Вопросы

1. **Виртуальная память** - это механизм, который позволяет программам использовать больше памяти, чем физически доступно на компьютере. Он делает это, создавая абстракцию адресного пространства, которая отличается от физической памяти. Каждому процессу предоставляется собственное виртуальное адресное пространство, которое изолировано от других процессов.
2. **Свопинг** - это процесс перемещения данных между оперативной памятью и пространством подкачки (или файлом подкачки) на диске. Когда системе не хватает оперативной памяти, она может переместить некоторые страницы памяти в пространство подкачки, освобождая оперативную память для других задач. Когда эти страницы памяти снова требуются, они могут быть возвращены обратно в оперативную память.
3. **Страничная память** - это механизм управления памятью, при котором память делится на блоки одинакового размера, называемые страницами. Виртуальное адресное пространство также делится на страницы того же размера. Страницы виртуальной памяти могут быть отображены на страницы физической памяти, что позволяет эффективно использовать доступную память и изолировать процессы друг от друга.
4. **MMU (Memory Management Unit)** - это компонент аппаратного обеспечения, который отвечает за преобразование виртуальных адресов в физические адреса. Он использует таблицу страниц для отображения виртуальных адресов на физические адреса.
5. **TLB (Translation Lookaside Buffer)** - это кэш, который хранит недавние отображения из виртуальных адресов в физические адреса. Кэш утилиты MMU. Он используется для ускорения преобразования адресов, сокращая количество обращений к таблице страниц.
6. **Строка таблицы** страниц содержит информацию о соответствии виртуальных страниц программы физическим фреймам памяти. Она может включать в себя информацию, такую как номер виртуальной страницы, номер физического фрейма, флаги доступа и прав доступа (например, чтение, запись, выполнение).(такое как присутствующее или отсутствующее) и другую информацию.
7. **Хэш-таблицы** используются для эффективного хранения и поиска данных. Они работают, преобразуя ключ в индекс массива с помощью хэш-функции. Затем значение, связанное с ключом, хранится в массиве по этому индексу. При поиске значения по ключу хэш-функция снова используется для определения индекса, и значение извлекается из массива.
8. **Инвертированная таблица страниц** - это структура данных, используемая в некоторых системах управления памятью для отображения физических адресов на виртуальные адреса. В отличие от обычной таблицы страниц, которая хранит отображение для каждой виртуальной страницы, инвертированная таблица страниц хранит только одну запись для каждой физической страницы.
9. **Рабочий набор страниц** - это набор страниц памяти, которые активно используются процессом в данный момент. Он используется операционной системой для определения, какие страницы должны оставаться в оперативной памяти, а какие могут быть вытеснены.
10. **Алгоритм LRU (Least Recently Used)** - это стратегия управления памятью, которая вытесняет наименее недавно использованные страницы, когда требуется место для новых страниц. Он работает на предположении, что страницы, которые недавно использовались, скорее всего будут использоваться снова в ближайшем будущем.
11. **SysMain** (ранее известный как SuperFetch) - это служба в Windows, которая управляет предварительной загрузкой приложений в память для ускорения их запуска. Он анализирует использование приложений с течением времени и определяет, какие приложения и данные следует предварительно загрузить.

Мое: Кэширует данные, ускоряет запуск приложений т.к. они находятся в кэше, изменяет рабочий набор страниц, отслеживает активность ресурсов

1. **Файл hiberfil.sys** используется Windows для поддержки гибернации. Когда компьютер переходит в режим гибернации, все данные из оперативной памяти сохраняются в файле hiberfil.sys. При выходе из режима гибернации данные загружаются обратно в оперативную память.

(Если что без режима гибернации при выключении системы данные из ОЗУ утрачиваются)

1. **Файл pagefile.sys** - это файл подкачки в Windows. Он используется как виртуальная память, когда оперативной памяти становится недостаточно. Данные, которые в настоящее время не используются, могут быть перемещены из оперативной памяти в файл pagefile.sys, освобождая оперативную память для других задач.
2. **Файл swapfile.sys** - это еще один файл подкачки, используемый Windows, в основном для современных приложений Windows Store. Он работает аналогично файлу pagefile.sys, но обычно имеет меньший размер.
3. **Области адресного пространства в Windows** (от младших к старшим адресам) включают:
   * **Область нулевых страниц**: Это первые 64КБ виртуального адресного пространства, которые зарезервированы и не используются. Это предотвращает обращение к нулевому указателю.
   * **Область кода**: Здесь располагается исполняемый код приложения.
   * **Область данных**: Здесь хранятся глобальные и статические переменные.
   * **Heap**: Это область памяти, которую приложение может динамически выделить и освободить.
   * **Stack**: Здесь хранятся локальные переменные и информация о вызовах функций.
   * **Отображения файлов и разделяемая память**: Это области памяти, которые отображают файлы на диске или разделяют память между процессами.
   * **Свободное пространство**: Это нераспределенная область адресного пространства, которую можно использовать для дальнейшего расширения областей кода, данных, кучи или стека.

Вариант 2   
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. **Стандартный начальный размер области heap в Windows** составляет 1MB для 32-битных приложений и 2MB для 64-битных приложений.
2. **Изменение начального размера области heap в приложении Windows** можно выполнить, изменяя параметры компоновщика в настройках проекта в Visual Studio. Вы можете установить параметр “Heap Reserve Size” (Размер резерва heap) в желаемое значение.
3. **Стандартный размер области памяти stack в Windows** составляет 1MB для 32-битных приложений и 4MB для 64-битных приложений.
4. **Изменение размера области памяти stack в приложении Windows** можно выполнить, изменяя параметры компоновщика в настройках проекта в Visual Studio. Вы можете установить параметр “Stack Reserve Size” (Размер резерва стека) в желаемое значение.  
   Размер области памяти stack приложения можно изменить с помощью функции SetThreadStackGuarantee из Windows API. Эта функция позволяет установить минимальный гарантированный размер stack для новых потоков, создаваемых в приложении. При увеличении размера stack необходимо учесть доступное виртуальное адресное пространство и ограничения операционной системы.
5. **Функция Windows API GlobalMemoryStatus** используется для получения информации о состоянии физической и виртуальной памяти. Она заполняет структуру MEMORYSTATUS информацией о текущем состоянии памяти.
6. **Функция Windows API VirtualQuery** используется для получения информации о диапазоне страниц в адресном пространстве вызывающего процесса. Она заполняет структуру MEMORY\_BASIC\_INFORMATION информацией о текущем состоянии памяти. Значения атрибутов Protect, State и Type могут быть следующими:
   * **Protect**: Определяет защиту доступа к страницам в регионе. Значения могут включать PAGE\_READONLY, PAGE\_READWRITE, PAGE\_EXECUTE, PAGE\_EXECUTE\_READ, PAGE\_EXECUTE\_READWRITE и другие.
   * **State**: Определяет состояние страниц в регионе. Значения могут включать MEM\_COMMIT (страницы выделены и используются), MEM\_RESERVE (резервирование страниц без выделения памяти), MEM\_FREE (страницы выделены но уже не используются).
   * **Type**: Определяет тип страниц в регионе. Значения могут включать MEM\_IMAGE (страницы связаны с исполняемыми файлами и обычно содержат код или данные), MEM\_MAPPED (страницы отображают файл или устройство, исп для обмена данными между процессами), MEM\_PRIVATE (страницы являются частными для процесса).
7. **Рабочее множество** в Windows - это набор страниц памяти, которые были недавно обращены к процессу. Управление рабочим множеством в Windows можно выполнить с помощью функций API GetProcessWorkingSetSize, SetProcessWorkingSetSize, EmptyWorkingSet и других.
8. **Страница заблокирована** в Windows означает, что страница не может быть вытеснена из физической памяти. Функции API VirtualLock и VirtualUnlock используются для установки и снятия блокировки страниц. Максимальное количество страниц, которые можно заблокировать, зависит от размера рабочего множества процесса и может быть изменено.
9. **Heap** в Windows - это регион в адресном пространстве процесса, который можно использовать для динамического выделения памяти. **Heap процесса** - это heap, который создается системой для каждого процесса, и он используется для выделения памяти для процесса и его потоков. **Пользовательская heap** - это heap, который создается процессом с помощью функции HeapCreate. Heap состоит из одного или нескольких сегментов, каждый из которых представляет собой непрерывный диапазон страниц памяти.
10. **Области адресного пространства в Linux** (от младших к старшим адресам) включают:
    * **Область кода**: Здесь располагается исполняемый код приложения.
    * **Область данных**: Здесь хранятся глобальные и статические переменные.
    * **Heap**: Это область памяти, которую приложение может динамически выделить и освободить.
    * **Отображения файлов и разделяемая память**: Это области памяти, которые отображают файлы на диске или разделяют память между процессами.
    * **Stack**: Здесь хранятся локальные переменные и информация о вызовах функций.
11. В Linux память, выделенная с помощью функций malloc и calloc, обычно размещается в области heap. Это область памяти, которую приложение может динамически выделить и освободить. Heap расположен между областями данных и стека и может динамически расширяться и сжиматься.