**Вопрос 1** **Укажите правильный способ создания создания домена приложения?**

AppDomain Domain = AppDomain("Demo Domain");

AppDomain Domain = AppDomain.CreateDomain("Demo Domain");

AppDomain Domain = new AppDomain("Demo Domain");

**Ответ:** AppDomain Domain = AppDomain.CreateDomain("Demo Domain")

**Вопрос 2** **Какой метод вы будете использовать для загрузки сборки версии 1.1 для просмотра её метаданных по заданному имени файла, находясь при этом во второй версии сборки .NET Framework?**

Assembly.LoadModule

Assembly.ReflectionOnlyLoad

Assembly.LoadFile

Assembly.ReflectionOnlyLoadFrom

**Ответ:** Assembly.ReflectionOnlyLoadFrom

**Вопрос 3** **Что будет в результате выполнения кода?**

**using System;**

**class Animal {**

**public class Additional**

**{**

**public string MyField = "info";**

**}**

**public enum Some { barks, meows, winks };**

**public delegate void MyDel();**

**public string Name { get; set; }**

**public bool Mammal { get; set; }**

**public void DoSomething() {**

**string s = "some string";**

**int i = 10;**

**}**

**}**

**class Program {**

**static void Main(string[] args) {**

**Animal animal = new Animal();**

**Type t2 = animal.GetType();**

**foreach (Type tp in t2.GetNestedTypes()) {**

**Console.WriteLine("Nested type: {0}", tp.Name);**

**}**

**}**

**}**

исключение

Nested type: Additional Nested type: Some Nested type: MyDel

Nested type: Additional Nested type: Some

Nested type: Additional

пустая консоль

**Ответ:** Nested type: Additional Nested type: Some Nested type: MyDel

**Вопрос 4** **Какое свойство класса Process возвращает уникальный идентификатор процесса (Process Identifier или PID)?**

Id

PID

UniquId

ID

**Ответ:** Id

**Вопрос 5** **Какой из следующих примеров создаёт домен приложения и запускает процесс в нём?**

⭘ AppDomain d = AppDomain.CreateDomain("Domain", "Assembly");

⭘ AppDomain d = AppDomain.CreateDomain("Domain");

d.ExecuteAssemblyByName("Assembly");

⭘ AppDomain.ExecuteAssemblyByName("Assembly");

⭘ AppDomain d = AppDomain.ExecuteAssemblyByName("Domain", "Assembly");

**Ответ:** ⭘ AppDomain d = AppDomain.CreateDomain("Domain");   
d.ExecuteAssemblyByName("Assembly");

**Вопрос 6** **Какие из представленных классов могут участвовать в маршаллинге (передаче данных между AppDomain)? Два или более правильных ответа**

🞎 class A: MarshalByRefType {

public void Method() {

Console.WriteLine("Hello");

}

}

🞎 class A: IMarshallable {

public void Method() {

Console.WriteLine("Hello");

}

}

🞎 [Serializable]

class A

{

public void Method()

{

Console.WriteLine("Hello");

}

}

🞎 class A: MarshalByRefObject

{

public void Method()

{

Console.WriteLine("Hello");

}

}

🞎 class A: IRemotable

{

public void Method()

{

Console.WriteLine("Hello");

}

}

**Ответ:** 🞎 [Serializable]   
class A   
{   
 public void Method()   
 {   
 Console.WriteLine("Hello");   
 }   
}  
🞎 class A: MarshalByRefObject   
{   
 public void Method()   
 {   
 Console.WriteLine("Hello");   
 }   
}

**Вопрос 7** **Что такое поток исполнения?**

Это набор инструкций, организованный в последовательную процедуру на программном уровне

Это набор инструкций, организованный в параллельно выполняемую процедуру на программном уровне

Это реализация многозадачности на физическом уровне

**Ответ:** Это набор инструкций, организованный в последовательную процедуру на программном уровне

**Вопрос 8** **Какой из способов является правильным для закрытия домена приложения и освобождения любых ресурсов, занимаемых сборками, выполняющимися в пределах домена?**

⭘ AppDomain d = AppDomain.CreateDomain("NewDomain");

d = null;

⭘ AppDomain d = AppDomain.CreateDomain("NewDomain");

d.Close();

⭘ AppDomain d = AppDomain.CreateDomain("NewDomain");

d.Unload();

⭘ AppDomain d = AppDomain.CreateDomain("NewDomain");

AppDomain.Unload(d);

**Ответ:** ⭘ AppDomain d = AppDomain.CreateDomain("NewDomain"); AppDomain.Unload(d);

**Вопрос 9** **Что такое многопоточность?**

Это возможность запускать программу как сетевое приложение

Это возможность параллельно выполнять некоторый исполняемый код

Это возможность параллельно исполнять множество приложений в рамках операционной системы

**Ответ:** Это возможность параллельно выполнять некоторый исполняемый код

**Вопрос 10** **Какой из примеров кода корректно отображает модули, загружаемые Application.exe?**

⭘ Process p = new Process();

p.StartInfo.FileName = "Application.exe";

p.Start();

foreach (ProcessModuleCollection m in p.Modules) {

Console.WriteLine(m.FileName);

}

⭘ Process p = new Process();

p.StartInfo.FileName = "Application.exe";

p.Start();

foreach (ProcessModule m in p.Modules) {

Console.WriteLine(m.FileName);

}

⭘ Process p = new Process();

p.StartInfo.FileName = "Application.exe";

p.Start();

foreach (ProcessModule m in Process.Modules(p.Id)) {

Console.WriteLine(m.FileName);

}

⭘ Process p = new Process();

p.StartInfo.FileName = "Application.exe";

p.Start();

foreach (ProcessModuleCollection m in Process.Modules(p.Id)) {

Console.WriteLine(m.FileName);

}

**Ответ:** ⭘ Process p = new Process();   
p.StartInfo.FileName = "Application.exe";   
p.Start();   
foreach (ProcessModule m in p.Modules) {   
 Console.WriteLine(m.FileName);   
}