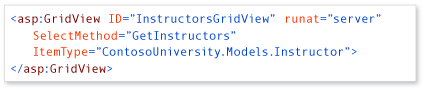
**QUESTION 1**

Which of the followings are inheritance strategies can be used with EF Code First?

|  |  |  |
| --- | --- | --- |
|  |  | All of the above |

**QUESTION 2**



For the above data binding to work in an aspx page, you must

|  |  |  |
| --- | --- | --- |
|  |  | have GetInstructors() method in code behind of the specific page that returns a IEnumerable Collection |

**QUESTION 3**

What is the output of this?

foreach  (char c in "ASP")

Console.WriteLine (c);



**QUESTION 4**

You want to respond to change in a dropdown list, Which is the best way to accomplish this

|  |  |  |
| --- | --- | --- |
|  |  | Set the AutoPost back property of the dropdown control to true |

**QUESTION 5**

When a user selects a value from a drop down box, you want to post back automatically. To do this which property of the control will you change to true?



AutoPostBack="True"

**QUESTION 6**

static void Main() {

Display (new Stock { Name="MSFT", SharesOwned=1000, Value=2000 });

Display (new House { Name="Mansion", Mortgage=100000, Value = 2000 }); }

public static void Display (Asset asset)

{ Console.WriteLine (asset.Name); Console.WriteLine (asset.Value); }

public class Asset

{ public string Name; public decimal Value; }

public class Stock : Asset   // inherits from Asset

{ public long SharesOwned; public decimal Value; }

public class House : Asset   // inherits from Asset

{ public decimal Mortgage; public decimal Value; }

The syntax works ...

 False

**QUESTION 7**

You want to store whether or not user is authenticated and make sure that all request by the user have this information availble, which object should you choose

|  |  |  |
| --- | --- | --- |
|  |  | Session |

**QUESTION 8**

In Code First approach

|  |  |  |
| --- | --- | --- |
|  |  | You can create database and schema from Domain Classes |

**QUESTION 9**

Indicate what application type you will use for each scenario

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| |  |  | | --- | --- | |  | Simple Survey and results | |  | Building a cross platform application and need a service consumable by any mobile device | |  | Need to develop device specific application | |  | Need to build a stateless application with reusable Entity Framework repositories | | |  |  | | --- | --- | | A. | Web Forms | | B. | MVC | | C. | Azure Mobile Services | | D. | Web Api | |

Simple Survey and results: Web Forms

Building a cross platform application and need a service consumable by any mobile device: Web Api

|  |
| --- |
| Need to develop device specific application: Azure Mobile Services |
| Need to build a stateless application with reusable Entity Framework repositories: |

MVC

**QUESTION 10**

In order to specify a named connection string in the configuration file

|  |  |  |
| --- | --- | --- |
|  |  | specify “name=connectionStringName” parameter in the base constructor of a context class |

**QUESTION 11**

You can have two methods with the same name each with different signatures (Parameters, input, output)

 True

**QUESTION 12**

You have to instantiate a class to use its method unless the method is



**QUESTION 13**

In Web Forms, you can do the following:

|  |  |  |
| --- | --- | --- |
|  |  | Call a method from code behind that is outside the page |
|  |  | Leverage Entity Framework |
|  |  | Add an MVC scaffolded page |

**QUESTION 14**

Which command enables migration in Code First. ?

|  |  |  |
| --- | --- | --- |
|  |  | Enable -Migartions |

**QUESTION 15**

public interface IHobby { HobbyName hobby { get; set; } }  
public interface ISkill { SkillName skill{ get; set; } }

public class Person: ISteerable, IBrakable

{     public IHobby hobby{ get; set; }  
    public ISkill skill { get; set; }  
    public Person () { hobby= new HobbyName(); skill= new SkillName(); } }

You  can use two interfaces to compose a new Class?

 True

**QUESTION 16**

The event to tespond to capture and store that a session has timed out is

|  |  |  |
| --- | --- | --- |
|  |  | Session\_End |

**QUESTION 17**

An abstract class can be instantiated

 False

**QUESTION 18**

Which of the following keyword must be used in order to achieve lazy loading?

|  |  |  |
| --- | --- | --- |
|  |  | Virtual |

**QUESTION 19**

Original statement:

static void Main()

{  int x = 12 \* 30;               Console.WriteLine (x);        }

Modified Statement

*static void Main()*

*{  Console.WriteLine (FeetToInches (30));*

*Console.WriteLine (FeetToInches (100));   }*

*static int FeetToInches (int feet)*

*{  int inches = feet \* 12;*

*return inches; }*

The process of changing a set of instructions into a method is known as



**QUESTION 20**

When a user refreshes a web page, the ViewState is lost

 True

**QUESTION 21**

ASP.NET Web forms also uses routing; If you build an application that uses Web Forms and MVC, you must be careful about potential route conflict

 True

**QUESTION 22**

The difference between List View and Grid View Control is

List View you have to generate the HTML vs Grid View you can autogenerate

**QUESTION 23**

string[] names = { "Tom", "Dick", "Harry", "Mary", "Jay" };

IEnumerable<string> query =  from     n in names

                                                where    n.Contains ("a")

                                                orderby  n.Length

                                                select   n.ToUpper();

What will be the items in query? Provide exact answer; Separate each answer with a comma (no space)



**QUESTION 24**

Which DataAnnotation attribute is used when you have multiple relationships between the two classes?

|  |  |  |
| --- | --- | --- |
|  |  | Inverse Property |

**QUESTION 25**

Which of the web application type uses viewstate?

|  |  |  |
| --- | --- | --- |
|  |  | Web Forms |

**QUESTION 26**

Match the right modifiers with the desired behavior

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| |  |  | | --- | --- | |  | class ClassA {int x;} | |  | class Class1 {} | |  | Public class Class2 {} | |  | class ClassB {  internal int x;   } | |
| |  |  | | --- | --- | | A. | Cannot be accessed from other types | | B. | Visible to other types in same assembly | | C. | Visible to everything, including types in other assemblies | | D. | Can be accessed from other types in same assembly | |