**Задание 04. Ответьте на следующие вопросы**

1. Что такое фреймворк OS?

Набор функций для взаимодействия с ос

1. Что такое POSIX?

**POSIX** (англ. Portable Operating System Interface — переносимый интерфейс операционных систем) — набор стандартов, описывающих интерфейсы между операционной системой и прикладной программой (системный API), библиотеку языка C и набор приложений и их интерфейсов.

1. Что такое аппаратное прерывание?

(сигнал от ..) (если скажешь что это парадигма как в википедии, тебе пизда)

аппаратные - возникают как реакция микропроцессора на физический сигнал от некоторого устройства (клавиатура, системные часы, клавиатура, жесткий диск и т.д.), по времени возникновения эти прерывания асинхронны, т.е. происходят в случайные моменты времени;

1. Что такое программное прерывание?

(сигнал от ..) (если скажешь что это парадигма как в википедии, тебе пизда)

программные - вызываются искусственно с помощью соответствующей команды из программы (int), предназначены для выполнения некоторых действий операционной системы, являются синхронными;

1. Что такое системный вызов?

системный вызов - механизм вызова прикладной программой функции ядра OS. Системный вызов осуществляется с помощью программного прерывания (RISC/x86 – int) или новый механизм Intel x86\_64 - SYSENTER/SYSEXIT, AMD - SYSCALL/SYSRET).

1. Что такое процесс OS?

**Проце́сс** — **это** в выполняемая в данный момент программа

1. Что такое контекст процесса OS?

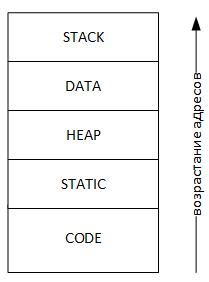
Каждый **процесс** UNIX имеет **контекст**, под которым понимается вся информация, требуемая для описания **процесса**. Эта информация сохраняется, когда выполнение **процесса** приостанавливается, и восстанавливается, когда планировщик предоставляет процессу вычислительные ресурсы.

**контекст процесса**: адресное пространство, содержимое регистров (общего назначения, счетчик команд, состояния процессора, вершина стека, …), объекты ядра OS (объекты процессов, потоков, безопасности, файлов и пр.), стек ядра (для этого процесса). Переключение контекстов. В ядре специальный стек для переключения контекстов.

1. Что такое адресное пространство процесса?

**А́дресное пространство** (англ. address space) — совокупность всех допустимых адресов каких-либо объектов вычислительной системы — ячеек памяти, секторов диска, узлов сети и т. п., которые могут быть использованы для доступа к этим объектам при определенном режиме работы (состоянии системы).

1. Перечислите области памяти адресного пространства процесса и поясните их назначение.



Stack - для временного хранения данных программы, результатов промежуточных вычислений

Data – для постоянного хранения инициализированных данных программы

Heap – для хранения динамических данных программы

Static – для хранения статических данных программы

Code – для хранения скомпилированного кода программы

1. Что такое стандартные потоки процесса?

**Стандартные потоки** ввода-вывода в системах типа UNIX (и некоторых других) — **потоки процесса**, имеющие номер (дескриптор), зарезервированный для выполнения некоторых «**стандартных**» функций.

1. Перечислите системные вызовы Windows для создания процесса?

CreateProcess, CreateProcessAsUser, CreateProcessWithTokenW, CreateProcessWithLogonW

1. Перечислите системные вызовы Linux для создания процесса?

Начинается с Fork(), он создает точный клон вызывающего процесса, так называемый «дочерний» процесс

Менеджер исполнения exec() заменяет образ процесса этого клона новой программой, которая должна быть выполнена.

fork() создает новое адресное пространство, которое полностью идентично адресному пространству основного процесса. После выполнения этого системного вызова мы получаем два абсолютно одинаковых процесса - основной и порожденный. Функция fork() возвращает 0 в порожденном процессе и PID (Process ID - идентификатор порожденного процесса) - в основном. PID - это целое число.

Теперь, когда мы уже создали процесс, мы можем запустить программу с помощью вызова exec. Параметрами функции exec является имя выполняемого файла и, если нужно, параметры, которые будут переданы этой программе. В адресное пространство порожденного с помощью fork() процесса будет загружена новая программа и ее выполнение начнется с точки входа (адрес функции main).

1. С помощью каких утилит можно увидеть перечень процессов в Windows?

Для управления процессами в командной строке есть две утилиты — **tasklist** и **taskkill**. Первая показывает список процессов на локальном или удаленном компьютере, вторая позволяет их завершить.

1. С помощью каких утилит можно увидеть перечень процессов в Linux?

|  |  |
| --- | --- |
| Ps | ps показывает запущенные процессы |
| Top | консольная команда, которая выводит список работающих в системе процессов и информацию о них |
| Lsof | LSOF является аббревиатурой от List Of Opened Files, и предназначена она для отображения открытых файлов различными процессами и/или пользователями |

1. Перечислите свойства процесса OS.

Потоки выполнения отличаются от традиционных процессов многозадачной операционной системы тем, что:

* процессы, как правило, независимы, тогда как потоки выполнения существуют как составные элементы процессов
* процессы несут значительно больше информации о состоянии, тогда как несколько потоков выполнения внутри процесса совместно используют информацию о состоянии, а также память и другие вычислительные ресурсы
* процессы имеют отдельные адресные пространства, тогда как потоки выполнения совместно используют их адресное пространство
* процессы взаимодействуют только через предоставляемые системой механизмы связей между процессами
* переключение контекста между потоками выполнения в одном процессе, как правило, быстрее, чем переключение контекста между процессами.

основные свойства процесса:

- процессу соответствует исполняемый программный файл;

- у процесса есть PID;

- у процесса есть Parent PID;

- в Windows: HANDEL – идентификатор объекта OS;

- в OS есть процесс инициализации (родитель для всех);

- запуск и управление (создать, остановить,…) процессом осуществляется с помощью системных вызовов;

- процессы изолированы друг от друга;

- процессу выделяется линейное адресное пространство (размер зависит от разрядности), сегменты: code, static, data, heap, stack;

- контекст процесса – данные, которые сохраняются при переключении процессов и предназначенные для продолжения работы;

- процессу автоматически доступны три процесса: ввода, вывода, вывод ошибок.

- при запуске OS некоторые процессы (Windows-сервисы, Linux-демоны) загружаются и стартуют автоматически, как правило используются для внутреннего назначения;

- в составе ОS есть таблица, содержащая объекты ядра процессов (состояние, приоритет, указатели на другие объекты); есть средства OS позволяющие ее просматривать;

- процесс – единица работы OS.

1. ***Что такое POSIX?***

**POSIX (англ. Portable Operating System Interface — переносимый интерфейс операционных систем)** — набор стандартов, описывающих интерфейсы между операционной системой и прикладной программой (системный API), библиотеку языка C и набор приложений и их интерфейсов.

1. ***Что такое системный вызов***?

**системный вызов** - механизм вызова прикладной программой функции ядра OS. Системный вызов осуществляется с помощью программного прерывания (RISC/x86 – int) или новый механизм Intel x86\_64 - SYSENTER/SYSEXIT, AMD - SYSCALL/SYSRET).

1. ***Что такое аппаратное прерывание, программное прерывание?***

**аппаратные** - возникают как реакция микропроцессора на физический сигнал от некоторого устройства (клавиатура, системные часы, клавиатура, жесткий диск и т.д.), по времени возникновения эти прерывания асинхронны, т.е. происходят в случайные моменты времени;

программные - вызываются искусственно с помощью соответствующей команды из программы (int), предназначены для выполнения некоторых действий операционной системы, являются синхронными;

1. ***Что такое процесс***?

**Проце́сс** — это в выполняемая в данный момент программа

**процесс OS** – единица работы OS - **объект ядра OS+адресное пространство**

1. ***Что такое контекст процесса?***

Каждый **процесс** UNIX имеет **контекст**, под которым понимается вся информация, требуемая для описания **процесса**. Эта информация сохраняется, когда выполнение **процесса** приостанавливается, и восстанавливается, когда планировщик предоставляет процессу вычислительные ресурсы.

1. ***Что такое родительский и дочерний процесс?***

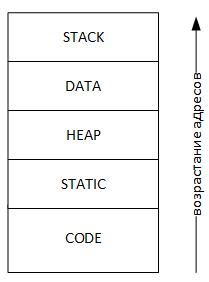
***Родительский процесс*** — это процесс, который создал («породил») один или несколько дочерних процессов («процессов потомков»). За счёт чего, процесс может стать дочерним или родительским и наоборот.( Процесс, который вызвал fork, является родительским процессом, а вновь созданный процесс-дочерним процессом.)

**Дочерний процесс** - это процесс, созданный другим процессом (родительским процессом). Этот метод относится к многозадачным операционным системам и иногда называется подпроцессом или традиционно подзадачей.

1. **Что такое процесс инициализации OS?**

в OS есть **процесс инициализации** (родитель для всех);

Процесс инициализации запускает все процессы, которые должны быть запущены и является для них родительским процессом

1. **Перечислите области памяти процесса и поясните их назначение.**
2. 
3. Stack - для временного хранения данных программы, результатов промежуточных вычислений
4. Data – для постоянного хранения инициализированных данных программы
5. Heap – для хранения динамических данных программы
6. Static – для хранения статических данных программы
7. Code – для хранения скомпилированного кода программы

**9.Чем отличаются системные процессы от пользовательских**?

**системные** **процессы:** процессы запускаемые автоматически при запуске OS;

**Системные процессы** - процессы, которые запускаются без вашего ведома, т. к. их работа необходима для работы системы, в общем их лучше не трогать, если не знаешь за что отвечает определенный процесс.   
**Пользовательские процессы** - приложения, которые запустил пользователь, т. е процесс запущен от имени пользователя.

1. Что такое Windows-сервисы, Linux-демоны?

windows-сервисы; Linux-демоны-процессы запускаемые автоматически при запуске OS, работающая в фоновом режиме без прямого взаимодействия с пользователем.

1. **С помощью каких системных вызовов можно создать дочерний процесс в Windows? Поясните разницу.**

CreateProcess, CreateProcessAsUser, CreateProcessWithTokenW, CreateProcessWithLogonW

12.**С помощью каких системных вызовов можно создать дочерний процесс в Linux? Поясните разницу.**

System

Execv

system-Функция system() определена в стандартной библиотеке языка С и позволяет вызывать из программы системную команду, как если бы она была набрана в командной строке.

 execv - execv вызывает уже скомпилированное приложение, а не создает новое

1. Какие потоки данных доступны любому процессу автоматически?

процессу автоматически доступны три потока: ввода, вывода, вывод ошибок.

1. Поясните назначение системного вызова WaitForSingleObject в Windows-приложении.

В случае многопоточности иногда нам нужно дождаться завершения потока, прежде чем продолжить выполнение других действий.Для этого вы можете использовать функцию Windows API WaitForSingleObject или WaitForMultipleObjects. Обе эти функции ждут, пока объект не будет помечен как сигнализированный, прежде чем вернуться.

1. Поясните назначение системного вызова wait в Linux-приложении.

Она блокирует вызывающий процесс до тех пор, пока один из его дочерних процессов не завершится (или не произойдет ошибка).

1. Дайте развернутое определение процесса OS.

процесс OS – единица работы OS - объект ядра OS+адресное пространство:

- создается ядром OS по системному вызову;

1. **Что такое поток управления OS?**

объект ядра операционной системы, которому OS выделяет процессорное время или последовательность инструкций, выполняемых процессором в выделенные OS интервалы времени.

1. **С помощью каких системных вызовов создаются потоки в Windows и Linux?**

CreateThread(Windows), pthread\_creare(Linux)

1. **Что такое системные и пользовательские потоки?**

Системный поток – поток, который создается и распределяется ядром ОС в пространстве ядра, а пользовательский – создается и управляется библиотекой потоков, планирование управляется самой библиотекой потоков в пользовательском пространстве.

1. **Что такое многопоточность?**

парадигма программирования, состоящая в том, что процесс, порождённый в операционной системе, может состоять из нескольких потоков, выполняющихся  без предписанного порядка во времени, и время переключения между потоками очень мало.

1. **Чем отличаются приоритетная многопоточность от кооперативной многопоточности?**

Приоритетная многопоточность предусматривает выбор потока и предоставление ему возможности работать до истечения некоторого строго определенного периода времени, а при кооперативной многопоточности следующий поток выполняется только после того, как текущий поток явно объявит каким-либо образом об освобождении процессорного времени для других потоков.

1. **Что такое диспетчеризация потоков управления OS?**

диспетчеризация потоков заключается в реализации найденного в результате планирования решения, т.е. в переключении процессора с одного потока на другой и осуществляется OS или самим потоком; Диспетчеризация сводится к следующему:

* сохранение контекста текущего потока, который требуется сменить;
* загрузка контекста нового потока, выбранного в результате планирования;
* запуск нового потока на выполнение

1. **Что такое контек2ст потока и для чего он нужен?**

Это данные о потоке: программный код, набор регистров, стек памяти, оперативная память, стек ядра, маркер доступа, которые необходимы для диспетчеризации (например, возобновления работы потока) и синхронизации потоков

Эта структура с именем CONTEXT отражает состояние регистров процессора на момент последнего исполнения потока. Когда потоку выделяется процессорное время, система инициализирует регистры процессора содержимым контекста и, разумеется, регистр - - указатель команд идентифицирует адрес следующей машинной команды, необходимой для выполнения потока.

1. **Перечислите состояния в которых может быть поток и поясните их назначение.**

**Run, create, exit, interrupt** – прерывание по окончанию кванта, **block** – заблокировать до наступления события, **unblock** – ожидаемое событие наступило, **suspend** – приостановить поток, **resume** – возобновить поток, **sleep** – остановить поток на заданное время, **wakeup** – возобновить работу

1. **Что такое LWP?**

Это средство достижения многозадачности, представляющее отдельный слой между потоками ядра и пользовательскими потоками, работающее в пользовательском пространстве поверх одного потока выполнения ядра, , разделяет виртуальное адресное пространство и системные ресурсы потока выполнения с другими легковесными процессами, в рамках того же процесса. В некоторых операционных системах пользовательские потоки реализуются непосредственно потоками ядра, в таких случаях термин «легковесный процесс», как правило, означает поток ядра.

1. **Что такое потокобезопасность программного кода?**

свойство программного кода (программы) корректно работать в нескольких потоках одновременно

1. **Что такое реентерабельность кода?**

свойство одной копии программного кода работать в нескольких потоках одновременно

1. **Что такое Fiber?**

механизм для ручного планирования выполнения кода в рамках потока

1. **Дайте развернутое определение потока OS**.

объект ядра операционной системы, его наименьшая единица работы,являющаяся средством диспетчеризации доступа к процессорному времени, имеющая свой идентификатор, контекст, состояние.

1. **Поясните понятие «мультизадачная OS с вытеснением».**

Это операционная система, выполняющая несколько задач в один промежуток времени, которая выделяет этим задачам кванты времени и принимает решения о переключении процессора с выполнения одной задачи на другую по истечению определенного временного интервала.

1. **Поясните понятие «циклическое планирование».**

Это такой вид планирования, при котором каждому процессу назначается определенный интервал времени – квант, - в течение которого ему предоставляется возможность выполнения и по истечению которого ресурс ЦП будет отобран у него отобран и передан другому процессу, а сам процесс будет помещен в самый конец очереди процессов на выполнение.

1. **Поясните понятие «приоритетное планирование».**

Это такой вид планирования, при котором каждому процессу присваивается значение приоритетности и запускается тот процесс, который находится в состоянии готовности и имеет наивысший приоритет.

1. **Поясните понятие «кооперативное планирование».**

Это такой вид планирования, при котором процесс получает столько процессорного времени, сколько он считает нужным. Таким образом, все процессы делят процессорное время, периодически передавая управление следующей задаче.

1. **Поясните понятие «OS реального времени».**

Это операционная система, реагирующая в определенный как можно более короткий промежуток времени на непредсказуемое появление (внешних) событий.

1. **Поясните понятие «приоритет процесса».**

Это величина, определяющая, как часто данный процесс, по сравнению с другими процессами, стоящими в очереди на выполнение процессора, будет исполняться процессором.

1. **Поясните выражение «поток уступает процессор другому потоку».**

Это означает, что текущий поток прерывает свое выполнение, освобождая процессорное время другому потоку.

1. **Windows: как поток может уступить процессор?**

Sleep(0) - процесс становится в конец очереди, yield?

1. **Windows: что такое базовый приоритет потока, как он вычисляется и диапазон его изменения?**

Это приоритет потока, основанный на приоритете процесса и относительном приоритете потока. Он складывается на основании класса приоритета породившего этот поток процесса и относительного класса приоритета потока. Диапазон изменения от 1 до 31 включительно([1;31]), вычисляется по специальной таблице (таблицу надеюсь на листик переписывать по памяти не нужно будет...).



1. **Windows: поясните назначение и принцип применения системного вызова SetThreadIdealProcessor.**

SetThreadIdealProcessor устанавливает предпочтительный процессор для потока, т.е. процессор на котором он будет работать. Если функция завершается успешно, величина возвращаемого значения - предшествующий привилегированный процессор или MAXIMUM\_PROCESSORS, если поток не имеет такового. Если функция завершается с ошибкой, величина возвращаемого значения равна - (минус) 1.

1. **Windows: поясните назначение и принцип применения системного вызова ResumeThread.**

ResumeThread уменьшает счет времени приостановки работы потока, вызванной системным вызовом SuspendThread. Когда счет времени приостановки работы уменьшается до нуля, выполнение потока продолжается. ResumeThread проверяет счет времени приостановки работы подчиненного потока. Если счет времени приостановки работы равен 0, поток в настоящее время не приостановлен. Иначе, счет времени приостановки работы подчиненного потока уменьшается. Если итоговое значение - 0, то выполнение подчиненного потока продолжается.

1. **Windows: поясните назначение и принцип применения системного вызова WaitForSingleObject.**

Приостанавливает выполнение текущего потока до тех пор, пока объект, переданный этой функции, не перейдет в сигнальное состояние или не завершит свое выполнение, но не на больший интервал времени, чем тот, что передан в параметры этого вызова.ь

1. **Windows: поясните назначение и принцип применения системных вызовов GetProcessPriorityBoost, GetThreadPriorityBoost, SetProcessPriorityBoost, SetThreadPriorityBoost.**

GetProcessPriorityBoost, GetThreadPriorityBoost возвращает true/false в зависимости от того, разрешено ли динамическое изменение приоритета процесса/потока.

SetProcessPriorityBoost, SetThreadPriorityBoost задают это разрешение.

1. **Linux: поясните принцип идентификации процессов и потоков и поясните почему он такой.**

Номер процесса всегда равен номеру главного потока, потому что ядро Linux не знает ничего о потоках, для ядра поток – это процесс (LWP); для ядра нет разницы между двумя изолированными процессами и двумя потоками в рамках одного процесса; два потока в одном процессе для ядра – это два процесса использующих общие ресурсы ядра.

1. **Linux: Поясните понятие «планировщик потоков».**

Это часть ядра, алгоритм, который определяет порядок выполнения потоков. Планировщик потоков ядра отвечает за то, какие потоки выполняются на процессорах системы.

1. **Linux: поясните принцип использования значения nice –процесса, диапазон его изменения, для какого режима работы планировщика это значение применяется?**

Это значение любезности, чем оно меньше, тем выше приоритет у потока. Диапазон приоритетов - [-20, 19], значение по умолчанию = 0; Применяется для вытесняющего режима работы (other) (Планировщик может принудительно забирать управление у потока (например по таймеру или при появлении потока с большим приоритетом))

1. **Linux: перечислите политики планирования, какая действует по умолчанию?**

Политики планирования процесса:

1) стандартная (SCHED\_OTHER, с разделением времени для процессов, работающих не в реальном времени); - по умолчанию 2) FIFO-политика (SCHED\_FIFO, реального времени);

3) карусельная (Round-Robin) политика (SCHED\_RR) реального времени;

4) пакетная политика(BATCH).

1. **Linux: как выяснить действующую политику планирования для процесса с помощью файловой системы proc?**

/proc/PID/sched (в строчке police будет)

0- OTHER

1-FIFO

2-RR

1. **Linux: с помощью какого системного вызова поток может уступить процессор.**

Уступить процессор sched\_yield()

1. **Linux: чем отличается системный вызов nice от вызова setpriority.**

Nice прибавляет к текущему значению любезности nice аргумент, а setpriority устанавливает конкретное значение nice

1. **Linux: поясните понятие «планировщик ввода вывода», каким образом можно выяснить какие планировщики ввода/ вывода доступны?**

Это программная прослойка между блочными устройствами (дисковые устройства) и низкоуровневыми драйверами ввода/вывода. Задача планировщика – оптимизация доступа процесса к дисковому устройству(уменьшение времени поиска данных, обеспечение приоритетности, гарантировать данные за заданное время).

dmesg | grep scheduler

1. **Linux: перечислите известные вам планировщики ввода/ вывода, кратко охарактеризуйте их.**

NOOP – простой I/O-планировщик, общая FIFO- очередь read/write-запросов, объединяет однотипные запросы для сокращения операций.

CFQ (Completely Fair Queueing, наиболее справедливая очередь) – у каждого процесса своя очередь, у каждой очереди свой квант времени, планировщик циклически обходит очереди, обслуживает очередь в течении кванта, чтение в очереди имеет приоритет. Поддержка i/o-приоритетов.

BFQ (Budget Fair Queueing) – базируется на CFQ, каждой CFQ-очереди выделяется бюджет, который растет для процессов с интенсивным вводом/выводом.

Deadline - I/O-планировщик пытается выполнить запрос в указанное время, две очереди read и write, read-очередь приоритетнее, запросы объединяются в пакеты по принципу «алгоритма лифта».

MQ-Deadline – модификация Deadline для новых устройств.

Kyber - для работы с быстрыми устройствами, две очереди read и write, read-очередь приоритетнее измеряетcя время завершения каждого запроса и корректирует фактический размер очереди для достижения установленных в настройках задержек.

1. **Linux: каким образом можно выяснить тип планировщика действующего для блокового устройства?**

cat /sys/block/hda/queue/scheduler

1. **Поясните понятие «виртуальная память».**

Виртуальная память–метод управления памятью процессора, предназначенный для выполнения программ, которым выделяется адресное пространство превышающее доступный физический объем памяти компьютера.

1. **Поясните понятие «свопинг».**

Свопинг–аппаратно-программный механизм OS обмена (вытеснения и загрузки) содержимым блоков оперативной физической памяти компьютера с устройством хранения данных с целью расширения адресуемого объема оперативной памяти компьютера.

1. **Поясните понятие «страничная память».**

Страничная память – реализации виртуальной памяти, при которой физическая память и адресное пространство разбивается на блоки (страницы), а также осуществляется страничный свопинг. Размеры страниц для X86-64: 4K, 2MB, 1GB**.**

1. **Поясните понятие MMU.**

Memory Management Unit – диспетчер памяти**,** аппаратное (программируемое) устройство, входящее в состав процессора и предназначенное для трансляции виртуальных адресов оперативной памяти в реальные.

1. **Поясните понятие TLB.**

Translation Lookaside Buffer – буфер быстрого преобразования адреса, компонент MMU, предназначенный для вычисления реальных адресов.

1. **Какая информация содержится в строке таблицы страниц**

****

1. **Поясните принцип применения хэш-таблиц.**

Хэш-таблицы, созданные на основе виртуальных адресов, решают проблему инвертированной таблицы, которая усложняет преобразование виртуальных адресов в физические: все находящиеся на данный момент в памяти виртуальные страницы, имеющие одинаковые хэш-значения, связываются в одну цепочку. Как только будет найден номер страничного блока, в TLB будет введена новая пара значений (виртуального, физического)

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

Инвертированная таблица страниц – таблица для физических страниц, в которой имеется одна запись для каждого страничного блока в реальной памяти, а не одна запись на каждую страницу в виртуальном адресном пространстве. В каждой записи отслеживается, что именно находится в страничном блоке.

1. **Поясните понятие «рабочий набор страниц».**

Рабочий набор страниц - это набор страниц памяти, которые в настоящее время использует процесс (принадлежат вашему процессу и не выгружены).

1. **Поясните принцип работы алгоритма LRU.**

Least recently used - это алгоритм, при котором вытесняются кэшированные значения, которые дольше всего не запрашивались. Соответственно, необходимо хранить время последнего запроса к значению. И как только число закэшированных значений превосходит максимально возможный размер кэша, необходимо вытеснить из кеша значение, которое дольше всего не запрашивалось.

1. **Windows: поясните назначение сервиса SysMain.**

Сервис SysMain постоянно пребывает в фоновом режиме, анализируя модели использования ОЗУ и выясняя, какие страницы используются чаще всего и предварительно подкачивает их в сжатом виде, делая последующий доступ к ним намного быстрее.

1. **Windows: поясните назначение файла hiberfil.sys.**

Файл hiberfil.sys — это файл гибернации(сна), используемый в Windows для хранения данных и их последующей быстрой загрузки в оперативную память при включении компьютера или ноутбука.

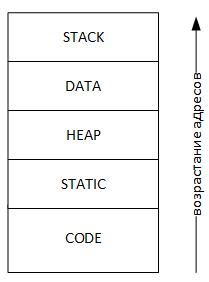
1. **Windows: поясните назначение файла pagefile.sys.**

pagefile.sys - это файл подкачки Windows.

1. **Windows: поясните назначение файла swapfile.sys.**

swapfile.sys - файл подкачки (в дополнение к pagefile.sys), но служащий исключительно для приложений из магазина приложений (UWP). В swapfile.sys записываются данные приложений из магазина, которые не требуются в настоящий момент времени, но могут внезапно потребоваться, и работает отличным от обычного файла подкачки Windows образом, представляя собой своего рода механизм «гибернации» для приложений.

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



Stack - для временного хранения данных программы, результатов промежуточных вычислений

Data – для постоянного хранения инициализированных данных программы

Heap – для хранения динамических данных программы

Static – для хранения статических данных программы

Code – для хранения скомпилированного кода программы

1. **Windows: какой стандартный начальный размер области heap?**

По умолчанию – 1MB, из них 4K сразу забирает процесс.

1. **Windows: каким образом можно изменить начальный размер области памяти heap приложения?**

Можно установить стартовое значение величины HEAP в параметрах Linker. В Visual Studio : Проект -> Свойства -> Свойства конфигурации -> Компоновщик -> Все параметры -> Резервный размер кучи.

1. **Windows: какой стандартный размер области памяти stack?**

1MB

1. **Windows: каким образом можно изменить размер области памяти stack приложения?**

в Visual Studio : Проект -> Свойства -> Свойства конфигурации -> Компоновщик -> Все параметры -> Резервный размер стека.

1. **Windows: поясните назначение функции Windows API: GlobalMemoryStatus.**

Получает информацию о текущем состоянии памяти компьютера, например, процент использования памяти, объем физической памяти, объем виртуальной памяти, размер файла подкачки и т.д.

1. **Windows: поясните назначение функции Windows API: VirtualQuery; перечислите значения атрибутов Protect, State и Type.**

Извлекает информацию о диапазоне страниц в виртуальном адресном пространстве вызывающего процесса.

****

****

****

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

Рабочее множество - количество памяти, требующееся процессу в заданный интервал времени.

Процесс устанавливает минимальный и максимальный размеры рабочего множества с помощью SetProcessWorkingSetSize.

Процесс удаляет как можно больше страниц из рабочего набора множества с помощью EmptyWorkingSet.

1. **Windows: что означает «страница заблокирована»? с помощью каких функций OS API можно установить блокировку страниц и снять блокировку? Какое максимальное количество страниц можно заблокировать?**

Если страница заблокирована, это означает, что остальным процессам запрещен доступ к этой странице, и последующий доступ к ней текущим процессом не приведет к ошибке. Можно установить/снять блокировку страниц с помощью функций VirtualLock и VirtualUnlock соответственно. Максимальное количество страниц, которые может заблокировать процесс, равно количеству страниц в его минимальном рабочем множестве за вычетом небольших накладных расходов.

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

Heap – область памяти адресного пространства, предназначенного для использования программной фрагментов динамически выделяемой памяти (malloc, new).

Heap процесса – куча, создаваемая в адресном пространстве процесса при его инициализации (ее размер по умолчанию — 1 Мб).

Пользовательская heap – куча, создаваемая пользователем.

При запуске [процесса](https://ru.wikipedia.org/wiki/%D0%9F%D1%80%D0%BE%D1%86%D0%B5%D1%81%D1%81_(%D0%B8%D0%BD%D1%84%D0%BE%D1%80%D0%BC%D0%B0%D1%82%D0%B8%D0%BA%D0%B0)) [ОС](https://ru.wikipedia.org/wiki/%D0%9E%D0%BF%D0%B5%D1%80%D0%B0%D1%86%D0%B8%D0%BE%D0%BD%D0%BD%D0%B0%D1%8F_%D1%81%D0%B8%D1%81%D1%82%D0%B5%D0%BC%D0%B0) выделяет [память](https://ru.wikipedia.org/wiki/%D0%9E%D0%BF%D0%B5%D1%80%D0%B0%D1%82%D0%B8%D0%B2%D0%BD%D0%B0%D1%8F_%D0%BF%D0%B0%D0%BC%D1%8F%D1%82%D1%8C) для размещения кучи. В дальнейшем [память](https://ru.wikipedia.org/wiki/%D0%9E%D0%BF%D0%B5%D1%80%D0%B0%D1%82%D0%B8%D0%B2%D0%BD%D0%B0%D1%8F_%D0%BF%D0%B0%D0%BC%D1%8F%D1%82%D1%8C) для кучи (под кучу) может выделяться динамически. [Память](https://ru.wikipedia.org/wiki/%D0%9E%D0%BF%D0%B5%D1%80%D0%B0%D1%82%D0%B8%D0%B2%D0%BD%D0%B0%D1%8F_%D0%BF%D0%B0%D0%BC%D1%8F%D1%82%D1%8C) кучи можно разделить на **занятую** (выделенную программе с помощью [функций](https://ru.wikipedia.org/wiki/%D0%A4%D1%83%D0%BD%D0%BA%D1%86%D0%B8%D1%8F_(%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D0%B5)), подобных malloc()) и **свободную** (ещё не занятую или уже освобождённую с помощью [функций](https://ru.wikipedia.org/wiki/%D0%A4%D1%83%D0%BD%D0%BA%D1%86%D0%B8%D1%8F_(%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D0%B5)), подобных free()). Для хранения данных о том, какая область кучи является занятой, а какая — свободной, обычно используется дополнительная область памяти. Перед началом работы программы выполняется инициализация кучи, в ходе которой [память](https://ru.wikipedia.org/wiki/%D0%9E%D0%BF%D0%B5%D1%80%D0%B0%D1%82%D0%B8%D0%B2%D0%BD%D0%B0%D1%8F_%D0%BF%D0%B0%D0%BC%D1%8F%D1%82%D1%8C), выделенная под кучу, отмечается как свободная.

1. **Linux: в какой части адресного пространства выделяется памяти с помощью функций malloc, calloc?**

В heap.

**1. Дайте определение понятию «синхронизация потоков».**

Синхронизация потоков – механизм упорядочивания выполнения программных блоков двух или более потоков, позволяющий потокам согласовывать свою работу с общими ресурсами.

**2. Объясните понятие «взаимная блокировка».**

Это ситуация, при которой несколько процессов находятся в состоянии ожидания ресурсов, занятых друг другом, и ни один из них не может продолжать свое выполнение.

**Взаимная блокировка** – это ситуация в которой, два или более процесса(потока) занимая некоторые ресурсы, пытаются заполучить некоторые другие ресурсы, занятые другими процессами/потоками и ни один из процессов/потоков не может занять необходимый им ресурс, и соответственно освободить занимаемый. (<https://javarush.ru/groups/posts/1422-vzaimnaja-blokirovkadeadlock-v-java-i-metodih-borjhbih-s-ney>)

**Взаимная блокировка** (deadlock) - явление при котором все потоки находятся в режиме ожидания. Происходит, когда достигаются состояния:

· взаимного исключения: по крайней мере один ресурс занят в режиме неделимости и, следовательно, только один поток может использовать ресурс в любой данный момент времени.

· удержания и ожидания: поток удерживает как минимум один ресурс и запрашивает дополнительные ресурсов, которые удерживаются другими потоками.

· отсутствия предочистки: операционная система не переназначивает ресурсы: если они уже заняты, они должны отдаваться удерживающим потокам сразу же.

· цикличного ожидания: поток ждёт освобождения ресурса другим потоком, который в свою очередь ждёт освобождения ресурса заблокированного первым потоком.

1. **Перечислите механизмы авторизации(синхранизации) OS.**

* Critical section ;
* Mutex;
* Semaphore;
* Atomic operation (interlocking function)
* Event;
* Waitable timer.

**4. Поясните в чем разница между механизмом mutex и semaphore.**

Mutex – механизм синхронизации нескольких потоков разных процессов, являющийся объектом ядра OS.

Семафоры — это объекты режима ядра, поэтому они имеют дескрипторы безопасности и описатели.

Мьютексы — это тоже объекты режима ядра, используемые для синхронизации, но они проще семафоров, поскольку не имеют счетчиков ( мьютекс объекта может захватить одновременно только один поток). (Книга)

В основе семафора лежит счётчик, над которым можно производить две [атомарные операции](https://ru.wikipedia.org/wiki/%D0%90%D1%82%D0%BE%D0%BC%D0%B0%D1%80%D0%BD%D0%B0%D1%8F_%D0%BE%D0%BF%D0%B5%D1%80%D0%B0%D1%86%D0%B8%D1%8F): увеличение и уменьшение значения на единицу, а мьютекс можно представить в виде переменной, которая может находиться в двух состояниях: в заблокированном и в незаблокированном. При входе в свою критическую секцию поток вызывает функцию перевода мьютекса в заблокированное состояние. При выходе из критической секции поток вызывает функцию перевода мьютекса в незаблокированное состояние. Соответственно, мьютекс может захватить одновременно только один поток, а семафор – несколько.

**5. Почему mutex, semaphore, event создают объект ядра OS, а critical section нет.**

Critical section – механизм синхронизации нескольких потоков **одного** процесса, НЕ ЯВЛЯЕТСЯ объектом ядра OS. (Лекция)

Объекты принадлежат ядру, а не процессу. Другими словами, если процесс создает какой-либо объект ядра, а затем завершает свою работу, то объект ядра может быть и не разрушен

не требует привлечения ядра Windows

1. **Поясните понятие «социальное время» и почему оно не монотонное?**

Социальное время это отсчет времени принятый в обществе

Социальное время не монотонное: так как каждый солнечный год увеличивается на 3 мс, люди измеряют время не точно, поэтому время от времени делались коррекции.

1. **Поясните понятие «эпоха Linux», назовите стартовую дату «эпохи Linux» и в каких единицах изменяется время?**

эпоха Linux - это система для описания момента времени

с полуночи (00:00:00 UTC) 1 января 1970 года (четверг); этот момент называют «эпохой Unix» (англ. Unix Epoch).

эпоха Unix (POSIX-время) c 01.01.1970 0:00:00 в секундах(из лекции).

1. **Поясните понятие «Coordinated Universal Time (UTC)».**

стандарт, по которому общество регулирует часы и время, отличается на целое количество секунд от атомного времени.

универсальное согласованное время (на Гринвичском меридиане, раньше GMT – Greenwich Meridian Time), усредненное значение, полученное на основе данных 50 лабораторий, оборудованных атомными часами (цезий-133), расхождение с солнечными часами примерно 3мс (атомные часы отстают) в сутки, коррекция при ошибке в 800 мс.

1. **Поясните понятия «относительное время» и «абсолютное время».**

**Относительное** время — это время протекания одного действия по отношению ко времени другого действия.

**Абсолютное** время - это конкретное время. 17:32 17 декабря 2020

1. **Поясните понятие «тик».**

Неофициальная единица измерения времени,

равна продолжительности одного импульса тактового генератора (часов).

1. **Поясните понятие «ожидающий таймер», перечислите типы таймеров, перечислите состояния, в которых может находится таймер.**

Ожидающие таймеры - объекты ядра, которые предназначены для отсчета промежутков времени, используется для синхронизации.

объект синхронизации,

два состояния: сигнальное – наступление заданного момента времени; несигнальное (активное и пассивное состояние) - ждет наступления заданного момента времени.

типы: с автоматическим или ручным сбросом.



1. **Перечислите типы часов, используемых в Linux, поясните их назначение.**

REALTIME – системное время(настенное),

MONOTONIC – с начала загрузки OS(монотонно возрастает), PROCESS – процессорное время( затраченное процессом),

THREAD – процессорное время(затраченное потоком).

1. **Поясните назначение констант HZ, CLOCKS\_PER\_SEC.**

HZ – частота системного таймера (обычно, 100,250, x86 сейчас **1000**), параметр ядра. Это значит, что прерывание таймера возникает HZ раз в секунду.

CLOCKS\_PER\_SEC - число, обозначающее количество тиков в секунду. CLOCKS\_PER\_SEC = 1000. Для каждой системы это число различно.

clock()/CLOCKS\_PER\_SEC = количество секунд.