1. Поясните понятие «виртуальная память».

**Виртуальная память** — это механизм управления памятью, при котором адреса, используемые программами (виртуальные адреса), отображаются на реальные физические адреса с помощью таблиц страниц. Виртуальная память позволяет:

* Использовать больше памяти, чем физически доступно.
* Защищать память процессов от несанкционированного доступа.
* Организовывать удобное адресное пространство для программ.

1. Поясните понятие «свопинг».

**Свопинг** — это процесс выгрузки страниц памяти, неактивных в данный момент, из оперативной памяти на диск (в область подкачки или swap-файл). Это освобождает оперативную память для более активных задач. Когда выгруженные страницы снова становятся нужны, они загружаются обратно в память.

1. Поясните понятие «страничная память».

**Страничная память** — это метод управления памятью, при котором виртуальное адресное пространство делится на равные по размеру блоки (страницы). Каждая страница виртуальной памяти отображается на фрейм (frame) физической памяти. Это упрощает управление памятью и минимизирует фрагментацию.

1. Поясните понятие MMU.

**MMU (Memory Management Unit)** — это аппаратный модуль (часть процессора), который управляет отображением виртуальных адресов в физические. Он обеспечивает:

* Преобразование адресов с помощью таблиц страниц.
* Проверку прав доступа.
* Ускорение доступа к памяти с использованием буфера TLB.

1. Поясните понятие TLB.

**TLB (Translation Lookaside Buffer)** — это высокоскоростной кэш в MMU, который хранит недавно использованные отображения виртуальных адресов на физические. Использование TLB ускоряет работу, так как обращение к памяти через таблицы страниц может быть медленным.

1. Какая информация содержится в строке таблицы страниц

* Физический адрес начала фрейма памяти.
* Статус страницы (валидна/невалидна).
* Права доступа (чтение, запись, выполнение).
* Флаги состояния (например, изменена ли страница или используется ли она).

1. Поясните принцип применения хэш-таблиц.

Хэш-таблицы используются для эффективного хранения и поиска данных. Они основаны на хэш-функциях, которые преобразуют ключи данных в хэш-коды. Хэш-таблица состоит из массива ячеек, где каждая ячейка содержит список элементов с одинаковыми хэш-кодами. Это позволяет быстро находить элементы по ключу.

1. Поясните применение «инвертированной таблицы физических» страничной памяти.

Инвертированная таблица физических страниц - это особый способ организации страничной памяти, где вместо того, чтобы хранить соответствия виртуальных страниц программы и физических фреймов в каждом процессе, используется общая таблица, известная как инвертированная таблица. Каждая запись в инвертированной таблице содержит информацию о соответствии виртуальной страницы идентификатору процесса и физическому адресу. Это позволяет экономить память, но требует дополнительных операций для поиска и обновления записей в таблице.

1. Windows: перечислите области адресного пространства (от младших к старшим адресам) и поясните их назначения.

В Windows области адресного пространства перечисляются от младших к старшим адресам следующие области:

а) Нижняя область адресного пространства (Lower Address Space) - эта область подразделяется на несколько сегментов:

- Сегмент кода (Code Segment) - в этом сегменте размещается исполняемый код программы.

- Сегмент данных (Data Segment) - в этом сегменте размещаются глобальные и статические данные программы.

- Сегмент стека (Stack Segment) - в этом сегменте размещается стек вызовов функций и локальные переменные.

б) Область адресного пространства, зарезервированная для системы (Reserved System Space) - эта область зарезервирована для нужд операционной системы и не доступна для пользовательских приложений.

в) Область адресного пространства, зарезервированная для пользовательских приложений (Reserved User Space) - это область адресного пространства, которая доступна для пользовательских приложений для выделения памяти и размещения данных.

1. **User Mode (0x00000000 - 0x7FFFFFFF):**
   * Code: выполняемый код программы.
   * Data: глобальные и статические данные.
   * Heap: динамически выделенная память.
   * Stack: память для локальных переменных и вызовов функций.
2. **Kernel Mode (0x80000000 - 0xFFFFFFFF):**
   * Kernel Code: код ядра операционной системы.
   * Device Drivers: память драйверов устройств.
   * System Cache: кеш системы и данные ядра.
3. Windows: какой стандартный начальный размер области heap?

1 мегабайт (1 MB). Но вообще, он динамически определяется.

Можно изменить с помощью функции HeapCreate из Windows API. При создании heap можно указать размер в байтах или в виде процентного соотношения от размера физической памяти компьютера. Также можно изменить размер heap с помощью функции HeapReAlloc, которая позволяет изменить размер уже существующего heap.

1. Windows: какой стандартный размер области памяти stack?

1 мегабайт (1 MB). Можно изменить с помощью функции SetThreadStackGuarantee из Windows API. Эта функция позволяет установить минимальный гарантированный размер stack для новых потоков, создаваемых в приложении.

1. Windows: что такое «рабочее множество»? поясните принцип управления рабочим множеством с помощью OS API.

Рабочее множество - набор физических страниц памяти, используемых процессом в данный момент. Управление рабочим множеством осуществляется с помощью OS API, таких как функции SetProcessWorkingSetSize и QueryWorkingSet из Windows API. С помощью SetProcessWorkingSetSize можно установить ограничения на размер рабочего множества процесса, а с помощью QueryWorkingSet можно получить информацию о текущем состоянии рабочего множества.

1. Windows: что такое «heap»? Что такое «heap процесса»? Что такое «пользовательская heap»? Поясните принцип устройства heap.

* **Heap:** область памяти для динамического выделения (например, через malloc или HeapAlloc).
* **Heap процесса:** основной heap, создаваемый системой для каждого процесса.
* **Пользовательская heap:** дополнительные heaps, которые может создавать разработчик с помощью функций, таких как HeapCreate.

Устройство heap основано на списке блоков памяти различных размеров, известном как список свободных блоков (free list). При запросе памяти heap ищет подходящий блок в списке свободных блоков и выделяет его для приложения. При освобождении памяти блок возвращается в список свободных блоков для последующего использования. Heap также выполняет операции сборки мусора и дефрагментации данных.

1. Linux: перечислите области адресного пространства (от младших к старшим адресам) и поясните их назначения.

В Linux области адресного пространства перечисляются от младших к старшим адресам следующие области:

а) Нижняя область адресного пространства:

- Область кода (Code Segment) - в этой области размещается исполняемый код программы.

- Область данных (Data Segment) - в этой области размещаются глобальные и статические данные программы.

- Область BSS (Block Started by Symbol) - в этой области размещаются неинициализированные глобальные и статические переменные.

- Область стека (Stack Segment) - в этой области размещается стек вызовов функций и локальные переменные.

б) Область адресного пространства, зарезервированная для библиотек и разделяемых объектов (Shared Libraries and Shared Objects) - в этой области размещаются библиотеки, используемые программой, и разделяемые объекты, которые могут быть загружены и используемы несколькими процессами одновременно.

в) Область адресного пространства, зарезервированная для отображения файлов в память (Memory-Mapped Files) - в этой области размещаются файлы, которые могут быть отображены в память и использованы как обычная память. Это позволяет обращаться к содержимому файла напрямую через указатели и использовать его без необходимости чтения и записи на диск.

г) Область адресного пространства, зарезервированная для стека сигналов (Signal Stack) - в этой области размещается стек, используемый для обработки сигналов, которые могут быть отправлены процессу.

д) Область адресного пространства, зарезервированная для динамической загрузки библиотек (Dynamic Linking) - в этой области размещаются библиотеки, которые загружаются динамически во время выполнения программы.

е) Область адресного пространства, зарезервированная для кучи (Heap Segment) - в этой области происходит выделение и управление динамической памятью с помощью функций, таких как malloc и calloc.

1. **Text Segment:** код программы.
2. **Data Segment:** глобальные и статические данные.
3. **Heap:** динамически выделенная память.
4. **Stack:** локальные переменные и вызовы функций.
5. **Shared Libraries:** подключаемые библиотеки.
6. **Kernel Space:** адресное пространство ядра.
7. Linux: в какой части адресного пространства выделяется памяти с помощью функций malloc, calloc?

Куча - область памяти, используемая для динамического выделения и освобождения памяти во время выполнения программы. Область адресного пространства, выделенная для кучи, находится в верхней части адресного пространства процесса и растет в направлении старших адресов при выделении памяти.