1. **Что такое POSIX?**

Набор стандартов, описывающее интерфейс взаимодействия прикладной программы и ОС.

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?**

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

программные (частный случай внутреннего прерывания) — инициируются исполнением специальной [инструкции](https://ru.wikipedia.org/wiki/%D0%9A%D0%BE%D0%B4_%D0%BE%D0%BF%D0%B5%D1%80%D0%B0%D1%86%D0%B8%D0%B8) в коде [программы](https://ru.wikipedia.org/wiki/%D0%9A%D0%BE%D0%BC%D0%BF%D1%8C%D1%8E%D1%82%D0%B5%D1%80%D0%BD%D0%B0%D1%8F_%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D0%B0), предназначены для выполнения некоторых действий операционной системы, являются синхронными

1. **Что такое процесс?**

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

1. **Что такое контекст процесса?**

Контекст процесса – данные, которые сохраняются при переключении процессов и предназначенные для продолжения работы; (адресное пространство, содержимое регистров, объекты ядра OS (объекты процессов, потоков, безопасности, файлов и пр.) стек ядра (для этого процесса))

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

Системный вызов *fork* создает точную копию исходного процесса, называемого **родительским процессом (parent process)**. Новый процесс называется **дочерним процессом (child process)**. У родительского и у дочернего процессов есть свои собственные (приватные) образы памяти. Если родительский процесс впоследствии изменяет какие-либо свои переменные, то эти изменения остаются невидимыми для дочернего процесса (и наоборот).

Функция exec() (execute) загружает и запускает другую программу. Таким образом, новая программа полностью замещает текущий процесс.

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

Это процесс прародитель для всех процессов

Тип начальный процесс, который является родительским для остальным

1. **Перечислите области памяти процесса и поясните их назначение.**



сегмент кода – содержит машинные команды, Адресуется регистром CS;

сегмент данных – содержит данные, то есть константы и рабочие области, необходимые программе. Адресуется регистром DS;(иниц переменные)

сегмент стека – содержит адреса возврата в точку вызова подпрограмм. Адресуется регистром SS.

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

**системные** **процессы:** процессы запускаемые автоматически при запуске OS; Windows: windows-сервисы; Linux-демоны.

процесс операционной системы управляет каким-либо ресурсом в интересах прикладного процесса.

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

 приложение, автоматически (если настроено) исполняемое системой при запуске операционной системы **Windows** и выполняющиеся вне зависимости от статуса пользователя.

Linux-daemons: systemd – демон инициализации, systemctl – правление демонами (запуск,остановка,… ).

Де́мон — компьютерная программа в UNIX-подобных системах, запускаемая самой системой и работающая в фоновом режиме без прямого взаимодействия с пользователем.

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

Функция **CreateProcess**создает новый процесс и его главный поток. Новый процесс выполняет указанный исполняемый файл. Функция CreateProcess предназначена для запуска новой программы. Функции WinExec и LoadModule также работают, но не предоставляют таких возможностей, как CreateProcess.

Функция **LoadModule** загружает и исполняет приложение или создает новый экземпляр существующего приложения.

Самый простой способ запустить какую-то программу из своего приложе­ния — использовать функцию **WinExec.**

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

С помощью функции System(). Этот способ сравнительно прост, однако неэффективен и создает определенные риски с точки зрения безопасности.

С помощью функций fork() и exec() - более продвинутая техника с точки зрения гибкости, скорости и безопасности.

Функция exec() (execute) загружает и запускает другую программу. Таким образом, новая программа полностью замещает текущий процесс.

 При вызове fork() порождается новый процесс (процесс-потомок), который почти идентичен порождающему процессу-родителю.

1. **Какие потоки данных доступны любому процессу автоматически?**

**стандартные потоки ввода/вывода процесса**: потоки имеющие зарезервированные номера - дескрипторы (номера), поток ввода (0), поток вывода (1), поток вывода ошибок (2).

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

Вызов *WaitForSingleObject* используется для ожидания события. Ожидание может касаться множества возможных событий. Если в параметре указан процесс, то вызывающая программа дожидается окончания конкретного процесса.

Ожидает, пока указанный объект не перейдет в сигнальное состояние или пока не истечет интервал времени ожидания.

DWORD WaitForSingleObject(

[in] HANDLE hHandle,

[in] DWORD dwMilliseconds

);

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

* создается ядром OS по системному вызову;
* адресное пространство (данные, программа, стек, куча);
* ресурсы: регистры, открытые файлы, родительский процесс, перечень связанных (дочерних) процессов, реальные страницы памяти, виртуальное адресное пространство, маркеры доступа (безопасность);
* процесс может создавать (с помощью системного вызова) дочерние процессы, в общем случае может образовываться дерево процессов;
* обычно соответствует работающей программе (например, exe-файлу в Windows);
* OS хранит список (или таблицу) объектов работающих процессов;
* при приостановке процесса в объекте процесса сохраняется вся информация (регистры, уведомления OS,… - контекст процесса), позволяющая возобновить работу процесса;
* процессы изолированы друг от друга;
* для обмена данными между процессами, применяется специальный механизм – механизм межпроцессного взаимодействия (IPC);