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

При запуске приложения операционная система создает для него отдельный процесс,

которому выделяется определённое адресное пространство в памяти и который изолирован от других процессов.

Процесс может иметь несколько потоков. Как минимум, процесс содержит один - главный поток.

В приложении на C# точкой входа в программу является метод Main.

Вызов этого метода автоматически создает главный поток. А из главного потока могут запускаться вторичные потоки.

В .NET исполняемые файлы не обслуживаются прямо внутри процесса Windows.

ОНИ обслуживаются в отдельном логическом разделе внутри процесса, который называется доменом приложения (Application Domain — AppDomain).

В процессе может содержаться несколько доменов приложений.

С точки зрения архитектуры отладчика поток — это базовые единицы вычислений.

Поток последовательно выполняет свои инструкции в контексте одного вызова стека,

перемещение из одной кодовой контекста в другую.

2. Как получить информацию о процессах?

Process[] procList = Process.GetProcesses(); + foreach

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(ThreadStart start, int maxStackSize);

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

th.Start();

5. В каких состояниях может быть поток?

Перечисление ThreadState:

•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?

► 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?

• Lowest

• BelowNormal

• Normal (по умолчанию)

• AboveNormal

• Highest

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

Для уменьшения издержек, связанных с созданием потоков, платформа .NET поддерживает специальный механизм, называемый пул потоков.

Пул состоит из двух основных элементов: очереди методов и рабочих потоков.

9. Что такое критическая секция? Поясните использование.

Критические секции -- это объекты, используемые для блокировки доступа всех нитей (threads) приложения, кроме одной, к некоторым важным данным в один момент времени.

10. Что такое мьютекс? Поясните использование

System.Threading.Mutex позволяет организовать критическую секцию для нескольких процессов

► WaitOne() - входа в критическую секцию,

►ReleaseMutex() – для выхода из неё (выход может быть произведён только в том же потоке выполнения, что и вход).

11. Что такое семафор? Поясните использование

►объект синхронизации, позволяющий войти в заданный участок кода не более чем N потокам (N – ёмкость семафора)

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.

Неблокирующие средства синхронизации позволяют осуществлять совместный доступ к простым ресурсам нескольких потоков без блокировки, паузы или ожидания.

13. Для чего можно использовать класс Timer?

Позволяет запускать определенные действия по истечению некоторого периода времени.