DotNet_Developer_Assessment

Test Summary

- No. of Sections: 2
- No. of Questions: 32
- Total Duration: 120 min

Section 1 - MCQ

Section Summary

- No. of Questions: 30
- Duration: 30 min

Additional Instructions:

None

Q1. What is FilterConfig.cs in ASP.Net MVC ?

- FilterConfig.cs is used to register global MVC filters, HandleErrorAttribute is registered by default filter. We can also register other filters.
- FilterConfig.cs is used to register global MVC bundles.
- None
- All

Q2. RedirectToAction() Method for which Status code represents?

- 304
- 302
- 301
- 300

Q3. By default, Web API sends HTTP response with which of the following status code for all uncaught exception?

- 404 - Not Found
- 500 - Internal Server Error
- 204 - No Content
- None of the above

Q4. Which type of validation is used to check password and confirm password in a login form?

- CustomValidator
- RangeValidator

- CompareValidator
- RequiredFieldValidator

Q5. What is the name of the configuration files that the App_Start folder contains?

- BundleConfig.cs
- FilterConfig.cs
- RouteConfig.cs
- RouteConfig.cs

Q6. Which property of the session object is used to set the local identifier?

- LCID
- SessionId
- Key
- Item

Q7. WIF enables you to create a custom token. To be able to use the token, you must create a custom token handler by overriding which of the following?

- SecurityToken
- SecurityTokenHandler
- SWTToken
- Saml2SecurityTokenHandler

Q8. WCF is a framework of _____ .NET Framework

- 1.0
- 2.0
- 2.5
- 3.0 and above

Q9. Middlwares can be configured in _____ method of Startup class

- Configure
- ConfigureService

- Main
- ConfigureMiddleware

Q10. WCF stands for

- Windows Connection Framework
- Windows Common Framework
- Windows Communication Foundation
- Windows Communication Framework

Q11. Which of the following is used to check the validity of the model in Web API?

- Mode.Valid
- Model.IsValid
- ModelState.IsValid
- ModelState.Valid

Q12. What are characteristics best define .NET Core?

- Flexible deployment
- Cross-platform
- Command-line tools
- All of the above

Q13. Which type of contract is applicable to interface in WCF?

- Message contract
- Service contract
- Operation contract
- Data contract

Q14. Web API extract the values of complex type parameters of an action method from _____ by default.

- HTTP Request Body
- HTTP Header
- Message Header

Query String

Q15. Every command in .NET Core command line interface starts with _____

core

dotnet

.net

aspdotnet

Q16. Which of the following class is used to send HTTP requests in .NET 4.5?

WebClient

HttpClient

MessageClient

All of the above

Q17. .NET Core supports which of the following platforms?

Windows

Linux

Mac

All of the above

Q18. Which of the following does not have any visible interface?

Datagrid

Repeater

DropdownList

Datalist

Q19. Web API 2 is supported in _____

- .NET 4.5
- .NET 4.0
- .NET 3.5
- .NET 3.0

Q20. Web API extract the values of primitive type parameters of an action method from _____ by default

- HTTP Request Body
- HTTP Header
- Message Header
- Query String

Q21. Which of the following validation control is used to ensure that a user does not skip a form entity field?

- CompareValidator
- RequiredFieldValidator
- RangeValidator
- RegularExpressionValidator

Q22. The model is a _____

- Shape of data
- Html content
- Collection of data
- Type of data

Q23. Web API Filters are used _____

- to add an extra logic before or after action method executes
- to provide authentication and authorization
- to Launch Web API
- to host Web API

Q24. Which attribute is used to define a service class in WCF?

- Service
- WCFService
- ServiceContract
- None of the above

Q25. Which of the following is an entry point of .NET Core Application?

- Main method of Program class
- Configure method of Startup class
- ConfigureService method of Startup class
- Application_start method of Global.asax

Q26. What are the various types of filters in an ASP.NET MVC application?

- Authorization filters
- Action filters
- Result filters
- All of these

Q27. Which of the following is a formatter class for JSON?

- JsonMediaTypeFormatter
- JsonMediaFormatter
- Json.Net
- None of the above

Q28. _____ is not a member of ADODBCommand object

- ExecuteReader
- ExecuteStream
- ExecuteScalar
- Open

Q29. Which one is not a class in WCF?

BasicHttpBinding

BasicHttpContextBinding

BasicHttpsBinding

ClientBinding

Q30. Which namespace is used for ASPX View Engine?

System.Web.Razor

System.Web.Mvc.WebFormViewEngine

Both A & B

None

Section 2 - Coding

Section Summary

- No. of Questions: 2
- Duration: 90 min

Additional Instructions:

None

Q1. Write a Shuffle operation that disorders the elements of a collection in a random fashion. A shuffle operation is useful in any context. There is no Shuffle operation in System.Collections.Generic.List<T>. In similar Java libraries, there is a shuffle method.

In which class do you want to place the Shuffle operation? You may consider making use of extension methods.

You can decide on programming either a mutating or a non-mutating variant of the operation. Be sure to understand the difference between these two options.

Test the operation on List<Card>. The class Card (representing a playing card) is one of the classes we have seen earlier in the course.

Note:
If the length of the list is either zero or negative values, then should display as "Invalid Input"

Input Format

The first input consists of the length of the list
Followed by all inputs are corresponding to the list elements

Sample Input

```
6
5
3
2
```

Sample Output

```
 5  3  2  7 -4  0

-4  0  2  3  5  7
```

Sample Input

```
0
```

Sample Output

```
Invalid Input
```

Time Limit: - ms Memory Limit: - kb Code Size: - kb

Q2. In the Die event example, we have a public event called twoSixesInARow which is triggered if a die shows two sixes in a row. In the sample client program, we add an anonymous method to this event which reports the string parameter of the event on standard output.

Add yet another method to the twoSixesInARow event which counts the number of times 'two sixes in a row' appear. For this purpose, we need - quite naturally - an integer variable for counting. Where should this variable be located relative to the 'counting method': Will you place the variable inside the new method, inside the Die class, or inside the client class of the Die?

Add a similar event called fullHouse, of the same type Notifier, which is triggered if the Die tosses a full house. A full house means (inspired from the rules of Yahtzee) two tosses of one kind and three tosses of another kind - in a row. For instance, the toss sequence 5 6 5 6 5 leads to a full

house. Similarly, the 1 4 4 4 1 leads to a full house. The toss sequence 5 1 6 6 6 6 5 does not contain a full house sequence, and the toss sequence 6 6 6 6 6 is not a full house.

Be sure to test-drive the program and watch for triggering of both events.

Note:
If iteration count is either zero or negative, then should display as "Invalid Input"

Input Format

The Input consists of the iteration count.

Sample Input

```
1000
```

Sample Output

```
1: 4
2: 2
3: 5
4: 1
```

Sample Input

```
0
```

Sample Output

```
Invalid Input
```

Time Limit: - ms Memory Limit: - kb Code Size: - kb

Answer Key & Solution

Section 1 - MCQ

Q1

FilterConfig.cs is used to register global MVC filters, HandleErrorAttribute is registered by default filter. We can also register other filters.

Solution

No Solution

Q2

302

Solution

No Solution

Q3

500 - Internal Server Error

Solution

No Solution

Q4

CompareValidator

Solution

No Solution

Q5

RouteConfig.cs

RouteConfig.cs

Solution

No Solution

Q6

LCID

Solution

No Solution

Q7

SecurityTokenHandler

Solution

No Solution

Q8

3.0 and above

Solution

No Solution

Q9

Configure

Solution

No Solution

Q10 Windows Communication Foundation

Solution

No Solution

Q11 ModelState.IsValid

Solution

No Solution

Q12 All of the above

Solution

No Solution

Q13 Operation contract

Solution

No Solution

Q14 HTTP Request Body

Solution

No Solution

Q15 dotnet

Solution

No Solution

Q16 HttpClient

Solution

No Solution

Q17 All of the above

Solution

No Solution

Q18 Repeater

Solution

No Solution

Q19 .NET 4.5

Solution

No Solution

Q20 Query String

Solution

No Solution

Q21 RequiredFieldValidator

Solution

No Solution

Q22 Shape of data

Solution

No Solution

Q23 to add an extra logic before or after action method executes

Solution

No Solution

Q24 ServiceContract

Solution

No Solution

Q25 Main method of Program class

Solution

No Solution

Q26 All of these

Solution

No Solution

Q27 JsonMediaTypeFormatter

Solution

No Solution

Q28 Open

Solution

No Solution

Q29 ClientBinding

Solution

No Solution

Q30 System.Web.Mvc.WebFormViewEngine

Solution

No Solution

Section 2 - Coding

Q1

Test Case

Input

```
6
5
3
2
```

Output

```
  5   3   2   7 -4   0

 -4   0   2   3   5   7
```

Weightage - 25

Input

```
0
```

Output

```
Invalid Input
```

Weightage - 25

Input

```
-6
```

Output

```
Invalid Input
```

Weightage - 25

Input

Output

```
  5   3   2   7 -4   0

 -4   0   2   3   5   7
```

Weightage - 25

Sample Input

```
6
5
3
2
```

Sample Output

```
  5   3   2   7 -4   0

 -4   0   2   3   5   7
```

Sample Input

```
0
```

Sample Output

```
Invalid Input
```

Solution

```
using System;
using System.Collections.Generic;

public static class ListExtension{

  public static void Shuffle<T>(this List<T> lst){
   Random randomSupplier = new Random(unchecked((int)DateTime.Now.Ticks));
   for(int i = 0; i < lst.Count - 1; i++){
     int r = randomSupplier.Next(i+1, lst.Count);
     T remember = lst[r];
     lst[r] = lst[i];
     lst[i] = remember;
    }
  }
}

class App{
    public static void Main(){
     List<int> list = new List<int>(new int[]{5, 3, 2, 7, -4, 0});
    //  int input = Convert.ToInt32(Console.WriteLine());
    //  for(int i=0; i<input; i++){
    //      int temp = Convert.ToInt32(Console.WriteLine());
    //      list.push(temp);
    //  }
     ReportList(list);
     Console.WriteLine();
     list.Sort();
     ReportList(list);
     Console.WriteLine();
     for(int i = 1; i < 10; i++){
       list.Shuffle();
       ReportList(list);
       Console.WriteLine();
     }
  }

  public static void ReportList<T>(List<T> list){
    foreach(T el in list)
      Console.Write("{0, 3}", el);
    Console.WriteLine();
  }

}
```

Q2

Test Case

Input

```
100
```

Output

```
1:  6
2:  3
3:  1
4:  4
```

Weightage - 25

Input

```
1000
```

Output

```
1:  4
2:  2
3:  2
You have a Full House
```

Weightage - 25

Input

Output

0	Invalid Input

Weightage - 25

Input **Output**

-6	Invalid Input

Weightage - 25

Sample Input **Sample Output**

1000	1: 4 2: 2 3: 5 4: 1

Sample Input **Sample Output**

0	Invalid Input

Solution

```
using System;
using System.Collections.Generic;

public class Die {
  public delegate void Notifier(string message);  // A delegate type

  private int numberOfEyes;
  private Random randomNumberSupplier;
  private int maxNumberOfEyes;
  private List<int> history;
  public event Notifier twoSixesInARow;
  public event Notifier fullHouse;

  public int NumberOfEyes{
     get {return numberOfEyes;}
  }

public Die (): this(6){}

public Die (int maxNumberOfEyes){
  randomNumberSupplier = new Random(unchecked((int)DateTime.Now.Ticks));
  this.maxNumberOfEyes = maxNumberOfEyes;
  numberOfEyes = randomNumberSupplier.Next(1, maxNumberOfEyes + 1);
  history = new List<int>();
  history.Add(numberOfEyes);
}

public void Toss (){
  numberOfEyes = randomNumberSupplier.Next(1,maxNumberOfEyes + 1);
  history.Add(numberOfEyes);
  if (DoWeHaveTwoSixesInARow(history))
     twoSixesInARow("Two sixes in a row");
  if (DoWeHaveFullHouse(history))
     fullHouse("You have a Full House");
}

private bool DoWeHaveTwoSixesInARow(List<int> history){
  int histLength = history.Count;
  return histLength >= 2 &&
         history[histLength-1] == 6 &&
```

```
        history[histLength-2] == 6;
}



private bool DoWeHaveFullHouse(List<int> history){
  int d1 = 0, d2 = 0,   // number of eyes of the two dies involved in the full house. 0 means uninitialized.
      n1 = 0, n2 = 0,   // how many do we have of each.
      histLength = history.Count;
  if (histLength >= 5){
    for (int i = -5; i <= -1; i++){    // consult history of last 5 tosses
      if      (d1 == 0) {d1 = history[histLength + i]; n1 = 1;}  // the first toss determines d1
      else if (d2 == 0 && d1 != history[histLength + i])
                        {d2 = history[histLength + i]; n2 = 1;}  // a later toss, typically the second, determines d2
      else if (d1 == history[histLength + i]) n1++;
      else if (d2 == history[histLength + i]) n2++;
      else return false;                                         // if we ever come here we have encountered non-full-house toss
    }
    return (n1 == 2 && n2 == 3) || (n1 == 3 && n2 == 2);
 } else return false;
}

public override String ToString(){
  return String.Format("Die[{0}]: {1}", maxNumberOfEyes, NumberOfEyes);
  }
}

class diceApp {

  public static void Main(){

    Die d1 = new Die();

    d1.twoSixesInARow +=
     delegate (string mes){
       Console.WriteLine(mes);
     };

    d1.fullHouse +=
     delegate (string mes){
       Console.WriteLine(mes);
     };

    int doubleSixes = 0;

    d1.twoSixesInARow +=
     delegate (string mes){   // counts how many times we get
       doubleSixes++;         // two sixes in a row
     };

    for(int i = 1; i < 1000; i++){
      d1.Toss();
      Console.WriteLine("{0}: {1}", i, d1.NumberOfEyes);
    }

    Console.WriteLine("Number of two sixes in a row: {0}.", doubleSixes);

 }
}
```