**1. Что такое процесс, домен, поток? Как они связаны между собой?**

Процесс – набор ресурсов, используемый отдельным экземпляром приложения.

Домен приложения — это механизм, реализованный в .NET, который позволяет запустить группу приложений в одном процессе, обеспечивая относительную изоляцию их друг от друга, в то же время позволяя им взаимодействовать друг с другом значительно быстрее, чем в случае отдельных процессов.

Поток – используемый внутри процесса путь выполнения.

**2.Как получить информацию о процессах?**

var allProcess = Process.GetProcesses() ;

CurrentPriority

Id StartAddress

StartTime

**3. Как создать и настроить домен?**

AppDomain newD = AppDomain.CreateDomain("New");

newD.Load("имя сборки");

AppDomain.Unload(newD);

**4. Как создать и настроить поток?**

public Thread(ThreadStart start);

public Thread(ParameterizedThreadStart start);

public Thread(ThreadStart start, int maxStackSize);

Thread th = new Thread((new Point()).Move);

th.Start();

Thread thrd = new Thread((new Point()).Move)

{ Name = "Point Move",

Priority = ThreadPriority.BelowNormal,

IsBackground = true, };

**5. В каких состояниях может быть поток?**

Перечисление ThreadState – состояние потока:

•Aborted: поток остановлен, но пока еще окончательно не завершен

•AbortRequested: для потока вызван метод Abort, но остановка потока еще не произошла •Background: поток выполняется в фоновом режиме

•Running: поток запущен и работает (не приостановлен)

•Stopped: поток завершен

•StopRequested: поток получил запрос на остановку

•Suspended: поток приостановлен

•SuspendRequested: поток получил запрос на приостановку

•Unstarted: поток еще не был запущен

•WaitSleepJoin: поток заблокирован в результате действия методов Sleep или Join

**6. Какие методы управления потоками вы знаете, для чего и как их использовать?**

Методы класса Thread:

> GetDomain - статический, возвращает ссылку домен приложения

> GetDomainId - статический, возвращает id домена приложения, в котором выполняется текущий поток

> Sleep – статический, останавливает поток на определенное количество миллисекунд

> Abort - уведомляет среду CLR о том, что надо прекратить поток (происходит не сразу)

> Interrupt - прерывает поток на некоторое время

> Join - блокирует выполнение вызвавшего его потока до тех пор, пока не завершится поток, для которого был вызван данный метод

> Resume - возобновляет работу приостановленного потока

> Start - запускает поток

> Suspend - приостанавливает поток

> Yield - передаёт управление следующему ожидающему потоку системы

**7. Какие приоритеты потока вы знаете?**

Приоритеты - перечисление ThreadPriority:

? Lowest

? BelowNormal

? Normal (по умолчанию)

? AboveNormal

? Highest

**8. Что такое пул потоков и для чего он используется?**

Класс System.Threading.ThreadPool обеспечивает приложение пулом рабочих потоков, управляемых системой, позволяя пользователю сосредоточиться на выполнении задач приложения, а не на управлении потоками. Если имеются небольшие задачи, которые требуют фоновой обработки, пул управляемых потоков — это самый простой способ воспользоваться преимуществами нескольких потоков.

Потоки из пула являются фоновыми. Для каждого потока используется размер стека по умолчанию, поток запускается с приоритетом по умолчанию и находится в многопотоковом подразделении. Когда поток в пуле завершает свою задачу, он возвращается в очередь потоков в состоянии ожидания. С этого момента его можно использовать вновь. Повторное использование позволяет приложениям избежать дополнительных затрат на создание новых потоков для каждой задачи.

Для каждого процесса существует только один пул потоков.

**9. Что такое критическая секция? Поясните использование.**

Критическая секция — участок исполняемого кода программы, в котором производится доступ к общему ресурсу (данным или устройству), который не должен быть одновременно использован более чем одним потоком исполнения. При нахождении в критической секции двух (или более) потоков возникает состояние «гонки» («состязания»).

**10. Что такое мьютекс? Поясните использование**

Еще один инструмент управления синхронизацией потоков представляет класс Mutex, также находящийся в пространстве имен System.Threading. Данный класс является классом-оболочкой над соответствующим объектом ОС Windows "мьютекс".

**11. Что такое семафор? Поясните использование**

Еще один инструмент, который предлагает нам платформа .NET для управления синхронизацией, представляют семафоры. Семафоры позволяют ограничить доступ определенным количеством объектов.

**12. Что такое неблокирующие средства синхронизации?**

Неблокирующая синхронизация — подход в параллельном программировании на симметрично-многопроцессорных системах, в котором принят отказ от традиционных примитивов блокировки, таких, как семафоры, мьютексы и события. Разделение доступа между потоками идёт за счёт атомарных операций и специальных, разработанных под конкретную задачу, механизмов блокировки.

Преимущество неблокирующих алгоритмов — в лучшей масштабируемости по количеству процессоров. К тому же, если ОС прервёт один из потоков фоновой задачей, остальные, как минимум, выполнят свою работу, не простаивая. Как максимум — возьмут невыполненную работу на себя.

**13. Для чего можно использовать класс Timer?**

Создает событие через заданный интервал с возможностью генерировать повторяющиеся события.