**Дайте определение понятию "процесс" и "поток". Дайте определение понятию “метод”. В каких состояниях может пребывать поток?**

**Дайте определение понятию “процесс”.**

Процесс — это совокупность кода и данных, разделяющих общее виртуальное адресное пространство. Процессы изолированы друг от друга, поэтому прямой доступ к памяти чужого процесса невозможен (взаимодействие между процессами осуществляется с помощью специальных средств). Для каждого процесса ОС создает так называемое «виртуальное адресное пространство», к которому процесс имеет прямой доступ. Это пространство принадлежит процессу, содержит только его данные и находится в полном его распоряжении. Операционная система же отвечает за то, как виртуальное пространство процесса проецируется на физическую память.

**Дайте определение понятию “поток”.**

Один поток («нить» или «трэд») – это одна единица исполнения кода. Каждый поток последовательно выполняет инструкции процесса, которому он принадлежит, параллельно с другими потоками этого процесса.

**Дайте определение понятию “метод”.**

Метод - это функция или процедура, принадлежащая какому-то классу или объекту.

Как и процедура в процедурном программировании, метод состоит из некоторого количества операторов для выполнения какого-то действия и имеет набор входных аргументов.

**В каких состояниях может пребывать поток?**

**Ready (готов)** – находящийся в пуле (pool) потоков, ожидающих выполнения;

**Running (выполнение)** - выполняющийся на процессоре;

**Waiting (ожидание)**, также называется idle или suspended, приостановленный - в состоянии ожидания, которое завершается тем, что поток начинает выполняться (состояние Running) или переходит в состояние Ready;

**Terminated (завершение)** - завершено выполнение всех команд потока. Впоследствии его можно удалить. Если поток не удален, система может вновь установить его в исходное состояние для последующего использования.

 Поток находится в состоянии выполнения (running state), если она фактически выполняется процессором. В SMP-системах в состоянии выполнения могут находиться одновременно несколько потоков.

• Планировщик переводит поток в состояние ожидания (wait state), если он выполняет функцию ожидания несигнализирующих объектов, например, потоков или процессов, или перехода в сигнальное состояние объектов синхронизации, о чем говорится в главе 8. Операции ввода/вывода также будут ожидать завершения передачи дисковых или иных данных, но ожидание может быть вызвано и другими многочисленными функциями. О потоках, находящихся в состоянии ожидания, нередко говорят как о блокированных(blocked) или спящих (sleeping).

• Поток находится в состоянии готовности (ready state), если она может выполняться. Планировщик в любой момент может перевести такой поток в состояние выполнения. Когда процессор станет доступным, планировщик запустит тот из потоков, находящихся в состоянии готовности, который обладает наивысшим приоритетом, а при наличии нескольких потоков с одинаковым высшим приоритетом запущен будет та, который пребывал в состоянии готовности дольше всех. При этом поток проходит через состояние простоя (standby state), или резервное состояние.

• Обычно, в соответствии с приведенным описанием, планировщик помещает поток, находящийся в состоянии готовности, на любой доступный процессор. Программист может указать маску родства процессоров (processor affinity mask) для потока (см. главу 9), предоставляя потоку процессоры, на которых он может выполняться. Используя этот способ, программист может распределять процессоры между потоками. Соответствующими функциями являются SetProcessorAffinityMask и GetProcessorAffinityMask. Функция SetThreadIdealProcessor позволяет указать предпочтительный процессор, подлежащий использованию планировщиком при первой же возможности.

• После истечения кванта времени, отведенного выполняющемуся потоку, планировщик без ожидания переводит его в состояние готовности. В результате выполнения функции Sleep(0) поток также будет переведен из состояния выполнения в состояние готовности.

• Планировщик переводит ожидающий поток в состояние готовности сразу же, как только соответствующие дескрипторы оказываются в сигнальном состоянии, хотя при этом поток фактически проходит через промежуточное переходное состояние (transition state). В подобных случаях принято говорить о том, что поток пробуждается (wakes).

• Не существует способа, позволяющего программе определить состояние другого потока (разумеется, если поток выполняется, то он находится в состоянии выполнения, и поэтому ему нет никакого смысла определять свое состояние). Даже если бы такой способ и существовал, то состояние потока может измениться еще до того, как опрашивающий поток успеет предпринять какие-либо действия в ответ на полученную информацию.

• Поток, независимо от его состояния, может быть приостановлен (suspended), и приостановленный поток не будет запущен, даже если он находится в состоянии готовности. В случае приостановки выполняющегося потока, независимо от того, по собственной ли инициативе или по инициативе потока, выполняющегося на другом процессоре, он переводится в состояние готовности.

• Поток переходит в состояние завершения (terminated state) тогда, когда его выполнение завершается, и остается в этом состоянии до тех пор, пока остается открытым хотя бы один из ее дескрипторов. Это позволяет другим потокам запрашивать состояние данного потока и его код завершения.