1. **Что такое поток управления OS?**

**поток (управления) OS** – объект ядра операционной системы, которому OS выделяет процессорное время.

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

CreateThread в Windows

pthread\_create в Linux

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

**Поток пользовательского уровня (user thread)**- высокоуровневый *поток*, *операции* над которым включены в *интерфейс* пользователя ОС.

**Поток ядра (kernel thread)**- низкоуровневый системный *поток*, поддержанный и использующийся на уровне ядра операционной системы; используется для реализации **потоков пользовательского уровня**.

1. **Что такое многопоточность?**

многопоточность – модель (парадигма) программирования (OS, системы программирования, программы) поддерживать потоки управления.

1. Чем отличаются приоритетная многопоточность от и кооперативной многопоточности?
2. **Что такое диспетчеризация потоков управления OS?**

*Диспетчеризация* – алгоритм, устанавливающий порядок обслуживания очередей потоков процессорами.

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

контекст потока – данные необходимые для возобновления работы потока при его приостановке (диспетчеризация, синхронизация): программный код, набор регистров, стек памяти, оперативная память, стек ядра, маркер доступа);

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

состояния потока: исполняется, готов к исполнению, блокирован, спит; приостановлен;

1. **Что такое LWP?**

LWP - light-weight process

Легковесным процессом является процесс, поддерживающий работу потока пространства пользователя. В Linux создание легковесных процессов осуществляется при помощи нестандартизированного системного вызова clone().

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

**Потокобезопасность кода (программы) –** свойство программного кода (программы) корректно работать в нескольких потоках одновременно.

1. **Что такое реентерабельность кода?**

**Реентерабельность кода (программы)** – свойство одной копии программного кода работать в нескольких потоках одновременно. Реентерабельный код всегда потокобезопасен. Реентерабельный код не использует статическую память и не изменяет сам себя, все данные сохраняются в динамической памяти.

1. **Что такое Fiber?**

фибра – механизм для ручного планирования выполнения кода в рамках потока.

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

поток – средство диспетчеризации доступа к процессорному времени (квант примерно 20мс);

поток – последовательность команд процессора;

поток – наименьшая единица работы ядра OS;

Функция **CreateThread** создает поток, который выполняется в пределах виртуального адресного пространства вызывающего процесса.

**HANDLE** **CreateThread**(

**LPSECURITY\_ATTRIBUTES** *lpThreadAttributes*, // дескриптор защиты **SIZE\_T** *dwStackSize*,                       // начальный размер стека **LPTHREAD\_START\_ROUTINE** *lpStartAddress*,    // функция потока **LPVOID** *lpParameter*,                       // параметр потока **DWORD** *dwCreationFlags*,                    // опции создания **LPDWORD** *lpThreadId*                        // идентификатор потока

);

**Параметры**

***lpThreadAttributes***

[in] Указатель на структуру **SECURITY\_ATTRIBUTES**, которая обуславливает, может ли возвращенный дескриптор быть унаследован дочерними процессами. Если ***lpThreadAttributes*** является значением ПУСТО (NULL), дескриптор не может быть унаследован.

***dwStackSize***

[in] Начальный размер стека, в байтах. Система округляет это значение до самой близкой страницы памяти. Если это значение нулевое, новый поток использует по умолчанию размер стека исполняемой программы. Дополнительную информацию см. в статье [**Размер стека потока**](http://narovol.narod.ru/_tbkp/New_MSDN_API/size_stack_thread.htm)

***lpStartAddress***

[in] Указатель на определяемую программой функцию типа **LPTHREAD\_START\_ROUTINE**, код которой исполняется потоком и обозначает начальный адрес потока. Для получения дополнительной информации о функции потока, см. **[ThreadProc](http://narovol.narod.ru/_tbkp/New_MSDN_API/fn_threadproc.htm)**.

***lpParameter***

[in] Указатель на переменную, которая передается в поток.

***dwCreationFlags***

[in] Флажки, которые управляют созданием потока. Если установлен флажок **CREATE\_SUSPENDED**, создается поток в состоянии ожидания и не запускается до тех пор, пока не будет вызвана функция **[ResumeThread](http://narovol.narod.ru/_tbkp/New_MSDN_API/fn_resumethread.htm)**. Если это значение нулевое, поток запускается немедленно после создания. В это время, никакие другие значения не поддерживаются.

***lpThreadId***

[out] Указатель на переменную, которая принимает идентификатор потока.

**Возвращаемые значения**

Если функция завершается успешно, величина возвращаемого значения - дескриптор нового потока.

Если функция завершается с ошибкой, величина возвращаемого значения - ПУСТО (NULL). Чтобы получать дополнительные данные об ошибках, вызовите **[GetLastError](http://narovol.narod.ru/_tbkp/New_MSDN_API/Debbag_error/fn_getlasterror.htm)**[.](http://narovol.narod.ru/_tbkp/New_MSDN_API/Debbag_error/fn_getlasterror.htm)

Функция **TerminateThread** завершает работу потока.

**BOOL** **TerminateThread**(

**HANDLE** *hThread*,  // дескриптор потока

**DWORD** *dwExitCode* // код завершения для потока

);

**Параметры**

***hThread***

[in/out] Дескриптор потока, который завершает работу.

***dwExitCode***

[in] Код завершения работы потока. Используйте функцию **[GetExitCodeThread](http://www.vsokovikov.narod.ru/New_MSDN_API/Process_thread/fn_getexitcodethread.htm)**, чтобы извлечь значение выхода потока.

**Возвращаемые значения**

Если функция завершается успешно, величина возвращаемого значения - не ноль.

Если функция завершается с ошибкой, величина возвращаемого значения - ноль. Чтобы получить дополнительные данные об ошибках, вызовите [**GetLastError**](http://www.vsokovikov.narod.ru/New_MSDN_API/Debbag_error/fn_getlasterror.htm).

Функция **SuspendThread** приостанавливает работу заданного потока.

**DWORD** **SuspendThread**(

**HANDLE** *hThread* // дескриптор потока

);

**Параметры**

***hThread***

[in] Дескриптор потока, работа которого приостанавливается.

**Возвращаемые значения**

Если функция завершается успешно, величина возвращаемого значения - счет времени предыдущей приостановки работы потока; иначе, оно равно **- (минус) 1**. Чтобы получить дополнительные данные об ошибках, используйте функцию **[GetLastError](http://www.vsokovikov.narod.ru/New_MSDN_API/Debbag_error/fn_getlasterror.htm)**.

Функция **ResumeThread** уменьшает счет времени приостановки работы потока. Когда счет времени приостановки работы уменьшается до нуля, выполнение потока продолжается.

**DWORD** **ResumeThread**(

**HANDLE** *hThread*     // дескриптор потока

);

**Параметры**

***hThread***

[in] Дескриптор для потока, который будет перезагружен.

**Возвращаемые значения**

Если функция завершается успешно, величина возвращаемого значения - предшествующий счет времени приостановки работы потока.

Если функция завершается с ошибкой, величина возвращаемого значения равна **- (минус) 1**. Чтобы получить дополнительные данные об ошибках, вызовите **[GetLastError](http://www.vsokovikov.narod.ru/New_MSDN_API/Debbag_error/fn_getlasterror.htm)**.

**DWORD** — 32-битное беззнаковое целое.

Создание потока происходит с помощью функции pthread\_create(pthread\_t \*tid, const pthread\_attr\_t \*attr, void\*(\*function)(void\*), void\* arg), где: tid - идентификатор потока, attr - параметры потока (NULL - атрибуты по умолчанию, подробности в man), function - указатель на потоковую функцию, в нашем случае threadFunc и arg - указатель на передаваемые данные в поток.

Функция pthread\_join ожидает завершения потока thread. Второй параметр этой функции - результат, возвращаемый потоком.