1. Компилятор - это программное обеспечение, которое преобразует исходный код программы на высокоуровневом языке программирования в машинный код, который может быть выполнен компьютером. Некоторые известные компиляторы - GCC, Clang, Microsoft Visual C++.  
  
2. Компиляция программ - это процесс преобразования исходного кода программы в машинный код. Дебаггинг - процесс поиска и устранения ошибок в программе. ГДБ (GDB) - отладчик для программ на языке Си. Дебаггинг памяти - процесс отладки программы для поиска утечек памяти или других проблем с памятью.  
  
3. Структура программы на С - программы на языке C состоят из объявлений переменных, операторов и функций.  
  
4. Арифметические операции и их порядок - в языке C используются стандартные арифметические операции: сложение, вычитание, умножение, деление. Порядок выполнения операций определяется правилами приоритета операторов.  
  
5. Типы функций - в языке C есть различные типы функций, такие как функции без возвращаемого значения (void), функции с возвращаемым значением (int, double, char и т. д.), а также функции с параметрами.  
  
6. Функция main - в языке C основная функция, которая вызывается при запуске программы.  
  
7. Строки - массив символов, заканчивающийся нулевым символом. Строки могут быть заданы с помощью двойных кавычек. Для ввода и вывода строк используются функции printf и scanf.  
  
8. Возврат функции - функции в языке C могут возвращать значение при помощи оператора return.  
  
9. Переменные - в языке C переменные имеют имена, которые могут содержать буквы, цифры и знак подчеркивания.  
  
10. Big endian and little endian - это способы упорядочения байтов в памяти. В big endian старший байт хранится первым, а в little endian - младший байт. Последовательность данных  
  
11. Операторы(логические, в том числе <, > и тд) Деление и модуль. Побитовые операторы - в языке C используются операторы деления (/) и модуля (%). Также есть побитовые операторы (И, ИЛИ, исключающее ИЛИ, сдвиг влево, сдвиг вправо).  
  
12. Условные операторы. Различие между if и if-else. Оператор IF - условный оператор, который выполняет определенный набор инструкций, если условие истинно.  
  
13. Оператор Switch - оператор выбора, который позволяет выбрать одну из нескольких альтернатив, основываясь на значении выражения.  
  
14. Циклические операторы. Цикл Do, While, Break, Continue - в языке C есть различные виды циклов: do-while, while, for. Break используется для выхода из цикла, а continue для перехода к следующей итерации цикла.  
  
15. Функции(глобальные, локальные и статические переменные). Функция, возврат множества значений - в языке C функция может возвращать только одно значение. Однако она может модифицировать значения переменных, переданных ей по указателю, чтобы вернуть несколько значений.  
  
16. Ввод/вывод. Что такое printf, как выглядит, спецификация форматирования, разобраться с форматами вывода  
  
17. Форматированный вход - ввод данных с определенным форматом, используя функцию scanf.  
  
18. Массивы символов - структура данных, представляющая собой последовательность символов. В C строки обычно представлены массивами символов. String/share  
  
19. Указатели - переменные, содержащие адреса других переменных. Они используются для работы с памятью напрямую.  
  
20. Физическая и виртуальная память - физическая память - это оборудование компьютера для хранения данных, виртуальная память - это расширение физической памяти, которое позволяет работать с большими объемами данных.  
  
21. Переменные адреса - в языке C можно получить адрес переменной с помощью оператора &. Адресация переменных  
  
22. Массив указателей - массив, элементами которого являются указатели на переменные.  
  
23. Поиск и алгоритмы сортировки - в языке C можно использовать различные алгоритмы для поиска и сортировки данных, такие как линейный поиск, бинарный поиск, сортировка пузырьком, сортировка слиянием и другие

24. Определяемые пользователем типы данных strchr/union. Связанные списки

Типы данных, определяемые пользователем: объединения.

Объединение (union) представляет собой частный случай структуры, все поля которой располагаются по одному и тому же адресу. Формат описания такой же, как у структуры, только вместо ключевого слова struct используется слово union. Длина объединения равна наибольшей из длин его полей. В каждый момент времени в переменной типа объединение хранится только одно значение, и ответственность за его правильное использование лежит на программисте.

Функция strchr ищет первое вхождения символа, код которого указан в аргументе ch, в строке, на которую указывает аргумент str.

Связанный список - это набор динамически выделяемых узлов, организованных таким образом, что каждый узел содержит одно значение и один указатель. Указатель всегда указывает на следующий член списка. Если указатель NULL, то это последний узел в списке. Связанный список хранится с использованием локальной переменной-указателя, которая указывает на первый элемент списка. Если этот указатель также имеет значение NULL, то список считается пустым.

Существует три основных типа связанных списков.

Односвязный список.

Это самый распространенный тип связных списков. В каждом узле есть указатель на следующий узел.

Двусвязный список.

Если добавить указатель на предыдущий узел, получится двусвязный список. В нём можно «двигаться» в обоих направлениях: и вперед, и назад.

Поэтому двусвязный список еще называют двунаправленным.

Кольцевой связный список.

В кольцевом связанном списке последний элемент связан с первым элементом.

Кольцевой связный список может быть как односвязным, так и двусвязным.

В односвязном указатель на следующий элемент последнего узла указывает на первый узел.

В двусвязном указатель на предыдущий элемент первого узла указывает на последний узел.

https://codechick.io/tutorials/dsa/dsa-linked-list-types

25. Структуры данных

Структура в языке программирования Си представляет собой составной тип данных, который состоит из других компонентов. При этом в отличие от массива эти компоненты могут представлять различные типы данных.

Определение структуры

Для определения структуры применяется ключевое слово struct, а сам формат определения выглядит следующим образом:

struct имя\_структуры

{

компоненты\_структуры

};

Имя\_структуры представляет произвольный идентификатор, к которому применяются те же правила, что и при наименовании переменных.

После имени структуры в фигурных скобках помещаются компоненты структуры - объекты, которые составляют структуру.

Следует отметить, что в отличие от функции при определении структуры после закрывающей фигурной скобки идет точка с запятой.

Например, определим простейшую структуру:

struct person

{

int age;

char \* name;

};

Здесь определена структура person, которая имеет два элемента: age (представляет тип int) и name (представляет указатель на тип char).

Все элементы структуры объявляются как обычные переменные. Но в отличие от переменных при определении элементов структуры для них не выделяется память, и их нельзя инициализировать. По сути мы просто определяем новый тип данных.  
  
26. Сборщик мусора

Сборщик мусора (garbage collector) - это функция исполнительной системы (runtime system) языка программирования. Сборщик мусора выполняет обнаружение и утилизацию недостижимых объектов, не нуждаясь в управлении приложением, хотя приложение может иметь в своем распоряжении различные средства контроля работы сборщика. Детальное рассмотрение всех проблем сборки мусора требует отдельной книги.  
  
27. Многопоточность. Дать определение процесс и поток. Библиотеки на многопоточность в си. Всё о мьютексе. Семафоры

Многопоточность в языке Си

Многопоточность - это метод программирования, позволяющий повысить производительность приложения с помощью параллельных процессов.

Использование множества потоков может значительно увеличить скорость выполнения программы. В настоящее время возможности повышения частоты работы процессора практически исчерпаны и производители идут по пути увеличения ядер процессора. Не все задачи допускают распараллеливание, но там где это возможно прирост может составлять чуть меньшее число раз чем количество ядер (включая виртуальные от технологии Hyper-threading).

**Би**блиотеки на многопоточность: pthread.h и функций pthread\_create() и pthread\_join():

Поток – отдельное исполняемое задание внутри процесса.

Процесс – это исполняемая копия приложения.

Процесс – это исполняемая копия приложения. Например, когда вы открываете приложение MS Word, то запускаете процесс, исполняющий программу MS Word. Поток – отдельное исполняемое задание внутри процесса. Процесс может содержать множество исполняемых потоков. После запуска приложения исполняется главный поток, который далее может порождать другие потоки.

Каждый процесс обладает собственной памятью. Потоки же, которые запущены внутри процесса, разделяют память между собой. Процесс внутри операционной системы обладает собственным идентификатором. Потоки существуют внутри процесса и обладают идентификатором внутри работающего приложения. Каждый из потоков имеет свой собственный стек (он не делит его с другими потоками и другие потоки не могут в него залезть) и собственный набор регистров (поток не изменит значения регистра другого потока во время работы). Часто потоки называют «легковесными» процессами, так как они требуют гораздо меньше ресурсов для работы, чем новый процесс. В зависимости от реализации, обычный настольный компьютер может эффективно использовать от единиц, до десятков тысяч потоков.

Мьютекс (mutex - mut ually ex clusive), который также называют защелкой - это механизм изоляции, используемый сервером баз данных для синхронизации доступа нескольких потоков к совместно используемым ресурсам. Мьютексы аналогичны семафорам, которые используются некоторыми операционными системами для регулирования доступа нескольких процессов к совместно используемым данным.

Cемафор – это объект, который используется для контроля доступа нескольких потоков до общего ресурса. В общем случае это какая-то переменная, состояние которой изменяется каждым из потоков. Текущее состояние переменной определяет доступ к ресурсам.

В pthreads семафор – это переменная типа sem\_t, которая может находиться в заданном числе состояний. Каждый поток может увеличить счётчик семафора, или уменьшить его. В литературе операция увеличения значения счётчика называется V (от датского verhogen – увеличивать, или vrijgave - освобождать). Для простоты можно запоминать как английское vacate (освобождать). Уменьшение счётчика – это операция P (proberen – тестировать, или passeren – проходить, или pakken – захватывать, или даже probeer te verlagen – попытаться уменьшить), или по-английски procure – добывать.

*Замечание*: иногда мьютекс называют двоичным семафором, указывая не то, что мьютекс может находиться в двух состояниях. Но здесь есть важное отличие: разблокировать мьютекс может только тот поток, который его заблокировал, семафор же может «разблокировать» любой поток.

Семафоры описаны в библиотеке semaphore.h. Работа с семаформаи похожа на работу с мьютексами. Сначала необходимо инициализировать семафор с помощью функции