Подготовка к экзамену по программированию. Теоретическая часть:

Вопросы:

1. Понятие алгоритм. Формальные признаки. Способы задания алгоритма. Виды

алгоритмов. Теорема Дейкстра.

Понятие алгоритм - совокупность точно заданных правил решения некоторого класса задач или набор инструкций, описывающих порядок действий исполнителя для решения определённой задачи.

Формальные признаки –

Дискретность (то свойство алгоритма, когда алгоритм разбивается на конечное число элементарных действий (шагов));

Детерминированность (определённость) (свойство алгоритма, которое означает, что при каждом запуске алгоритма с одними и теми же исходными данными должен быть получен один и тот же результат);

Понятность ( свойство алгоритма, которое означает, что алгоритм содержит только те команды, которые входят в систему команд исполнителя, для которого он предназначен);

Завершаемость (конечность) (свойство, позволяющее алгоритму при корректно заданных исходных данных завершать свою работу за конечное число шагов);

Массовость (универсальность) (применимость алгоритма ко всем задачам рассматриваемого типа, при любых исходных данных.);

Результативность (любой алгоритм должен завершаться за конечное (может быть очень большое) число шагов);

способы задания алгоритма -

1) Словесный способ - отражает содержание выполняемых действий средствами естественного языка.

2) Формально-словесный способ - основан на записи содержания выполняемых действий с использованием изобразительных возможностей языка математики, дополненного с целью указания необходимых пояснений средствами естественного языка

3) Графический способ (в виде блок-схемы) - представляет собой изображение логико-математической структуры алгоритма, при котором все этапы процесса обработки данных представляются с помощью определенного набора геометрических фигур (блоков), имеющих строго определенную конфигурацию в соответствии с характером выполняемых действий

Виды алгоритмов -

1) Линейный алгоритм – алгоритм, в котором все этапы решения задачи выполняются в естественном порядке следования.

2) Циклический алгоритм – алгоритм, в котором присутствуют многократно повторяющиеся участки вычисления.

3) Разветвляющий алгоритм - алгоритм, в котором в зависимости от условия выполняется либо одна, либо другая последовательность действий.

4) Вспомогательный алгоритм - алгоритм, который можно использовать в других алгоритмах, указав только его имя

Теорема Дейкстра - Алгоритм находит кратчайшие пути от одной из вершин графа до всех остальных.

2. Понятие языка программирования. Компилируемые и интерпретируемые языки

программирования.

Понятие языка программирования - Формальный язык, предназначенный для записи компьютерных программ. Язык программирования определяет набор лексических, синтаксических и семантических правил, определяющих внешний вид программы и действия, которые выполнит исполнитель под её управлением.

Компилируемые и интерпретируемые языки программирования - Компилируемый язык — это такой язык, что программа, будучи скомпилированной, содержит инструкции целевой машины; этот машинный код непонятен людям. Интерпретируемый же язык — это такой, в котором инструкции не исполняются целевой машиной, а считываются и исполняются другой программой (которая обычно написана на языке целевой машины).

3. Классификация языков программирования. Языки низкого уровня. Языки высокого

уровня.

Классификация языков программирования -

Они делятся на несколько классификаций:

Высокоуровневые и низкоуровневые.

Компилируемые и интерпретируемые.

Объектно-ориентированные, функциональные, процедурные и декларативные.

Языки низкого уровня - это язык программирования, который практически не отличается от архитектуры набора команд компьютера. Языки низкого уровня обычно используются для написания системного программного обеспечения, такого как операционные системы и драйверы устройств, или для доступа к аппаратным устройствам.

Примерами языков программирования низкого уровня являются машинный код и язык ассемблера.

Языки высокого уровня - язык программирования, который обеспечивает описание задачи в наглядном, легко воспринимаемом виде, удобном для программиста. Языкам высокого уровня свойственно умение работать с комплексными структурами данных. В большинство из них интегрирована поддержка строковых типов, объектов, операций файлового ввода-вывода

4. Язык Си. Общие сведения. Элементы языка Си. Лексемы. Комментарии. Ключевые

слова. Идентификаторы.

Язык Си:

Общие сведения - Компилируемый статически типизированный язык программирования общего назначения, разработанный в 1969-1973 годах сотрудником Bell Labs Деннисом Ритчи как развитие языка Би. Первоначально был разработан для реализации операционной системы UNIX, но впоследствии был перенесён на множество других платформ. Согласно дизайну языка, его конструкции близко сопоставляются типичным машинным инструкциям, благодаря чему он нашёл применение в проектах, для которых был свойственен язык ассемблера, в том числе как в операционных системах, так и в различном прикладном программном обеспечении для множества устройств - от суперкомпьютеров до встраиваемых систем

Элементы языка Си:

* переменные и их типы данных
* операторы и выражения
* циклы и условные операторы
* функции и их вызовы
* массивы и указатели
* структуры и объединения
* файловый ввод-вывод
* директивы препроцессора
* операции ввода-вывода

Лексемы - минимальные значимые единицы текста в программе:

- идентификаторы;

- ключевые (зарезервированные) слова;

- знаки операций;

- константы;

- разделители (скобки, точка, запятая, пробельные символы).

Границы лексем определяются другими лексемами, такими, как разделители или знаки операций, а также комментариями

Комментарии - это последовательность символов, которая начинается с косой черты и звездочки либо двумя слешами (//). Компилятор воспринимает комментарий как один пробельный символ, в остальных случаях он игнорируется.

Ключевые слова - это зарезервированные слова, которые имеют специальное значение в языке программирования. Они используются для определения структуры программы, выполнения операций и управления потоком выполнения.

Идентификаторы - это имена, задаваемые в программе для переменных, типов, функций и меток

5. Элементы языка Си. Константы. Целочисленные константы. Константы с

плавающей запятой. Константы перечисления.

Элементы языка Си:

* переменные и их типы данных
* операторы и выражения
* циклы и условные операторы
* функции и их вызовы
* массивы и указатели
* структуры и объединения
* файловый ввод-вывод
* директивы препроцессора
* операции ввода-вывода

Константы - это величина (число или символ), которая при выполнении программы всегда остаётся неизменной, т. е. не может изменяться во время выполнения программы

Целочисленные константы - это постоянные элементы данных, которые не имеют дробных частей или экспонент

Константы с плавающей запятой - это десятичное число, которое представляет знаковое вещественное число.

Константы перечисления - это определенные значения, связанные с элементами перечислимого типа в программировании. Они используются для идентификации отдельных членов перечисления или перечисляемых типов без необходимости приведения типа или использования операторов сравнения.

6. Элементы языка Си. Символьные константы. Управляющая последовательность.

Строки. Разделители (символы пунктуации) и специальные символы.

Элементы языка Си:

* переменные и их типы данных
* операторы и выражения
* циклы и условные операторы
* функции и их вызовы
* массивы и указатели
* структуры и объединения
* файловый ввод-вывод
* директивы препроцессора
* операции ввода-вывода

Символьные константы - это некоторый символ алфавита, заключенный в одиночные кавычки.

Управляющая последовательность в C - это символы или последовательность символов, которые могут использоваться внутри строкового литерала. Назначение управляющей последовательности - представлять символы, которые нельзя использовать обычным образом с клавиатуры.

\r Возврат каретки в начало строки

\n Переход на новую строку

\t Горизонтальная табуляция

\v Вертикальная табуляция

\" Двойные кавычки

\' Апостроф (одинарные кавычки)

\\ Обратный слеш (косая черта)

\0 Нулевой символ (символ конца строки)

\? Знак вопроса

\a Сигнал зуммера (динамика) компьютера

Строки - тип данных, значениями которого является произвольная последовательность символов алфавита

Разделители (символы пунктуации) и специальные символы - Разделитель - это последовательность из одного или нескольких символов для указания границы между отдельными, независимыми областями в обычном тексте, математических выражениях или других потоках данных. Специальные символы - это символы, используемые для совершения каких-либо операция либо действий:

7. Типы данных. Фундаментальные типы. Массивы и указатели. Взаимосвязь

массивов и указателей.

Типы данных -

1) char: представляет один символ. Диапазон от -128 до 127. Занимает в памяти 1 байт

2) unsigned char: представляет один символ. Диапазон от 0 до 255. Занимает в памяти 1 байт

3) short: представляет целое число в диапазоне от –32768 до 32767. Занимает в памяти 2 байта

4) unsigned short: представляет целое число в диапазоне от 0 до 65535. Занимает в памяти 2 байта

5) int: представляет целое число. В зависимости от процессора, может занимать 2 либо 4 байта. Диапазон от –32768 до 32767 (при 2 байтах) или от −2 147 483 648 до 2 147 483 647 (при 4 байтах).

6) unsigned int: представляет положительное целое число. В зависимости от процессора может занимать 2 байта или 4 байта. от 0 до 65535 (для 2 байт), либо от 0 до 4 294 967 295 (для 4 байт).

7) long: представляет целое число в диапазоне от -2 147 483 648 до 2 147 483 647. Занимает в памяти 4 байта

8) unsigned long: представляет целое число в диапазоне от 0 до 4 294 967 295. Занимает в памяти 4 байта

9) long long: представляет целое число в диапазоне от -9223372036854775807 до +9 223 372 036 854 775 807. Занимает в памяти, как правило, 8 байт

10) unsigned long long: представляет целое число в диапазоне от 0 до 18 446 744 073 709 551 615. Занимает в памяти, как правило, 8 байт

11) float: представляет вещественное число одинарной точности с плавающей точкой в диапазоне +/- 3.4E-38 до 3.4E+38. В памяти занимает 4 байта

12) double: представляет вещественное число двойной точности с плавающей точкой в диапазоне +/- 1.7E-308 до 1.7E+308. В памяти занимает 8 байт

13) long double: представляет вещественное число двойной точности с плавающей точкой в диапазоне +/- 3.4E-4932 до 1.1E+4932. В памяти занимает 10 байт

Фундаментальные типы - Стандарт С89 определяет пять фундаментальных типов данных: char — символьные данные, int — целые, float — с плавающей точкой, double — двойной точности, void — без значения.

Массивы и указатели - Массив — это тип данных, в котором хранится упорядоченный набор однотипных элементов. Указатель — это переменная, в которой хранится адрес памяти объекта.

Взаимосвязь массивов и указателей - В языке Си массивы и указатели тесно связаны. С помощью указателей мы также легко можем манипулировать элементами массива, как и с помощью индексов. Имя массива без индексов в Си является адресом его первого элемента. Соответственно через операцию разыменования мы можем получить значение по этому адресу

8. Типы данных. Производные типы. Структуры.

Типы данных - из 7 вопроса

Производные типы данных - это те типы данных, которые создаются путем объединения примитивных типов данных и других производных типов данных

Производные типы - Производные типы - Это типы данных, которые не зависят от реализации и могут быть реализованы тем или иным способом. Они обычно создаются с помощью комбинации основных или встроенных типов данных и связанных с ними операций.

Например:

1) список

2) массив

3) стек

4) очередь

Структуры - это способ организации информации для более эффективного использования. В программировании структурой обычно называют набор данных, связанных определённым образом.

9. Типы данных. Производные типы. Объединения.

Типы данных - из 7 вопроса

Производные типы - Это типы данных, которые не зависят от реализации и могут быть реализованы тем или иным способом. Они обычно создаются с помощью комбинации основных или встроенных типов данных и связанных с ними операций.

Например:

1) список

2) массив

3) стек

4) очередь

Объединения - это механизм, который позволяет хранить данные разных типов в одном и том же месте памяти. Это может быть полезно, например, для работы с структурами данных, которые могут содержать данные разных типов.

10. Типы данных. Производные типы. Перечисления. Объявление нового имени типа.

Типы данных - из 7 вопроса

Производные типы - Это типы данных, которые не зависят от реализации и могут быть реализованы тем или иным способом. Они обычно создаются с помощью комбинации основных или встроенных типов данных и связанных с ними операций.

Например:

1) список

2) массив

3) стек

4) очередь

**Перечисления представляют типы данных, которые содержат набор констант, и каждой константе сопоставлено определенное числовое значение. Перечисление определяется с помощью ключевого слова enum**

Перечисления - это способ определения нового типа данных, который может принимать только определенные значения. Например, можно определить тип данных, который может быть равен только ‘red’, ‘green’ или ‘blue’.

Объявление нового имени типа - Объявление нового типа имени в C означает создание нового типа данных с новым именем. Это может быть сделано с помощью ключевого слова typedef или с использованием struct или union.

11. Арифметические операции. Унарные операции. Операции сравнения. Логические

операции. Битовые операции.

Арифметические операции - Всего существует 4 вида арифметических операций, которые вам прекрасно знакомы со школы: сложение (+), вычитание (-), умножение (\*) и деление (/).

Унарные операции - В языке Си имеются следующие унарные операции: - арифметическое отрицание (отрицание и дополнение); ~ побитовое логическое отрицание (дополнение); ! логическое отрицание; \* разадресация (косвенная адресация); & вычисление адреса; + унарный плюс; ++ увеличение (инкремент); -- уменьшение (декремент); sizeof размер.

Операции сравнения - В языке Си используются следующие операции сравнения, т.е. отношения между объектами:==– равно или эквивалентно; != – не равно; <– меньше; <=– меньше либо равно; >– больше; >=– больше либо равно. Пары символов соответствующих операций разделять нельзя.

Логические операции - Три основные логические операции в языке Си: ! операция отрицания (НЕ), && конъюнкция, логическое умножение (И), | | дизъюнкция, логическое сложение (или).

|  |  |
| --- | --- |
| && | Логический оператор И создает значение 1, если оба операнда имеют ненулевое значение. Если любой из операндов имеет значение 0, результат равен 0. Если первый операнд операции логического И имеет значение 0, то второй операнд не вычисляется. |
| **||** | Оператор логического ИЛИ выполняет над своими операндами операцию включающего ИЛИ. Если оба операнда имеют значение 0, результат будет равен 0. Если любой из операндов имеет ненулевое значение, результат будет равен 1. Если первый операнд операции логического ИЛИ имеет ненулевое значение, то второй операнд не вычисляется. |

Битовые операции - В Си существуют 6 операторов для манипулирования битами:<< — сдвиг влево>> — сдвиг вправо~ — поразрядная инверсия| — поразрядное или& — поразрядное И^ — поразрядное исключающее или. Если нужно записать сразу несколько бит: unsigned char x = 0; x = 1<<номер\_бита | 1<<номер\_бита | 1<<номер\_бита ...; Номера битов отсчитываются справа налево, начиная с нуля. 0 — это младший бит (справа), 7 — это старший бит (слева).

12. Условная операция. Последовательность. Операции присваивания. Операция

получения размера объекта. Приоритеты операций. Расстановка приоритетов.

Условная операция - это конструкция, которая позволяет выполнять различные блоки кода в зависимости от условия.

Последовательность - это способ организации кода, при котором операторы выполняются один за другим сверху вниз. Последовательность является основным способом организации кода в Си, и все операторы, функции и блоки кода располагаются в определенном порядке.

Операции присваивания - Операция присваивания в языке Си служит для присвоения значения переменной или выражения. Она имеет вид: переменная = выражение; где переменная - это идентификатор переменной, а выражение - некоторое арифметическое, логическое или строковое выражение, значение которого присваивается переменной. После выполнения операции присваивания значение переменной становится равным значению выражения, и это значение можно использовать в дальнейшем коде.

Операция получения размера объекта - Операция получения размера объекта (sizeof operator) в Си служит для определения размера (в байтах) заданного типа данных или объекта.

Приоритеты операций - Приоритеты операций в Си определяют порядок выполнения операций в выражении. В Си есть шесть уровней приоритетов операций:

1) Постфиксные операторы инкремента (++) и декремента (–).

2) Унарное отрицание (-).

3) Бинарные операции \* / % и ++ –

4) Бинарные операции + - и << >> .

5) Бинарное умножение (.\*) и деление (/).

6) Бинарное сложение (+).

Расстановка приоритетов - В языке программирования C расстановка приоритетов операторов определяется их позициями в таблице приоритетов, представленной ниже (от высшего приоритета к низшему) Если в выражении используются несколько операторов одного приоритета, они выполняются слева направо. Использование скобок позволяет изменять порядок выполнения операторов. Внутри скобок операторы выполняются в обычном порядке, слева направо.

13. Понятие оператора. Пустой оператор. Операторы объявления имен. Операторы выражения. Функциональные блоки.

Понятие оператора - это конструкция, определяющая действие, которое нужно выполнить. Операторы могут быть арифметическими (например, +, -, \*), логическими (например, &&, ||), операторами присваивания (=, +=) и другими

Пустой оператор - Пустой оператор (;) в языке программирования Си используется для разделения двух независимых операторов в одной строке. Он не выполняет никаких действий, но является обязательным для того, чтобы компилятор мог различить два отдельных оператора.

Операторы объявления имен - Операторы объявления имени (typedef, enum, struct, union и typedef) в языке программирования Си используются для создания пользовательских типов данных и объявления переменных этих типов.

Операторы выражения - это ключевые слова, которые используются для выполнения операций над операндами (переменными, значениями) для получения результата. В C есть следующие операторы выражения:

1)Арифметические операторы: + (сложение), - (вычитание), \* (умножение), / (деление), % (остаток от деления).

2)Операторы сравнения: == (равно), != (не равно), < (меньше), > (больше), <= (меньше или равно), >= (больше или равно).

3)Логические операторы: && (логическое И), || (логическое ИЛИ), ! (логическое НЕ).

4)Битовые операторы: & (побитовое И), | (побитовое ИЛИ), ^ (побитовое исключающее ИЛИ), ~ (побитовое НЕ), << (сдвиг влево), >> (сдвиг вправо).

Функциональные блоки - это блоки кода, которые выполняют определенную функцию или операцию. Они могут быть созданы с использованием функций, структур, объединений и других элементов языка программирования C. Функциональные блоки могут быть использованы для организации кода, уменьшения его сложности и улучшения читаемости программы.

14. Оператор условия. Оператор переключателя.

Оператор условия - это конструкция, которая позволяет проверить условие и выполнить соответствующий блок кода.

Оператор переключателя - Оператор switch в C позволяет выбрать один из нескольких путей выполнения кода в зависимости от значения определенного выражения. Он работает путем сравнения значения выражения с каждым значением в списке case, и выполнение продолжается с соответствующим блоком кода case. Если ни одно из значений не совпадает, выполняется блок кода default, если он присутствует.

15. Операторы цикла.

Операторы цикла - Операторы цикла в C используются для повторения блоков кода до тех пор, пока не будет выполнено определенное условие. В C существуют три основных типа операторов цикла: for, while и do-while. Оператор for используется для выполнения блока кода определенное количество раз или пока не будет достигнуто указанное условие. Оператор while используется для выполнения блока кода до тех пор, пока выполняется указанное условие. Оператор do-while выполняет блок кода хотя бы один раз, а затем проверяет условие.

16. Операторы перехода.

Операторы перехода - Операторы перехода в C используются для перехода к определенной метке или адресу в коде. В C есть два основных оператора перехода: goto и return. Оператор goto позволяет перейти к любой метке в коде, в то время как оператор return используется для возврата из функции и передачи управления обратно вызывающей функции.

17. Понятие функции. Реализация функции. Объявление функции. Вызов функции.

Рекурсивный вызов.

Понятие функции - это блоки кода, которые могут быть вызваны из разных частей программы. Они позволяют разбивать программу на более мелкие и понятные части, что облегчает чтение, понимание и изменение кода. Функции могут принимать аргументы (параметры) и возвращать значения. Это позволяет передавать информацию между различными частями программы и использовать функции для выполнения различных задач.

Реализация функции - это тело функции, то есть код, который выполняется, когда функция вызывается. Она состоит из объявлений и определений функций, а также кода, который выполняет задачу, указанную в объявлении функции.

Объявление функции - это указание того, что функция существует и что она делает. Оно включает в себя имя функции, список параметров и возвращаемый тип. Объявление функции может быть использовано для ссылки на функцию из других частей программы, но не определяет ее реализацию.

Вызов функции - это оператор, который выполняет функцию. Он состоит из имени функции и списка аргументов, которые передаются в функцию. Вызов функции может быть выполнен в любом месте программы, где разрешено использование функции.

Рекурсивный вызов - это вызов функции из самой себя. Он используется, когда функцию необходимо вызвать несколько раз с разными аргументами, и позволяет упростить код, уменьшая количество повторяющихся блоков.

18. Указатель на функцию.

Указатель на функцию - это переменная, которая хранит адрес функции в памяти. Он может быть использован для вызова функции по адресу, передан в другую функцию в качестве аргумента или возвращен как результат другой функции. Указатели на функции могут быть определены как переменные или как параметры функций.

19. Файлы. Файловая система. Свойства файла (имя, расширение, атрибуты, время,

владелец и группа, права доступа).

Файлы - это способ хранения данных на компьютере. Они используются для чтения и записи информации на диск. Файлы могут быть открыты, прочитаны, записаны и закрыты. В C есть различные функции для работы с файлами, такие как fopen, fread, fwrite, fclose и другие.

Файловая система - это структура, которая организует файлы на диске. Она определяет, где хранятся файлы, как они называются и как к ним можно получить доступ. В C файловая система представлена библиотекой stdio.h.

Свойства файла (имя, расширение, атрибуты, время, владелец и группа, права доступа) - Свойства файла - это характеристики, которые описывают файл и предоставляют информацию о нем.

1) имя - это уникальный идентификатор файла. Оно состоит из собственно имени файла и расширения, которое указывает на тип файла.

2) расширение - это часть имени файла после точки. Оно используется для определения типа файла и программы, которая может открыть его.

3) атрибуты - это свойства файла, которые определяют, как он будет обрабатываться операционной системой. Например, атрибуты могут указывать, является ли файл скрытым, системным или только для чтения.

4) время - это дата и время создания или последнего изменения файла.

5) владелец и группа - то идентификаторы пользователя и группы, которым принадлежит файл.

6) права доступа - это набор разрешений, определяющих, какие операции могут быть выполнены с файлом.

20. Файлы. Операции с файлами. Типы файлов.

Файлы - это способ хранения данных на компьютере. Они используются для чтения и записи информации на диск. Файлы могут быть открыты, прочитаны, записаны и закрыты. В C есть различные функции для работы с файлами, такие как fopen, fread, fwrite, fclose и другие.

Операции с файлами - В C есть несколько функций для работы с файлами. Вот некоторые из них:

– fopen - открывает файл для чтения или записи.

– fclose - закрывает открытый файл.

– fread - читает данные из файла.

– fwrite - записывает данные в файл.

– feof - проверяет, достигнут ли конец файла.

Типы файлов - В C нет понятия типов файлов. Все файлы обрабатываются как последовательности байтов. Однако, в некоторых случаях, файлы могут иметь определенный формат, который может требовать специальной обработки. Например, текстовые файлы могут обрабатываться по-разному, чем двоичные файлы.

21. Файлы. Функции для работы с файлами. Операции, связанные с открытием файла.

Дескриптора файла типа int, дескриптор файла типа FILE (FILE\*),

файла типа HANDLE

Файлы - это способ хранения данных на компьютере. Они используются для чтения и записи информации на диск. Файлы могут быть открыты, прочитаны, записаны и закрыты. В C есть различные функции для работы с файлами, такие как fopen, fread, fwrite, fclose и другие.

Функции для работы с файлами - В C есть функции для работы с файлами, которые позволяют открывать, читать, записывать и закрывать файлы. Некоторые из этих функций включают fopen, fread, fwrite и fclose.

Операции, связанные с открытием файла - В языке программирования C, операции, связанные с открытием файла, включают в себя использование функций fopen(), fclose() и некоторых других. Функция fopen() используется для открытия файла, а функция fclose() - для его закрытия. Кроме того, существуют функции для чтения из файла (например, fread()) и записи в файл (например, fwrite()).

Дескриптора файла типа int, дескриптор файла типа FILE (FILE\*), дескриптор файла типа HANDLE - В C, дескрипторы файлов обычно имеют тип int. Однако, в некоторых случаях, могут использоваться другие типы дескрипторов, такие как FILE\* или HANDLE.

Тип FILE\* используется для работы с потоками ввода-вывода в стандартной библиотеке C. Этот тип дескриптора позволяет выполнять различные операции с файлами, такие как чтение, запись и закрытие.

Тип HANDLE используется в Win32 API для работы с дескрипторами объектов. Этот тип может быть использован для работы с файлами на Windows системах.

СЛЕДУЮЩИЕ ВОПРОСЫ НЕ 100% НУЖНЫ!!!!

22. Файлы. Функции для работы с файлами. Операции, не связанные с открытием файла (удаление, копирование, перенос/переименование). Функции для работы с каталогами (создание, удаление, получение списка файлов).

Файлы - это способ хранения данных на компьютере. Они используются для чтения и записи информации на диск. Файлы могут быть открыты, прочитаны, записаны и закрыты. В C есть различные функции для работы с файлами, такие как fopen, fread, fwrite, fclose и другие.

Функции для работы с файлами - В C есть функции для работы с файлами, которые позволяют открывать, читать, записывать и закрывать файлы. Некоторые из этих функций включают fopen, fread, fwrite и fclose.

Операции, не связанные с открытием файла (удаление, копирование, перенос/переименование) - В языке программирования C нет встроенных функций для удаления, копирования или перемещения файлов. Однако, эти операции можно выполнить с помощью системных вызовов, таких как system() или fork().

Функции для работы с каталогами (создание, удаление, получение списка файлов) - В языке программирования C отсутствуют встроенные функции для работы с каталогами. Однако можно использовать системные вызовы, такие как “opendir”, “readdir” и “closedir”.

23. Структуры данных. Понятие Структуры данных. Линейные структуры данных

(массив, список, очередь).

Структуры данных:

Понятие Структуры данных - Структуры в C позволяют группировать различные данные вместе. Это может быть полезно, если у вас есть несколько связанных переменных, которые нужно использовать вместе. Например, если вы создаете программу для управления базой данных, вы можете создать структуру для каждого элемента в вашей базе данных. Это позволит вам хранить всю информацию об элементе вместе, вместо того чтобы хранить каждую переменную отдельно.

Линейные структуры данных (массив, список, очередь) - В C существует несколько типов линейных структур данных: массивы, списки и очереди.

Массивы - это последовательность элементов одного типа, расположенных в памяти рядом друг с другом. С массивами можно работать как с единым целым, выполняя операции над всеми элементами сразу.

Списки - это структуры данных, которые позволяют хранить данные в виде последовательности элементов. Каждый элемент списка содержит информацию о следующем элементе, что позволяет осуществлять навигацию по списку.

Очереди - это структуры данных, которые используются для хранения данных в порядке поступления. Они обычно используются для обработки задач в порядке их поступления.

24. Структуры данных. Понятие Структуры данных. Линейные структуры данных

(стек, дек, хеш-таблица).

Структуры данных:

Понятие Структуры данных - Структуры в C позволяют группировать различные данные вместе. Это может быть полезно, если у вас есть несколько связанных переменных, которые нужно использовать вместе. Например, если вы создаете программу для управления базой данных, вы можете создать структуру для каждого элемента в вашей базе данных. Это позволит вам хранить всю информацию об элементе вместе, вместо того чтобы хранить каждую переменную отдельно.

Линейные структуры данных (стек, дек, хеш-таблица) - В C нет прямых эквивалентов стека, дека и хеш-таблицы, но есть похожие структуры данных, такие как массивы и связанные списки.

Связанный список - это структура данных, которая состоит из узлов, каждый из которых содержит данные и ссылку на следующий узел в списке. Это позволяет создавать линейные структуры данных, подобные стеку, деку и очереди. Однако, они не имеют такой же производительности, как специализированные структуры данных.

Хеш-таблицы также не являются частью стандартного C, но могут быть реализованы с использованием массивов и связанных списков. Каждый элемент данных хранится в массиве, а связанный список используется для хранения всех элементов с одинаковым хеш-значением.

25. Структуры данных. Понятие Структуры данных. Циклические структуры данных

(циклический список, циклическая очередь, циклический дек). Назначение

циклических структур данных.

Структуры данных:

Понятие Структуры данных - Структуры в C позволяют группировать различные данные вместе. Это может быть полезно, если у вас есть несколько связанных переменных, которые нужно использовать вместе. Например, если вы создаете программу для управления базой данных, вы можете создать структуру для каждого элемента в вашей базе данных. Это позволит вам хранить всю информацию об элементе вместе, вместо того чтобы хранить каждую переменную отдельно.

Циклические структуры данных (циклический список, циклическая очередь, циклический дек) -

Циклический (кольцевой) список - это структура данных, представляющая собой последовательность элементов, последний элемент которой содержит указатель на первый элемент списка, а первый (в случае двунаправленного списка) – на последний. Основная особенность такой организации состоит в том, что в этом списке нет элементов, содержащих пустые указатели, и, следовательно, нельзя выделить крайние элементы.

Циклическая очередь – это линейная структура данных, в которой операции выполняются по принципу FIFO (первым пришел – первым обслужен), а последняя позиция соединяется с первой позицией, образуя круг.

В программировании циклический дек (или кольцевой двусвязный список) - это структура данных, которую можно использовать для создания структур данных, подобных стеку и очереди, но с добавлением возможности бесконечного циклического обхода. Это полезно в некоторых приложениях, таких как обработка текста, где вы можете захотеть обработать каждое слово в тексте.

Назначение циклических структур данных - В программировании назначение циклических структур данных может включать в себя создание стеков, очередей и дек на основе линейных структур данных, таких как массивы и связанные списки, а также использование этих структур для реализации алгоритмов и решения задач.

26. Структуры данных. Понятие Структуры данных. Нелинейные структуры данных

(граф, дерево).

Структуры данных:

Понятие Структуры данных - Структуры в C позволяют группировать различные данные вместе. Это может быть полезно, если у вас есть несколько связанных переменных, которые нужно использовать вместе. Например, если вы создаете программу для управления базой данных, вы можете создать структуру для каждого элемента в вашей базе данных. Это позволит вам хранить всю информацию об элементе вместе, вместо того чтобы хранить каждую переменную отдельно.

Нелинейные структуры данных (граф, дерево) - В программировании нелинейные структуры данных могут включать графы и деревья. Графы используются для представления сложных структур данных, таких как социальные сети или сети дорог. Деревья используются для организации данных в иерархическом порядке, например, в базах данных или при обходе деревьев в программировании.

27. Структуры данных. Понятие Структуры данных. Виды структур данных. Функции для работы с динамической памятью (malloc, free, realloc).

Структуры данных:

Понятие Структуры данных - Структуры в C позволяют группировать различные данные вместе. Это может быть полезно, если у вас есть несколько связанных переменных, которые нужно использовать вместе. Например, если вы создаете программу для управления базой данных, вы можете создать структуру для каждого элемента в вашей базе данных. Это позволит вам хранить всю информацию об элементе вместе, вместо того чтобы хранить каждую переменную отдельно.

Виды структур данных - В программировании существуют различные виды структур данных, включая линейные и циклические структуры, а также нелинейные структуры, такие как графы и деревья. Линейные структуры включают массивы, связанные списки, стеки, очереди и деки. Циклические структуры включают циклические списки, циклические очереди и циклические деки. Нелинейные структуры включают графы и деревья.

Функции для работы с динамической памятью (malloc, free, realloc) - В C существуют функции для работы с динамической памятью, такие как malloc, free и realloc.

malloc - это функция, которая выделяет память для хранения данных. Она принимает размер требуемой памяти в байтах и возвращает указатель на выделенную область памяти.

free - это функция, которая освобождает ранее выделенную память. Она принимает указатель на ранее выделенную область памяти и освобождает эту область.

realloc - это функция, которая изменяет размер выделенной области памяти. Она принимает указатель на ранее выделенную область памяти, новый размер области в байтах и флаг, указывающий, нужно ли скопировать содержимое области в новую область памяти (если новый размер больше текущего). Если новый размер равен нулю, то функция освобождает выделенную область памяти (эквивалентно вызову free).