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

**Процесс** – объект, кот. создается ОС для каждого приложения в момент его запуска  
– выделяет для приложухи изолир. адресное пр-во и поддерж. 1/неск потоков выполнения  
1) для каждого .exe созд. отд. изол процесс  
2) не влияют др. на др  
3) доступ из одного в др. невозм.  
4) каждый имеет. PID (Process ID)

**Домен** – отд. логический раздел внутри процесса, в кот. обслуж. исп.файлы  
(мб неск. в процессе)  
1) сущ. внутри процессов  
2) сод. загруженные сборки  
3) домен по умолч. (AppDomain.CurrentDomain) запускает процесс при старте  
4) обесп. уровень изоляции кода

**Поток** (Thread) – посл-сть выполняемых команд процессора  
 – низкоуровневый инструмент для организации параллельной работы

2. Как получить информацию о процессах?

var allProcess = Process.GetProcesses()  
( CurrentPriority, Id, StartAddress, StartTime… )

3. Как создать и настроить домен?

созд. и уничт. в ходе работы в рамках процесса

AppDomain newD = AppDomain.CreateDomain(“NEW”);  
newD.Load(“Имя сборки”);  
AppDomain.Unload(newD);

4. Как создать и настроить поток?

public Thread(ThreadStart start); //делегат, с методов для вып. в потоке  
public Thread(ParameterizedStart start); //при запуске метода перед. ему д-е в виде объ  
public Thread(ThreadStart start, int maxStackSize);

Thread th = new Thread((new Point()).Move);  
th.Start(); //запуск потока

5. В каких состояниях может быть поток?

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

Методы:  
GetDomain стат., возвр. ссылку домен  
GetDomainId стат., возвр. id домена, в кот. вып. тек. поток  
Sleep стат., остан. поток на нек t  
Abort прекратить поток (не сразу)  
Interrupt прерывает поток на нек t  
Join блок вып вызвавшего потока, пока не заверш. поток, для кот. вызв. метод  
Resume возобн работу приостановленного потока  
Start запуска поток  
Suspend приостан поток  
Yield передает упр след ожидающему потоку

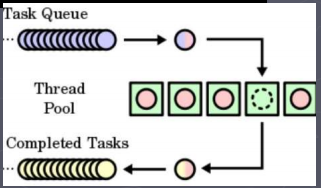
**класс Thread:**

Context ctx = Thread.CurrentContext; //получ. контекст, в кот. вып. поток  
var curt = Thread.CurrentThread; //получ. ссылклу на вып. поток  
Console.WriteLine(“ “+curt.Name); //имя потока  
if (currt.IsAlive); //раб. ли поток в тек. момент  
if (!currt.IsBackground); //явл. ли фоновым

7. Какие приоритеты потока вы знаете?

**Перечисление ThreadPriority:**  
Lowest  
BelowNormal  
Normal (по умолч.)  
AboveNormal  
Highest

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

**Пул потоков** – спец. механизм, для уменьш. издержек, связанных с созд. потоков  
сост. из:  
\* очередь методов  
\* рабочие потоки  


**класс ThreadPool:**  
\* SetMaxThreads() – изм. емкость пула (макс число рабочих потоков)  
\* SetMinThreads() – уст. кол-во раб. потоков, созд. без задержки  
\* QueueUserWorkItem() – помещение метода в очередь пула

9. Что такое критическая секция? Поясните использование.

**Крит. секция** – участок исп.кода программы, в кот. производится доступ к общему ресурсу (данным или у-ву), кот. не дб одновременно исп. более чем одним потоком исполнения

public class Unsafe {  
 private static int x, y;  
 public void Div() {  
 if (y!=0) Console.WriteLine(x / y);  
 y = 0;  
(необх. гарантировать выполнение оп-ров, только одним потоком в любой момент t)

10. Что такое мьютекс? Поясните использование

Systsem.Threading.Mutex  
позв. организ. критич. секцию для неск. процессов  
WainOne() – выхода в крит. секцию  
ReleaseMutex() – для выхода из нее (выход мб произведен только в том же потоке выполнения, что и вход)

11. Что такое семафор? Поясните использование

\* объект синхронизации, позв. войти в зад. участока кода не более чем N потокам  
(N – емкость семафора)  
\* получение и снятие блокировки в случае семафора м. вып. из разных потоков  
\* класс System.Threading.Semaphore – между процессами  
\* класс SemaphoreSlim – в рамках одного процесса  
\* Wait() – получ. блокировки  
\* Release() – снятие блокировки

12. Что такое неблокирующие средства синхронизации?

13. Для чего можно исп. класс Timer?

позв. запускать опр. д-вия по истечению нек. периода времени

int num = 0;  
//устан. метод обратного вызова  
TimerCallback tm = new TimerCallback(Count);  
//созд. таймер  
Timer timer = new Timer(tm, num, 0, 2000);  
 num – объект, перед. в кач. параметра в метод Count  
 0 – кол-во миллисекунд, через кот. таймер б. запуск  
 2000 – интервал между вызовами метода Count