|  |
| --- |
| DAY 18Th ASSIGNMENT  BY  J SIVA NAGA PRASANNA |

|  |
| --- |
| 1 .what is the use of XML? |

|  |
| --- |
| * XML describes structure of document * XML is used to describe data |

|  |
| --- |
| 2. write points discussed about XML in class |
| * Xml stands for eXtensible markup language * Xml is used for universal data transfer mechanism to send data across different platforms * Xml will have user defined tags * Xml can only have one root tag * Xml is a case sensitive |

|  |
| --- |
| 3.Research and write the benfits of JSON over xml |
| * JSON is easier to read . its expanded form than XML. * JSON can have a substantially lower character count reducing the overhead in data transfers. * JSON is much easier to parse. |

|  |
| --- |
| **4.** **Create a simple xml to illustrate**  **a. Tag based xml with 10 products**  **b. Attribute based xml** |
| **A. Tag based xml with 10 products** |
| Code:  <products>  <product>  <ID1>1000</ID1>  <name>tv1</name>  <price>40000</price>  </product>  <product>  <ID2>1001</ID2>  <name>tv2</name>  <price>50000</price>  </product>  <product>  <ID3>1002</ID3>  <name>tv3</name>  <price>60000</price>  </product>  <product>  <ID4>1003</ID4>  <name>tv4</name>  <price>70000</price>  </product>  <product>  <ID5>1004</ID5>  <name>tv5</name>  <price>80000</price>  </product>  <product>  <ID6>1005</ID6>  <name>tv6</name>  <price>90000</price>  </product>  <product>  <ID7>1006</ID7>  <name>tv7</name>  <price>18000</price>  </product>  <product>  <ID8>1007</ID8>  <name>tv8</name>  <price>28000</price>  </product>  <product>  <ID9>1008</ID9>  <name>tv9</name>  <price>48000</price>  </product>  <product>  <ID10>1009</ID10>  <name>tv10</name>  <price>48000</price>  </product>  </products> |
|  |

|  |
| --- |
| **B. Attribute based xml** |
| <product>  <product ID1="1001" Name="mobile1" price="20000"/>  <product ID2="1002" Name="mobile2" price="21000"/>  <product ID3="1003" Name="mobile3" price="22000"/>  <product ID4="1004" Name="mobile4" price="23000"/>  <product ID5="1005" Name="mobile5" price="24000"/>  <product ID6="1006" Name="mobile6" price="25000"/>  <product ID7="1007" Name="mobile7" price="26000"/>  <product ID8="1008" Name="mobile8" price="27000"/>  <product ID9="1009" Name="mobile9" price="28000"/>  <product ID10="1000" Name="mobile" price="29000"/>  </product> |
| OUTPUT: |
|  |

|  |
| --- |
| **5.Convert the above xml to JSON and display the JSON data** |
| **Tag based Xml:** |
| Output:  [https://www.freeformatter.com/3.7.0.0/img/minus.gif  {https://www.freeformatter.com/3.7.0.0/img/minus.gif  "ID1": "1000",  "name": "tv1",  "price": "40000"  },  {https://www.freeformatter.com/3.7.0.0/img/minus.gif  "ID2": "1001",  "name": "tv2",  "price": "50000"  },  {https://www.freeformatter.com/3.7.0.0/img/minus.gif  "ID3": "1002",  "name": "tv3",  "price": "60000"  },  {https://www.freeformatter.com/3.7.0.0/img/minus.gif  "ID4": "1003",  "name": "tv4",  "price": "70000"  },  {https://www.freeformatter.com/3.7.0.0/img/minus.gif  "ID5": "1004",  "name": "tv5",  "price": "80000"  },  {https://www.freeformatter.com/3.7.0.0/img/minus.gif  "ID6": "1005",  "name": "tv6",  "price": "90000"  },  {https://www.freeformatter.com/3.7.0.0/img/minus.gif  "ID7": "1006",  "name": "tv7",  "price": "18000"  },  {https://www.freeformatter.com/3.7.0.0/img/minus.gif  "ID8": "1007",  "name": "tv8",  "price": "28000"  },  {https://www.freeformatter.com/3.7.0.0/img/minus.gif  "ID9": "1008",  "name": "tv9",  "price": "48000"  },  {https://www.freeformatter.com/3.7.0.0/img/minus.gif  "ID10": "1009",  "name": "tv10",  "price": "48000"  }  ]  **Attribute based xml:** |
| [https://www.freeformatter.com/3.7.0.0/img/minus.gif  {https://www.freeformatter.com/3.7.0.0/img/minus.gif  "@ID1": "1001",  "@Name": "mobile1",  "@price": "20000"  },  {https://www.freeformatter.com/3.7.0.0/img/minus.gif  "@ID2": "1002",  "@Name": "mobile2",  "@price": "21000"  },  {https://www.freeformatter.com/3.7.0.0/img/minus.gif  "@ID3": "1003",  "@Name": "mobile3",  "@price": "22000"  },  {https://www.freeformatter.com/3.7.0.0/img/minus.gif  "@ID4": "1004",  "@Name": "mobile4",  "@price": "23000"  },  {https://www.freeformatter.com/3.7.0.0/img/minus.gif  "@ID5": "1005",  "@Name": "mobile5",  "@price": "24000"  },  {https://www.freeformatter.com/3.7.0.0/img/minus.gif  "@ID6": "1006",  "@Name": "mobile6",  "@price": "25000"  },  {https://www.freeformatter.com/3.7.0.0/img/minus.gif  "@ID7": "1007",  "@Name": "mobile7",  "@price": "26000"  },  {https://www.freeformatter.com/3.7.0.0/img/minus.gif  "@ID8": "1008",  "@Name": "mobile8",  "@price": "27000"  },  {https://www.freeformatter.com/3.7.0.0/img/minus.gif  "@ID9": "1009",  "@Name": "mobile9",  "@price": "28000"  },  {https://www.freeformatter.com/3.7.0.0/img/minus.gif  "@ID10": "1000",  "@Name": "mobile",  "@price": "29000"  }  ] |

|  |
| --- |
| 6.The below requirement, create a layered architecture  project with seperate class library for Business logic.    1. create console application  2. create windows application  Business Requirement:  FIND FACTORIAL OF A NUMBER:  a)0 = 1  b) positive number (up to 7) = factorial answer  c)> 7 = -999 (as answer)  d)< 0 = -9999 (as answer)  put the screen shots of the output |
| Algebra:  using System;  using System.Collections.Generic;  using System.Linq;  using System.Text;  using System.Threading.Tasks;  namespace MathematicsLibrary  {  public class Algebra  {  public static int Factorial(int n)  {  int fact = 1;  if (n == 0)  return 1;  else if (n > 7)  return -999;  else if (n < 0)  return -9999;  else  {  for (int i = 1; i <= n; i++)  fact = fact \* i;  return fact;  }  }  }  } |
| Program.cs  Code:  using System;  using System.Collections.Generic;  using System.Linq;  using System.Text;  using System.Threading.Tasks;  using MathematicsLibrary;    namespace day18pro1  {  internal class Program  {  static void Main(string[] args)  {  int n;  Console.WriteLine("Enter number");  n = Convert.ToInt32(Console.ReadLine());  Console.WriteLine(Algebra.Factorial(n));  Console.ReadLine();  }  }  } |
| Windowsapp:  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 MathematicLibrary;  namespace MyApp1  {  public partial class Form1 : Form  {  public Form1()  {  InitializeComponent();  }  private void button1\_Click(object sender, EventArgs e)  {  int n=Convert.ToInt32(textBox1.Text);  int result=Algebra.Factorial(n);  textBox2.Text = result.ToString();  }  } |
| Output: |

|  |
| --- |
| **7.For the above method, Implement TDD**  **And write 4 test cases and put the code in word document.**  **put the screen shot of all test cases failing.**  **make the test cases pass.**  **put the screen shot.** |
| Algebra Tests:  using Microsoft.VisualStudio.TestTools.UnitTesting;  using MathematicsLibrary;  using System;  using System.Collections.Generic;  using System.Linq;  using System.Text;  using System.Threading.Tasks;  namespace MathematicsLibrary.Tests  {  [TestClass()]  public class AlgebraTests  {  [TestMethod()]  public void FactorialTest\_Zero\_input()  {  //Arrange  int n = 0;  int expected = 1;  //Act  int actual = Algebra.Factorial(n);  //Assert  Assert.AreEqual(expected, actual);  }  [TestMethod()]  public void FactorialTest\_One\_To\_Seven\_Input()  {  //Arrange  int n = 6;  int expected = 720;  //Act  int actual = Algebra.Factorial(n);  //Assert  Assert.AreEqual(expected, actual);  }  [TestMethod()]  public void FactorialTest\_greater\_than\_Seven\_Input()  {  //Arrange  int n = 11;  int expected = -999;  //Act  int actual = Algebra.Factorial(n);  //Assert  Assert.AreEqual(expected, actual);  }  [TestMethod()]  public void FactorialTest\_Negitive\_Input()  {  //Arrange  int n = -4;  int expected = -9999;  //Act  int actual = Algebra.Factorial(n);  //Assert  Assert.AreEqual(expected, actual);  }  }  } |
| Algebra.cs:  using System;  using System.Collections.Generic;  using System.Linq;  using System.Text;  using System.Threading.Tasks;  namespace MathematicsLibrary  {  public class Algebra  {  public static int Factorial(int n)  {  if (n == 0)  return 1;  else if (n < 0)  return -9999;  else if (n > 7)  return -999;  else  {  int fact = 1;  for (int i = 1; i <= n; i++)  fact = fact \* i;  return fact;  return 0;  }  }  }  } |
| Output: |
|  |