7. Какой из следующих типов графов может иметь несколько кратчайших путей между двумя вершинами?

А) Ориентированный граф

Б) Неориентированный граф +

В) Дерево

Г) Граф без циклов

8. Какое из следующих утверждений верно для алгоритма Дейкстры?

А) Он работает только для ориентированных графов

Б) Он не может работать с отрицательными весами +

В) Он всегда находит циклы

Г) Он использует жадный метод для нахождения решения

9. Какой из следующих методов используется для оптимизации алгоритма Дейкстры?

А) Использование матрицы смежности вместо списка смежности

Б) Использование кучи (очереди с приоритетом) +

В) Применение случайных чисел

Г) Упрощение графа

10. Какое из следующих утверждений о задаче нахождения кратчайшего пути является верным?

А) Задача всегда имеет решение, если граф связный и не содержит циклов +

Б) Задача никогда не имеет решения

В) Задача всегда имеет единственное решение

Г) Задача может быть решена только для ориентированных графов

Критерии оценивания тестовых заданий для текущего контроля

Максимальное количество баллов за решение одного теста составляет 5 баллов, которые переводятся в академические оценки следующим образом:

- оценка «отлично» (5 баллов) выставляется при условии правильного ответа студента не менее чем на 85% контрольных заданий
- оценка «хорошо» (4 балла) выставляется при условии правильного ответа студента не менее чем на 70% контрольных заданий
- оценка «удовлетворительно» (3 балла) выставляется при условии правильного ответа студента не менее чем на 51% контрольных заданий
- оценка «неудовлетворительно» (1-2 балла) выставляется при условии правильного ответа студента менее чем на 50% контрольных заданий

Тест №15

Тема 15. Нахождение кратчайших путей между парами вершин. Алгоритм Флойда.

1. Что такое задача нахождения кратчайших путей между парами вершин?

А) Задача, связанная с нахождением самого длинного пути в графе

Б) Задача нахождения кратчайшего пути между каждой парой вершин в графе +

В) Задача нахождения только одного пути в графе

Г) Задача, которая не имеет решения

2. Какой алгоритм используется для нахождения кратчайших путей между всеми парами вершин?

А) Алгоритм Дейкстры

Б) Алгоритм Флойда-Уоршелла +

В) Алгоритм Краскала

Г) Алгоритм Беллмана-Форда

3. Какова временная сложность алгоритма Флойда-Уоршелла?

А) O(V)

Б) O(V log V)

В) O(V^3) +

Г) O(V^2)

4. Какой тип графа поддерживает алгоритм Флойда-Уоршелла?

А) Только ориентированные графы

Б) Как ориентированные, так и неориентированные графы +

В) Только неориентированные графы

Г) Графы без рёбер

5. Как работает алгоритм Флойда-Уоршелла?

А) Итеративно обновляет матрицу расстояний для всех пар вершин +

Б) Находит кратчайший путь только между двумя вершинами

В) Использует жадный подход для нахождения кратчайших путей

Г) Не требует предварительной инициализации данных

6. Какова цель использования вспомогательной матрицы в алгоритме Флойда-Уоршелла?

А) Для хранения всех рёбер графа

Б) Для отслеживания промежуточных вершин на кратчайших путях +

В) Для хранения весов рёбер

Г) Для определения степени вершин

7. Какова начальная инициализация матрицы расстояний в алгоритме Флойда-Уоршелла?

А) Все значения равны нулю

Б) Все значения равны бесконечности, кроме диагональных элементов +

В) Все значения равны единице

Г) Все значения равны весам рёбер

8. Какой из следующих результатов можно получить по завершении алгоритма Флойда-Уоршелла?

А) Кратчайшие пути только между двумя вершинами

Б) Кратчайшие пути между всеми парами вершин +

В) Только длины рёбер

Г) Количество вершин в графе

9. Какой из следующих типов графов может быть обработан алгоритмом Флойда-Уоршелла?

А) Ориентированные графы с положительными весами

Б) Графы с отрицательными весами, но без отрицательных циклов +

В) Графы только с положительными весами

Г) Графы без рёбер

10. Как можно определить наличие отрицательного цикла в графе с использованием алгоритма Флойда-Уоршелла?

А) Проверить, есть ли рёбра, вес которых равен нулю

Б) Проверить, есть ли отрицательные значения на диагонали матрицы расстояний после завершения алгоритма +

В) Проверить, есть ли изолированные вершины

Г) Проверить, есть ли рёбра, соединяющие вершины с одинаковыми весами

Критерии оценивания тестовых заданий для текущего контроля

Максимальное количество баллов за решение одного теста составляет 5 баллов, которые переводятся в академические оценки следующим образом:

- оценка «отлично» (5 баллов) выставляется при условии правильного ответа студента не менее чем на 85% контрольных заданий

- оценка «хорошо» (4 балла) выставляется при условии правильного ответа студента не менее чем на 70% контрольных заданий

- оценка «удовлетворительно» (3 балла) выставляется при условии правильного ответа студента не менее чем на 51% контрольных заданий

- оценка «неудовлетворительно» (1-2 балла) выставляется при условии правильного ответа студента менее чем на 50% контрольных заданий

Тесть№16

Тема 16. Поиск центра ориентированного графа.

1. Что такое центр ориентированного графа?

А) Вершина, с которой можно добраться до всех других вершин за минимальное время

Б) Вершина, минимизирующая максимальное расстояние до других вершин +

В) Вершина с наибольшей степенью

Г) Вершина, соединенная со всеми другими вершинами

2. Какой из следующих методов чаще всего используется для поиска центра ориентированного графа?

А) Алгоритм Дейкстры

Б) Поиск в ширину (BFS) или поиск в глубину (DFS) для определения расстояний +

В) Алгоритм Флойда-Уоршелла

Г) Алгоритм Краскала

3. Каково основное свойство центра ориентированного графа?

А) Он всегда является единственной вершиной

Б) Он минимизирует максимальное расстояние до всех других вершин +

В) Он всегда связан с рёбрами равной длины

Г) Он находится в середине графа

4. Как можно определить радиус ориентированного графа?

А) Максимальное расстояние от центра до всех других вершин +

Б) Минимальное расстояние между любыми двумя вершинами

В) Сумма всех расстояний между вершинами

Г) Среднее расстояние между всеми вершинами

5. Какое из следующих утверждений верно для центра ориентированного графа?

А) Центр всегда совпадает с одной из наиболее посещаемых вершин

Б) Центр может не существовать, если граф не связан +

В) Центр всегда находится на границе графа

Г) Центр всегда является вершиной с максимальной степенью

6. Как можно найти центр графа, используя алгоритм Флойда-Уоршелла?

А) Определить кратчайшие пути между всеми парами вершин и найти вершину с минимальным максимальным расстоянием +

Б) Найти вершину с минимальным количеством рёбер

В) Найти вершину, соединённую со всеми другими вершинами

Г) Проверить наличие циклов в графе

7. Какой из следующих методов может помочь в визуализации центра графа?

А) Построение матрицы расстояний +

Б) Использование случайных значений

В) Применение жадного алгоритма

Г) Построение кривой для каждой вершины

8. Какое значение имеет "диаметр" ориентированного графа в контексте поиска центра?

А) Максимальное расстояние между любыми двумя вершинами в графе +

Б) Минимальное расстояние от центра до других вершин

В) Среднее расстояние между любыми двумя вершинами

Г) Сумма всех расстояний в графе

9. Какой из следующих подходов может использоваться для нахождения центра ориентированного графа, если граф содержит отрицательные веса?

А) Поиск центров с помощью модификации алгоритма Флойда-Уоршелла +

Б) Невозможно найти центр в графе с отрицательными весами

В) Использование жадных методов

Г) Применение только поиска в глубину

10. Какое из следующих утверждений о центре ориентированного графа является верным?

А) Центр ориентированного графа может быть не единственным +

Б) Центр всегда совпадает с центром тяжести графа

В) Центр всегда находится в верхней части графа

Г) Центр ориентированного графа всегда является конечной вершиной

Критерии оценивания тестовых заданий для текущего контроля

Максимальное количество баллов за решение одного теста составляет 5 баллов, которые переводятся в академические оценки следующим образом:

- оценка «отлично» (5 баллов) выставляется при условии правильного ответа студента не менее чем на 85% контрольных заданий

- оценка «хорошо» (4 балла) выставляется при условии правильного ответа студента не менее чем на 70% контрольных заданий

- оценка «удовлетворительно» (3 балла) выставляется при условии правильного ответа студента не менее чем на 51% контрольных заданий

- оценка «неудовлетворительно» (1-2 балла) выставляется при условии правильного ответа студента менее чем на 50% контрольных заданий

Тест №17

Тема 17. Обход ориентированных графов. Поиск в глубину.

1. Что такое поиск в глубину (DFS) в контексте ориентированных графов?

А) Метод обхода графа, при котором исследуются все возможные пути из одной вершины, прежде чем вернуться назад +

Б) Метод, который находит кратчайший путь между двумя вершинами

В) Метод, который сначала исследует все соседние вершины

Г) Метод, который не использует рекурсию

2. Какой из следующих шагов выполняется при реализации поиска в глубину?

А) Посещение текущей вершины и рекурсивный вызов для всех её соседей +

Б) Сначала посещение всех соседей, а затем возврат к текущей вершине

В) Сортировка вершин по весу

Г) Построение матрицы смежности

3. Какова временная сложность алгоритма поиска в глубину для ориентированного графа?

А) O(V)

Б) O(E)

В) O(V + E) +

Г) O(V^2)

4. Как можно предотвратить зацикливание при выполнении поиска в глубину?

А) Использование массива или множества для отслеживания посещённых вершин +

Б) Проверка весов рёбер

В) Изменение порядка обхода

Г) Использование случайных чисел

5. Какой из следующих случаев может привести к неправильному результату при использовании поиска в глубину?

А) Граф является связным

Б) Граф содержит циклы +

В) Граф имеет одинаковые веса рёбер

Г) Граф состоит из одной вершины

6. Какое утверждение о рекурсивной реализации поиска в глубину является верным?

А) Рекурсия не используется в этой реализации

Б) Каждый вызов функции представляет собой отдельное состояние обхода графа +

В) Рекурсия всегда приводит к ошибкам

Г) Рекурсия позволяет избежать использования стека

7. Как можно использовать поиск в глубину для нахождения компонента связности в ориентированном графе?

А) Запускать DFS для каждой непосещённой вершины и отмечать все достигаемые вершины +

Б) Запускать DFS только один раз для графа

В) Сортировать все вершины по весу

Г) Определять только максимальное расстояние

8. Каково значение стека в реализации поиска в глубину?

А) Стек не используется в этом алгоритме

Б) Стек хранит вершины, которые должны быть посещены +

В) Стек используется для хранения весов рёбер

Г) Стек используется для сортировки вершин

9. Какой из следующих методов может быть использован для визуализации процесса поиска в глубину?

А) Графическое представление каждого шага обхода +

Б) Использование случайных чисел

В) Построение матрицы смежности

Г) Применение жадного алгоритма

10. Как можно адаптировать поиск в глубину для нахождения всех путей между двумя вершинами?

А) Изменить условие завершения рекурсии, чтобы продолжать обход +

Б) Использовать только итеративный подход

В) Сортировать вершины по весу

Г) Применять алгоритм Дейкстры

Критерии оценивания тестовых заданий для текущего контроля

Максимальное количество баллов за решение одного теста составляет 5 баллов, которые переводятся в академические оценки следующим образом:

- оценка «отлично» (5 баллов) выставляется при условии правильного ответа студента не менее чем на 85% контрольных заданий

- оценка «хорошо» (4 балла) выставляется при условии правильного ответа студента не менее чем на 70% контрольных заданий

- оценка «удовлетворительно» (3 балла) выставляется при условии правильного ответа студента не менее чем на 51% контрольных заданий

- оценка «неудовлетворительно» (1-2 балла) выставляется при условии правильного ответа студента менее чем на 50% контрольных заданий

Тесть №18

Тема 18. Алгоритм нахождения сильно связных компонент.

1. Что такое сильно связная компонента в ориентированном графе?

А) Подмножество вершин, в котором каждая вершина соединена с каждой другой

Б) Подмножество вершин, в котором для каждой пары вершин существует путь в обоих направлениях +

В) Подмножество вершин, не имеющее рёбер

Г) Подмножество вершин, имеющее максимальную степень

2. Какой алгоритм часто используется для нахождения сильно связных компонент?

А) Алгоритм Дейкстры

Б) Алгоритм Краскала

В) Алгоритм Косарайю +

Г) Алгоритм Беллмана-Форда

3. Какова основная идея алгоритма Косарайю?

А) Два прохода по графу: первый для записи порядка посещения вершин, второй для обхода по транспонированному графу +

Б) Один проход по графу с использованием жадного подхода

В) Использование случайных чисел для обхода

Г) Сортировка рёбер по весу

4. Какой из следующих шагов выполняется в первом проходе алгоритма Косарайю?

А) Запись вершин в стек в порядке завершения обхода +

Б) Посещение всех рёбер графа

В) Сортировка вершин по весу

Г) Построение матрицы смежности

5. Каковы временные затраты на выполнение алгоритма Косарайю?

А) O(V)

Б) O(E)

В) O(V + E) +

Г) O(V^2)

6. Какую роль играет транспонированный граф в алгоритме Косарайю?

А) Он не используется в алгоритме

Б) Он позволяет найти обратные пути для определения сильно связных компонент +

В) Он используется только для визуализации

Г) Он заменяет исходный граф

7. Как можно обозначить вершины, входящие в одну сильно связную компоненту?

А) Все вершины, которые могут быть достигнуты из одной вершины в обоих направлениях +

Б) Все вершины, имеющие одинаковую степень

В) Все вершины, соединенные прямыми рёбрами

Г) Все вершины, которые были посещены в одном проходе

8. Какой из следующих алгоритмов также может использоваться для нахождения сильно связных компонент?

А) Алгоритм Tarjan +

Б) Алгоритм Флойда-Уоршелла

В) Алгоритм Краскала

Г) Алгоритм Дейкстры

9. Каково значение списка смежности в алгоритме нахождения сильно связных компонент?

А) Для хранения весов рёбер

Б) Для представления графа и упрощения обхода вершин +

В) Для хранения всех рёбер графа

Г) Для определения степени вершин

10. Как можно визуализировать сильно связные компоненты графа?

А) Использовать цветовую кодировку для обозначения разных компонент +

Б) Построить матрицу смежности

В) Применить случайные числа

Г) Изменить порядок рёбер в графе

Критерии оценивания тестовых заданий для текущего контроля

Максимальное количество баллов за решение одного теста составляет 5 баллов, которые переводятся в академические оценки следующим образом:

- оценка «отлично» (5 баллов) выставляется при условии правильного ответа студента не менее чем на 85% контрольных заданий

- оценка «хорошо» (4 балла) выставляется при условии правильного ответа студента не менее чем на 70% контрольных заданий

- оценка «удовлетворительно» (3 балла) выставляется при условии правильного ответа студента не менее чем на 51% контрольных заданий

- оценка «неудовлетворительно» (1-2 балла) выставляется при условии

правильного ответа студента менее чем на 50% контрольных заданий

Тест№19

Тема 19. Представление неориентированных графов. Основные деревья минимальной стоимости.

1. Какое из следующих представлений неориентированного графа наиболее распространено?

А) Список смежности +

Б) Список рёбер

В) Матрица весов

Г) Графическая схема

2. Каковы основные преимущества использования списка смежности для представления графа?

А) Легче находить все рёбра

Б) Экономия памяти, особенно для разреженных графов +

В) Простота реализации

Г) Удобство для поиска кратчайших путей

3. Какое из следующих представлений графа позволяет быстро находить вес рёбер между двумя вершинами?

А) Список смежности

Б) Матрица смежности +

В) Список рёбер

Г) Словарь рёбер

4. Что такое основное дерево в неориентированном графе?

А) Подмножество рёбер, которое соединяет все вершины графа без циклов +

Б) Подмножество рёбер, которое минимизирует количество рёбер

В) Подмножество вершин, соединённых между собой

Г) Подмножество рёбер, имеющее максимальную стоимость

5. Какой алгоритм чаще всего используется для нахождения минимального основного дерева?

А) Алгоритм Дейкстры

Б) Алгоритм Краскала +

В) Алгоритм Флойда-Уоршелла

Г) Алгоритм Беллмана-Форда

6. Какова основная идея алгоритма Краскала?

А) Сначала сортировать рёбра по весу и добавлять их в основное дерево, избегая циклов +

Б) Начинать с произвольной вершины и добавлять все её рёбра

В) Использовать жадный подход для выбора рёбер

Г) Обходить граф в ширину

7. Какой из следующих шагов выполняется в алгоритме Краскала?

А) Объединение вершин в компоненты с помощью структуры данных "система непересекающихся множеств" +

Б) Сортировка вершин по весу

В) Посещение всех рёбер графа

Г) Использование рекурсивного обхода

8. Какова временная сложность алгоритма Краскала?

А) O(V)

Б) O(E)

В) O(E log E) (или O(E log V)) +

Г) O(V^2)

9. Какой из следующих алгоритмов также может быть использован для нахождения минимального основного дерева?

А) Алгоритм Дейкстры

Б) Алгоритм Прима +

В) Алгоритм Флойда-Уоршелла

Г) Алгоритм Краскала

10. Как можно визуализировать минимальное основное дерево?

А) Выделить рёбра, входящие в основное дерево, другим цветом +

Б) Построить матрицу смежности

В) Использовать случайные числа

Г) Изменить порядок рёбер в графе

Критерии оценивания тестовых заданий для текущего контроля

Максимальное количество баллов за решение одного теста составляет 5 баллов, которые переводятся в академические оценки следующим образом:

- оценка «отлично» (5 баллов) выставляется при условии правильного ответа студента не менее чем на 85% контрольных заданий

- оценка «хорошо» (4 балла) выставляется при условии правильного ответа студента не менее чем на 70% контрольных заданий

- оценка «удовлетворительно» (3 балла) выставляется при условии правильного ответа студента не менее чем на 51% контрольных заданий

- оценка «неудовлетворительно» (1-2 балла) выставляется при условии правильного ответа студента менее чем на 50% контрольных заданий

Тест №20

Тема 20. Алгоритм Прима. Алгоритм Крускала.

1. Какова основная цель алгоритма Прима?

А) Найти кратчайший путь между двумя вершинами

Б) Найти минимальное основное дерево в неориентированном графе +

В) Упорядочить рёбра по весу

Г) Найти все сильно связные компоненты

2. Какова основная идея алгоритма Прима?

А) Начать с произвольной вершины и добавлять к основному дереву наименьшее по весу ребро, соединяющее уже добавленные вершины с новыми +

Б) Сначала отсортировать все рёбра по весу и выбирать минимальные

В) Использовать рекурсивный обход графа

Г) Объединять компоненты с помощью системы непересекающихся

множеств

3. Какой метод используется для выбора минимального ребра в алгоритме Прима?

А) Приоритетная очередь +

Б) Список смежности

В) Сортировка рёбер

Г) Стек

4. Какова временная сложность алгоритма Прима при использовании приоритетной очереди?

А) O(V)

Б) O(E)

В) O(E log V) +

Г) O(V^2)

5. Какова основная идея алгоритма Крускала?

А) Начинать с произвольной вершины и добавлять все её рёбра

Б) Сначала сортировать все рёбра по весу и добавлять их в основное дерево, избегая циклов +

В) Использовать жадный подход для выбора рёбер

Г) Обходить граф в ширину

6. Какой шаг следует выполнить в алгоритме Крускала перед добавлением ребра в основное дерево?

А) Проверить, соединены ли вершины

Б) Проверить, не образуется ли цикл с помощью структуры данных "система непересекающихся множеств" +

В) Изменить порядок рёбер

Г) Сортировать вершины по весу

7. Какова временная сложность алгоритма Крускала?

А) O(E log E) (или O(E log V)) +

Б) O(V)

В) O(E)

Г) O(V^2)

8. Какой из следующих алгоритмов можно использовать для реализации проверки циклов в алгоритме Крускала?

А) Алгоритм Дейкстры

Б) Алгоритм объединения с ранговым методом +

В) Алгоритм Прима

Г) Алгоритм Флойда-Уоршелла

9. Какой из следующих графов лучше всего подходит для алгоритма Прима?

А) Разреженные графы

Б) Связные графы с положительными весами +

В) Графы с отрицательными весами

Г) Деревья

10. Как можно визуализировать минимальное основное дерево, найденное с помощью алгоритмов Прима или Крускала?

А) Выделить рёбра, входящие в основное дерево, другим цветом +

Б) Построить матрицу смежности

В) Изменить порядок рёбер в графе

Г) Использовать случайные числа

Критерии оценивания тестовых заданий для текущего контроля

Максимальное количество баллов за решение одного теста составляет 5 баллов, которые переводятся в академические оценки следующим образом:

- оценка «отлично» (5 баллов) выставляется при условии правильного ответа студента не менее чем на 85% контрольных заданий

- оценка «хорошо» (4 балла) выставляется при условии правильного ответа студента не менее чем на 70% контрольных заданий

- оценка «удовлетворительно» (3 балла) выставляется при условии правильного ответа студента не менее чем на 51% контрольных заданий

- оценка «неудовлетворительно» (1-2 балла) выставляется при условии правильного ответа студента менее чем на 50% контрольных заданий

Тест №21

Тема 21. Контейнерный класс vector.

1. Что такое контейнерный класс vector в C++?

А) Динамический массив +

Б) Статический массив

В) Связный список

Г) Массив фиксированного размера

2. Какое из следующих утверждений верно для класса vector?

А) Он может изменять размер только в процессе инициализации

Б) Он автоматически управляет памятью и может изменять свой размер во время выполнения +

В) Его размер фиксирован и не может изменяться

Г) Он не поддерживает случайный доступ к элементам

3. Какой метод используется для добавления элемента в конец вектора?

А) insert()

Б) pushback() +

В) add()

Г) append()

4. Какова временная сложность метода pushback() в векторе?

А) O(1) в любом случае

Б) O(1) амортизированная +

В) O(N)

Г) O(log N)

5. Какой метод используется для удаления последнего элемента из вектора?

А) popback() +

Б) remove()

В) erase()

Г) delete()

6. Как можно получить доступ к элементу вектора по индексу?

А) Используя оператор [] или метод at() +

Б) Используя метод get()

В) Используя метод access()

Г) Используя метод retrieve()

7. Какой метод возвращает текущее количество элементов в векторе?

А) size()

Б) length()

В) count()

Г) size() +

8. Какой метод можно использовать для удаления элемента по индексу?

А) remove()

Б) erase() +

В) delete()

Г) pop()

9. Как можно очистить все элементы вектора?

А) clear() +

Б) reset()

В) delete()

Г) removeall()

10. Какой из следующих методов можно использовать для получения первого элемента вектора?

А) back()

Б) front() +

В) first()

Г) head()

Критерии оценивания тестовых заданий для текущего контроля

Максимальное количество баллов за решение одного теста составляет 5 баллов, которые переводятся в академические оценки следующим образом:

- оценка «отлично» (5 баллов) выставляется при условии правильного ответа студента не менее чем на 85% контрольных заданий

- оценка «хорошо» (4 балла) выставляется при условии правильного ответа студента не менее чем на 70% контрольных заданий

- оценка «удовлетворительно» (3 балла) выставляется при условии правильного ответа студента не менее чем на 51% контрольных заданий

- оценка «неудовлетворительно» (1-2 балла) выставляется при условии правильного ответа студента менее чем на 50% контрольных заданий

Тест№22

Тема 22. Контейнерный класс list.

1. Что такое контейнерный класс list в C++?

А) Динамический массив

Б) Двусвязный список +

В) Статический массив

Г) Массив фиксированного размера

2. Какое из следующих утверждений верно для класса list?

А) Он поддерживает произвольный доступ к элементам

Б) Его размер фиксирован и не может изменяться

В) Он позволяет эффективно вставлять и удалять элементы в любой части списка +

Г) Он не поддерживает итераторы

3. Какой метод используется для добавления элемента в начало списка?

А) pushfront() +

Б) insert()

В) addfirst()

Г) pushback()

4. Какой метод используется для добавления элемента в конец списка?

А) insert()

Б) pushback() +

В) addlast()

Г) append()

5. Какой метод используется для удаления первого элемента из списка?

А) remove()

Б) popfront() +

В) erase()

Г) deletefirst()

6. Какой метод используется для удаления последнего элемента из списка?

А) popback() +

Б) remove()

В) erase()

Г) deletelast()

7. Как можно получить доступ к первому элементу списка?

А) front() +

Б) first()

В) head()

Г) getfirst()

8. Какой метод возвращает текущее количество элементов в списке?

А) length()

Б) count()

В) size() +

Г) total()

9. Какой метод можно использовать для вставки элемента в список на определенную позицию?

А) insert()

Б) insert() с итератором +

В) addat()

Г) place()

10. Как можно очистить все элементы списка?

А) clear() +

Б) reset()

В) deleteall()

Г) remove_all()

Критерии оценивания тестовых заданий для текущего контроля

Максимальное количество баллов за решение одного теста составляет 5 баллов, которые переводятся в академические оценки следующим образом:

- оценка «отлично» (5 баллов) выставляется при условии правильного ответа студента не менее чем на 85% контрольных заданий

- оценка «хорошо» (4 балла) выставляется при условии правильного ответа студента не менее чем на 70% контрольных заданий

- оценка «удовлетворительно» (3 балла) выставляется при условии правильного ответа студента не менее чем на 51% контрольных заданий

- оценка «неудовлетворительно» (1-2 балла) выставляется при условии правильного ответа студента менее чем на 50% контрольных заданий

Тест№23

Тема 23. Ассоциативные контейнеры set и map.

1. Что такое контейнер set в C++?

А) Ассоциативный контейнер, который хранит уникальные элементы +

Б) Контейнер, допускающий дубликаты

В) Динамический массив

Г) Связный список

2. Какое из следующих утверждений верно для контейнера set?

А) Элементы могут быть изменены после вставки

Б) Элементы хранятся в отсортированном порядке +

В) Он поддерживает произвольный доступ к элементам

Г) Он может хранить элементы произвольного типа без дополнительных ограничений

3. Какой метод используется для добавления элемента в set?

А) insert()

Б) insert() +

В) add()

Г) push_back()

4. Как можно проверить, содержит ли set определённый элемент?

А) count() +

Б) find()

В) exists()

Г) contains()

5. Какой метод используется для удаления элемента из set?

А) remove()

Б) erase() +

В) delete()

Г) pop()

6. Что такое контейнер map в C++?

А) Ассоциативный контейнер, который хранит пары "ключ-значение" +

Б) Контейнер, допускающий дубликаты

В) Динамический массив

Г) Связный список

7. Какой метод используется для добавления пары "ключ-значение" в map?

А) insert()

Б) insert() или оператор [] +

В) add()

Г) put()

8. Как можно получить значение по ключу в map?

А) Используя оператор [] или метод at() +

Б) get()

В) retrieve()

Г) access()

9. Какой метод возвращает количество элементов в map?

А) length()

Б) size() +

В) count()

Г) total()

10. Какой метод можно использовать для удаления пары "ключ-значение" из map?

А) remove()

Б) erase() +

В) delete()

Г) pop()

Критерии оценивания тестовых заданий для текущего контроля

Максимальное количество баллов за решение одного теста составляет 5 баллов, которые переводятся в академические оценки следующим образом:

- оценка «отлично» (5 баллов) выставляется при условии правильного ответа студента не менее чем на 85% контрольных заданий

- оценка «хорошо» (4 балла) выставляется при условии правильного ответа студента не менее чем на 70% контрольных заданий

- оценка «удовлетворительно» (3 балла) выставляется при условии правильного ответа студента не менее чем на 51% контрольных заданий

- оценка «неудовлетворительно» (1-2 балла) выставляется при условии правильного ответа студента менее чем на 50% контрольных заданий

Тест№24

Тема 24. Общая схема компиляции.

1. Что такое компиляция в контексте программирования?

А) Процесс преобразования исходного кода в исполняемый код +

Б) Процесс выполнения программы
В) Процесс отладки кода
Г) Процесс написания документации
2. Какова первая стадия компиляции?
А) Компоновка
Б) Препроцессинг +
В) Лексический анализ
Г) Синтаксический анализ
3. Что происходит на стадии лексического анализа?
А) Исходный код разбивается на токены +
Б) Проверяется синтаксис программы
В) Определяются типы данных
Г) Генерируется машинный код
4. Какова роль синтаксического анализа в компиляции?
А) Генерация исполняемого кода
Б) Проверка структуры программы и построение синтаксического дерева +
В) Оптимизация кода
Г) Сборка программы
5. Какой этап следует за синтаксическим анализом?
А) Семантический анализ +
Б) Генерация кода
В) Компоновка
Г) Оптимизация
6. Что включает в себя семантический анализ?
А) Проверка на наличие синтаксических ошибок
Б) Проверка типов и логики программы +
В) Оптимизация кода
Г) Генерация промежуточного кода
7. Какова цель оптимизации кода?
А) Упрощение структуры программы
Б) Улучшение производительности и уменьшение объема кода +
В) Устранение ошибок
Г) Генерация исполняемого файла
8. На каком этапе происходит генерация машинного кода?
А) Препроцессинг
Б) Лексический анализ
В) Синтаксический анализ
Г) Генерация кода +
9. Что такое компоновка в процессе компиляции?
А) Объединение различных объектных файлов в единый исполняемый файл +
Б) Процесс создания промежуточного кода
В) Оптимизация кода
Г) Проверка типов данных

10. Какой результат компиляции в конечном итоге получают разработчики?

А) Исходный код

Б) Исполняемый файл +

В) Промежуточный код

Г) Документация

Критерии оценивания тестовых заданий для текущего контроля

Максимальное количество баллов за решение одного теста составляет 5 баллов, которые переводятся в академические оценки следующим образом:

- оценка «отлично» (5 баллов) выставляется при условии правильного ответа студента не менее чем на 85% контрольных заданий

- оценка «хорошо» (4 балла) выставляется при условии правильного ответа студента не менее чем на 70% контрольных заданий

- оценка «удовлетворительно» (3 балла) выставляется при условии правильного ответа студента не менее чем на 51% контрольных заданий

- оценка «неудовлетворительно» (1-2 балла) выставляется при условии правильного ответа студента менее чем на 50% контрольных заданий

2.2 Оценочные материалы для проведения промежуточной аттестации

1. Что такое парадигма программирования?
А) Метод проектирования интерфейсов пользователя +
Б) Набор концепций и подходов к разработке программного обеспечения
В) Язык программирования высокого уровня
Г) Библиотека стандартных функций

2. Как называется парадигма, основанная на описании действий, необходимых для решения задачи?
А) Объектно-ориентированное программирование
Б) Логическое программирование
В) Функциональное программирование
Г) Императивное программирование +

3. Что является основной единицей объектно-ориентированного программирования?
А) Модуль
Б) Функция
В) Класс и объект +
Г) Переменная

4. Какая парадигма позволяет избежать побочных эффектов путем запрета изменения переменных внутри программы?
А) Логическое программирование

Б) Структурное программирование
В) Объявление типов
Г) Функциональное программирование +

5. Какой принцип ООП обеспечивает сокрытие деталей реализации класса?
А) Полиморфизм
Б) Инкапсуляция +
В) Наследование
Г) Абстракция

6. Какие языки программирования поддерживают функциональную парадигму?
А) Python, JavaScript, Ruby
Б) C++, Java, PHP
В) Haskell, Lisp, Erlang +
Г) SQL, HTML, CSS

Вопрос №7. Какое понятие описывает повторное использование кода путем наследования свойств родительского класса?
А) Композиция
Б) Агрегация
В) Интерфейсы
Г) Наследование +

8. В каком стиле программирования используется декларативный подход?
А) Императивный
Б) Феноменологический
В) Процедурный
Г) Декларативный +

9. Что обозначают методы перегрузки операторов в языках программирования?
А) Возможность переопределять стандартные операторы для собственных классов +
Б) Автоматический выбор метода компилятором
В) Оптимизация производительности приложения
Г) Подмена стандартной операции умножения другим действием

10. К какому типу относятся модели программирования с фиксированными структурами управления потоком исполнения?
А) Гибридные модели
Б) Параллельные модели
В) Программные автоматы +
Г) Распределённые системы

11. Для какой парадигмы характерно выполнение инструкций последовательно одну за другой?
А) Алгоритмическое программирование
Б) Функциональное программирование
В) Императивное программирование +
Г) Эвент-драйвен программирование

12. Назначение полиморфизма заключается в следующем:
А) Предоставляет общий интерфейс нескольким классам разных иерархий
Б) Упрощение структуры кода путём повторного использования
В) Изменение поведения методов объектов в зависимости от типа аргументов +
Г) Уменьшение количества строк исходного кода

13. Какая концепция поддерживает создание модулей и библиотек для повторного использования?
А) Модуляризация +
Б) Рекурсия
В) Шаблоны проектирования
Г) Динамическая загрузка

14. Основная цель структурного программирования состоит в:
А) Минимизации ошибок путём разделения задачи на части +
Б) Создании оптимальных алгоритмов
В) Сокращении объёма памяти приложений
Г) Улучшении визуализации процесса разработки

15. Какие типы моделей параллельного программирования существуют?
А) Мониторинговая и семафорная
Б) Конвейерная и цикловые процессы
В) Многопоточная и распределённая +
Г) Императивная и декларативная

16. Чему соответствует абстрактный синтаксис в теории формальных языков?
А) Семантическим правилам интерпретации выражений
Б) Представлению грамматики вне конкретной формы записи выражения +
В) Форматированию текста программы
Г) Методу обработки лексического анализа

17. Что означает термин "интенсиональное определение"?
А) Определение по поведению или результатам работы программы
Б) Определённый порядок передачи сообщений между объектами
В) Отражение смысла конструкции через её структуру +
Г) Определяет назначение конкретного модуля или библиотеки

18. Основной характеристикой какого стиля программирования является разделение процессов на потоки и нити?
А) Параллельное программирование +
Б) Общее программное обеспечение
В) Эвристическое программирование
Г) Прототипное программирование

19. Когда применяют прототипное программирование?
А) Если необходимо минимизировать количество используемого кода
Б) Если нужен быстрый запуск минимально работающего продукта +
В) Если проект требует строгих стандартов качества
Г) Если важен детальный дизайн архитектуры

20. Что относится к особенностям логической парадигмы программирования?
А) Запись программ как набора утверждений и правил вывода +
Б) Использование рекурсивных функций
В) Ограниченность глобального состояния
Г) Применение функциональных конструкций высшего порядка

21. Основное отличие асинхронного программирования от синхронного:
А) Последовательность выполнения операций
Б) Время выполнения команд
В) Отсутствие блокировки потоков при ожидании ввода-вывода +
Г) Повышенная производительность за счёт многопоточности

22. Какова главная задача рефлексивного подхода в метапрограммировании?
А) Увеличение скорости выполнения программ
Б) Возможность изменять поведение программы динамически +
В) Уменьшение размера исполняемого файла
Г) Совершенствование дизайна классов

23. Как называют методику построения программ, ориентированную на максимальную адаптацию и изменение структуры кода в процессе выполнения?
А) Рефакторинг
Б) Микроархитектурные шаблоны
В) Динамический рефлективный подход +
Г) Скриптинг

24. Установите соотнесите между терминами из первого столбца с правильными определениями во втором столбце.

Термины:	Определения
1. Императивное программирование	А) Стиль программирования, основанный на функциях.

2. Объектно-ориентированное программирование	Б) Подход, в котором описывается, что должно быть сделано
3. Декларативное программирование	В) Метод, при котором используется инкапсуляция, наследование и полиморфизм
4. Функциональное программирование	Г) Стиль программирования, в котором акцент делается на изменение состояния программы

Ответ: 1-Г, 2-В, 3-Б, 4-А

25. Вставьте пропущенное слово в предложение.
В логическом программировании используются _____ выражения для вывода заключений и решения задач.
Ответ: логические

Критерии оценки для проведения зачета по дисциплине (тестирование)

Шкала оценивания результатов промежуточной аттестации и критерии выставления оценок.

Оценка	**Уровень сформированности компетенции**	**Критерии**
«Отлично» («зачтено»)	Высокий	Выполнено более 80% заданий предложенного теста
«Хорошо» («зачтено»)	Хороший	Выполнено 60-80% заданий предложенного теста
«Удовлетворительно» («зачтено»)	Достаточный	Выполнено 35-60% заданий предложенного теста
«Неудовлетворительно» («не зачтено»)	Недостаточный	Выполнено менее 35% заданий предложенного теста

Обновление оценочных материалов дисциплины

обсуждено и рекомендовано к утверждению решением кафедры наименование кафедры
от ___ _____ 20_ г., протокол № ___

одобрено Научно-методическим советом _____ от ___ _____ 20__ г., протокол №____