### 30. Виртуальная память:

Виртуальная память — это механизм операционных систем, который предоставляет приложениям больше памяти, чем физически доступно на компьютере. Она создается за счет комбинированного использования оперативной памяти (RAM) и файла подкачки (swap file или swap partition) на диске. Виртуальная память дает приложениям впечатление, что у них есть более обширное адресное пространство, чем оно есть на самом деле.

### 31. Свопинг:

Свопинг — это процесс передачи данных между оперативной памятью и файлом подкачки (swap space) на диске. Когда система обнаруживает, что физическая оперативная память исчерпана, она может перемещать неиспользуемые части данных или кода из памяти в файл подкачки и наоборот. Это помогает управлять ограниченным объемом оперативной памяти и обеспечивает работоспособность системы.

### 32. Страничная память:

Страничная память — это механизм виртуальной памяти, в котором физическая и виртуальная память делится на небольшие фрагменты, называемые страницами. Операционная система использует страничную память для эффективного управления памятью и обеспечения быстрого доступа к данным.

### 33. MMU (Memory Management Unit):

MMU — это устройство в процессоре, которое обеспечивает преобразование виртуальных адресов в физические и управление страничной памятью. MMU играет ключевую роль в реализации виртуальной памяти.

### 34. TLB (Translation Lookaside Buffer):

TLB — это буфер преобразования адресов, используемый MMU для хранения недавно выполненных преобразований виртуальных адресов в физические. Это ускоряет процесс доступа к данным, так как предотвращает необходимость выполнения преобразования каждый раз при обращении к памяти.

### 35. Информация в строке таблицы страниц:

Строка таблицы страниц содержит информацию о соответствии виртуальных страниц физическим адресам, состоянии страницы (например, в памяти, на диске), права доступа (чтение, запись, выполнение) и другие управляющие атрибуты.

### 36. Применение хэш-таблиц:

Хэш-таблицы применяются для эффективного поиска и связывания данных. Они используют хэш-функцию для преобразования ключа в индекс, где хранится соответствующее значение. Это позволяет быстро находить данные по ключу без необходимости обхода всей структуры данных.

### 37. Инвертированная таблица физических страниц:

Инвертированная таблица физических страниц — это структура данных, которая обеспечивает отображение физических адресов на соответствующие виртуальные адреса. Она используется для реализации виртуальной памяти и управления страницами в системе.

### 38. Рабочий набор страниц:

Рабочий набор страниц — это набор страниц в виртуальной памяти, которые активно используются приложением в данный момент времени. Он может изменяться по мере выполнения программы.

### 39. Принцип работы алгоритма LRU:

LRU (Least Recently Used) отслеживает, какие страницы были давно неиспользуемыми, и при необходимости выталкивает их из оперативной памяти для освобождения места. Принцип LRU заключается в том, что страница, к которой дольше всего не обращались, имеет наименьший приоритет для сохранения в памяти.

### 40. SysMain (Superfetch) в Windows:

SysMain (ранее Superfetch) — служба Windows, отвечающая за управление кэшем данных в оперативной памяти, предсказывая и загружая в память те данные, которые вероятно будут использованы.

### 41. hiberfil.sys в Windows:

hiberfil.sys — это файл подкачки, используемый для хранения данных гибернации (гибернационное состояние) в Windows. Он содержит сохраненное состояние системы, позволяя компьютеру быстро восстановиться после выключения.

### 42. pagefile.sys в Windows:

pagefile.sys — это файл подкачки, используемый для временного хранения данных, которые не умещаются в оперативной памяти. Этот файл подкачки поддерживает виртуальную память.

### 43. swapfile.sys в Windows:

swapfile.sys — это файл подкачки, введенный в Windows 10 для замены функциональности hiberfil.sys и pagefile.sys. Он обеспечивает как поддержку гибернации, так и виртуальной памяти.

### 44. Области адресного пространства в Windows (от младших к старшим адрес

ам):

1. \*\*Пользовательское пространство:\*\* 0x00000000 - 0x7FFFFFFF

2. \*\*Ядро:\*\* 0x80000000 - 0xFFFFFFFF

### 45. Стандартный начальный размер области heap в Windows:

Стандартный размер heap зависит от компилятора и версии операционной системы. Обычно начальный размер heap — 1 мегабайт.

### 46. Изменение начального размера heap в Windows:

Начальный размер heap можно изменить с помощью функции `HeapCreate` или средств управления памятью, предоставляемых библиотекой C, такими как `malloc`.

### 47. Стандартный размер области stack в Windows:

Стандартный размер stack зависит от версии Windows и настроек компилятора. Обычно 1 мегабайт.

### 48. Изменение размера области stack в Windows:

Размер stack можно изменить с помощью функции `SetThreadStackGuarantee` или при компиляции программы с настройками стека.

### 49. Назначение функции Windows API: GlobalMemoryStatus:

`GlobalMemoryStatus` предоставляет информацию о состоянии оперативной памяти и виртуальной памяти системы.

### 50. Назначение функции Windows API: VirtualQuery:

`VirtualQuery` предоставляет информацию о регионах виртуальной памяти, включая атрибуты `Protect`, `State` и `Type`.

### 51. Рабочее множество в Windows:

Рабочее множество (Working Set) — это набор физических страниц, связанных с виртуальным адресным пространством процесса в данный момент времени. Управление рабочим множеством осуществляется с использованием OS API, таких как `SetProcessWorkingSetSize`.

### 52. Страница заблокирована в Windows:

Страница заблокирована (locked page) означает, что ей присвоен атрибут блокировки, что предотвращает ее вытеснение из физической оперативной памяти. Функции OS API, такие как `VirtualLock` и `VirtualUnlock`, используются для блокировки и разблокировки страниц.

### 53. Heap в Windows:

Heap — это область памяти, выделенная для хранения динамически создаваемых данных. Процесс может иметь несколько heap'ов, каждый из которых управляется через функции, предоставляемые библиотекой C (например, `HeapCreate`, `HeapAlloc`).

### 54. Области адресного пространства в Linux (от младших к старшим адресам):

1. \*\*User Space:\*\* 0x00000000 - 0xFFFFFFFF (зависит от архитектуры)

2. \*\*Kernel Space:\*\* 0xC0000000 - 0xFFFFFFFF (зависит от архитектуры)

### 55. Выделение памяти в Linux с помощью malloc и calloc:

Память в Linux выделяется в пользовательском пространстве с использованием функций `malloc` и `calloc`. Эти функции возвращают указатель на выделенную область памяти.