|  |
| --- |
| **Day-18 assignment**  **By**  **Bhanu Rama Krishna Prakash Jakkamsetti**  **16/2/2022** |

|  |
| --- |
| 1.What is the use of XML. |
| * XML is used for universal data transfer mechanism to send data across different platforms. |

|  |
| --- |
| 2.Write the points discussed about XML in the class |
| * Xml is case sensitive. * XML having only one root tag. * XML will have user defined tags. |

|  |
| --- |
| 3.Create a simple XML to illustrate:  a. Tag based XML with 10 products  b. Attribute based Xml. |
| Code for Tag based |
| <Products>  <Product>  <brand>puma</brand>  <price>1000</price>  <quantity>1</quantity>  </Product>  <Product>  <brand>nike</brand>  <price>2000</price>  <quantity>2</quantity>  </Product>  <Product>  <brand>addidas</brand>  <price>3000</price>  <quantity>3</quantity>  </Product>  <Product>  <brand>tommyhilfiger</brand>  <price>4000</price>  <quantity>4</quantity>  </Product>  <Product>  <brand>armani</brand>  <price>5000</price>  <quantity>5</quantity>  </Product>  <Product>  <brand>levis</brand>  <price>6000</price>  <quantity>6</quantity>  </Product>  <Product>  <brand>zara</brand>  <price>7000</price>  <quantity>7</quantity>  </Product>  <Product>  <brand>calvinklein</brand>  <price>8000</price>  <quantity>8</quantity>  </Product>  <Product>  <brand>prada</brand>  <price>9000</price>  <quantity>9</quantity>  </Product>  <Product>  <brand>gucci</brand>  <price>10000</price>  <quantity>10</quantity>  </Product>  </Products> |
| Output: |
|  |
| Code for Attribute based |
| <Products>  <Product brand="puma" price="1000" quantity="1" />  <Product brand="nike" price="2000" quantity="2" />  <Product brand="addidas" price="3000" quantity="3" />  <Product brand="tommyhilfiger" price="4000" quantity="4" />  <Product brand="armani" price="5000" quantity="5" />  <Product brand="zara" price="6000" quantity="6" />  <Product brand="calvinklein" price="7000" quantity="7" />  <Product brand="prada" price="8000" quantity="8" />  <Product brand="gucci" price="9000" quantity="9" />  <Product brand="bhanu" price="10000" quantity="10" />  </Products> |
| Output: |
|  |

|  |
| --- |
| 4. Convert the above Xml to JSON and display the JSON data. |
| Code: |
| <Products>  <Product brand="puma" price="1000" quantity="1" />  <Product brand="nike" price="2000" quantity="2" />  <Product brand="addidas" price="3000" quantity="3" />  <Product brand="tommyhilfiger" price="4000" quantity="4" />  <Product brand="armani" price="5000" quantity="5" />  <Product brand="zara" price="6000" quantity="6" />  <Product brand="calvinklein" price="7000" quantity="7" />  <Product brand="prada" price="8000" quantity="8" />  <Product brand="gucci" price="9000" quantity="9" />  <Product brand="bhanu" price="10000" quantity="10" />  </Products> |
| Output: |
| [  {  "@brand": "puma",  "@price": "1000",  "@quantity": "1"  },  {  "@brand": "nike",  "@price": "2000",  "@quantity": "2"  },  {  "@brand": "addidas",  "@price": "3000",  "@quantity": "3"  },  {  "@brand": "tommyhilfiger",  "@price": "4000",  "@quantity": "4"  },  {  "@brand": "armani",  "@price": "5000",  "@quantity": "5"  },  {  "@brand": "zara",  "@price": "6000",  "@quantity": "6"  },  {  "@brand": "calvinklein",  "@price": "7000",  "@quantity": "7"  },  {  "@brand": "prada",  "@price": "8000",  "@quantity": "8"  },  {  "@brand": "gucci",  "@price": "9000",  "@quantity": "9"  },  {  "@brand": "bhanu",  "@price": "10000",  "@quantity": "10"  }  ] |

|  |
| --- |
| 5.Research and write the benefits of JSON over XML. |
| * It consumes less space. * In most scenarios, JSON is undoubtedly easier to read in its expanded form than XML. * JSON can have a substantially lower character count reducing the overhead in data transfer. |

|  |
| --- |
| 6.Create a layered architecture project with separate class library for business logic.  Create console app  Create windows app  Requirement:  Find factorial of a number:  Positive number (up to 7) = factorial  0=1  >7 = -999  <0 = -9999 |
| Code for console: |
| using System;  using System.Collections.Generic;  using System.Linq;  using System.Text;  using System.Threading.Tasks;  using Mathematicslibrary;  namespace Day18\_project  {  /\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  \* Author:bhanu rama krishna prakash jakkamsetti  \* purpose:acces methods from ohter assembly by creating library  \* \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/  public class Program  {  static void Main(string[] args)  {  int a;  Console.WriteLine("enter number");  a=Convert.ToInt32(Console.ReadLine());  Console.WriteLine(Algebra.Factorial(a));  Console.ReadLine();  }  }  }  using System;  using System.Collections.Generic;  using System.Linq;  using System.Text;  using System.Threading.Tasks;  namespace Mathematicslibrary  {  public class Algebra  {  /// <summary>  /// finding facttorial  /// </summary>  /// <param name="a"></param>  /// <returns>factorial</returns>  public static int Factorial(int a)  {  if (a == 0)  return 1;  else if (a > 7)  return -999;  else if (a < 0)  return -9999;  else  {  int fact = 1;  for (int i = 1; i <=a; i++)  fact \*= i;  return fact;  }  }  }  } |
| Output: |
|  |
| Code for windows: |
| using System;  using System.Collections.Generic;  using System.ComponentModel;  using System.Data;  using System.Drawing;  using System.Linq;  using System.Text;  using System.Threading.Tasks;  using System.Windows.Forms;  using Mathematicslibrary;  namespace Desktopapp  {  public partial class Form1 : Form  {  public Form1()  {  InitializeComponent();  }  private void label1\_Click(object sender, EventArgs e)  {  }  private void button1\_Click(object sender, EventArgs e)  {  int n= Convert.ToInt32(textBox1.Text);  int result = Algebra.Factorial(n);  textBox2.Text = result.ToString();  Console.ReadLine();  }  }  }  using System;  using System.Collections.Generic;  using System.Linq;  using System.Text;  using System.Threading.Tasks;  namespace Mathematicslibrary  {  public class Algebra  {  /// <summary>  /// finding facttorial  /// </summary>  /// <param name="a"></param>  /// <returns>factorial</returns>  public static int Factorial(int a)  {  if (a == 0)  return 1;  else if (a > 7)  return -999;  else if (a < 0)  return -9999;  else  {  int fact = 1;  for (int i = 1; i <=a; i++)  fact \*= i;  return fact;  }  }  }  } |
| Output: |
|  |

|  |
| --- |
| 7.For the above method, implement TDD and write 4 test cases. |
| Code: |
| Code for test class:  using Microsoft.VisualStudio.TestTools.UnitTesting;  using MAthlibrary;  using System;  using System.Collections.Generic;  using System.Linq;  using System.Text;  using System.Threading.Tasks;  namespace MAthlibrary.Tests  {  [TestClass()]  public class MathoperationsTests  {  [TestMethod()]  public void FindFactorialTest\_zeroinput()  {  //arrange  int n = 0;  int expected = 1;  //act  int actual = Mathoperations.FindFactorial(n);  //assert  Assert.AreEqual(expected, actual);  }  [TestMethod()]  public void FindFactorialTest\_Onetoseven()  {  //arrange  int n = 5;  int expected = 120;  //act  int actual=Mathoperations.FindFactorial(n);  //assert  Assert.AreEqual(expected , actual);  }  [TestMethod()]  public void FindFactorialTest\_graeterthan7()  {  //arrange  int n = 9;  int expected = -999;  //act  int actual=Mathoperations.FindFactorial(n);  //assert  Assert.AreEqual(expected, actual);  }  [TestMethod()]  public void FindFactorialTest\_Lessthan0()  {  //arrange  int n = -10;  int expected = -9999;  //act  int actual= Mathoperations.FindFactorial(n);  //assert  Assert.AreEqual(expected, actual);    }  }  }  Code for main class:  using System;  using System.Collections.Generic;  using System.Linq;  using System.Text;  using System.Threading.Tasks;  namespace MAthlibrary  {  public class Mathoperations  {  public static int FindFactorial(int n)  {  if (n == 0)  return 1;  else if (n > 7)  return -999;  else if (n < 0)  return -9999;  else  {  int fact = 1;  for (int i = 1; i <= n; i++)  fact \*= i;  return fact;  }  }  }  } |
| Output: |
|  |

|  |
| --- |
| 8. Add one more method to check if the number is palindrome or not in the above algebra class and write test case for the same. |
| Code: |
| Code for testing class:  using Microsoft.VisualStudio.TestTools.UnitTesting;  using MAthlibrary;  using System;  using System.Collections.Generic;  using System.Linq;  using System.Text;  using System.Threading.Tasks;  namespace MAthlibrary.Tests  {  [TestClass()]  public class MathoperationsTests  {  [TestMethod()]  public void FindFactorialTest\_zeroinput()  {  //arrange  int n = 0;  int expected = 1;  //act  int actual = Mathoperations.FindFactorial(n);  //assert  Assert.AreEqual(expected, actual);  }  [TestMethod()]  public void FindFactorialTest\_Onetoseven()  {  //arrange  int n = 5;  int expected = 120;  //act  int actual = Mathoperations.FindFactorial(n);  //assert  Assert.AreEqual(expected, actual);  }  [TestMethod()]  public void FindFactorialTest\_graeterthan7()  {  //arrange  int n = 9;  int expected = -999;  //act  int actual = Mathoperations.FindFactorial(n);  //assert  Assert.AreEqual(expected, actual);  }  [TestMethod()]  public void FindFactorialTest\_Lessthan0()  {  //arrange  int n = -10;  int expected = -9999;  //act  int actual = Mathoperations.FindFactorial(n);  //assert  Assert.AreEqual(expected, actual);  }  [TestMethod()]  public void IsPalindromeTest()  {  //arrange  int input = 353;  int expected = 353;  //art  int actual=Mathoperations.IsPalindrome(input);  //assert  Assert.AreEqual(expected, actual);  }  }  }  Code for main class:  using System;  using System.Collections.Generic;  using System.Linq;  using System.Text;  using System.Threading.Tasks;  namespace MAthlibrary  {  public class Mathoperations  {  public static int FindFactorial(int n)  {  if (n == 0)  return 1;  else if (n > 7)  return -999;  else if (n < 0)  return -9999;  else  {  int fact = 1;  for (int i = 1; i <= n; i++)  fact \*= i;  return fact;  }  }  public static int IsPalindrome(int num)  {  int rem, temp, sum = 0;  temp = num;  while (num>0)  {  rem = num % 10;  sum = (sum \* 10) + rem;  num /= 10;  }  if (temp==sum)  Console.WriteLine($"{temp} is palindrome.");  else  Console.WriteLine($"{temp} is not palindrome.");  return temp;  }  }  } |
| Output: |
|  |