|  |
| --- |
| **Day-11 assignment**  **By**  **Bhanu Rama Krishna Prakash Jakkamsetti**  **7/2/2022** |

1. Research and write the difference between

abstract class and interface in C#

|  |  |
| --- | --- |
| Abstract class | interface |
| It contains both declaration and definition part. | It contains only a declaration part. |
| Multiple inheritance is not achieved by abstract class. | Multiple inheritance is achieved by interface. |
| It contains constructor. | It does not contains constructor. |
| It can contain static members. | It does not contain static members. |
| It can contain different types of access modifiers like public, private, protected etc. | It only contains public access modifier because everything in the interface is public. |
| The performance of an abstract class is fast. | The performance of interface is slow because it requires time to search actual method in the corresponding class. |
| It is used to implement the core identity of class. | It is used to implement peripheral abilities of class. |
| A class can only use one abstract class. | A class can use multiple interface. |
| If many implementation are of the same kind and use common behaviour, then it is superior to use abstract class. | In many implementation only share methods, then it is superior to use interface. |
| Abstract class can contain methods, fields, constants, etc. | Interface can only contains methods, properties, indexers, events. |
| It can be fully, partially or not implemented. | It should be fully implemented . |

2. Write the 6 points about interface discussed in the class

1. Interface is pure abstract class.
2. Interface name should start with I.
3. Interface acts like a contract.
4. By default the methods in interface are public and abstract.
5. Any class that is implementing interface must override all the methods.
6. Interface support multiple inheritance.

|  |
| --- |
| 3.Write example program for interfaces discussed in the class Ishape  Include the classes (circle, square, triangle, rectangle) |
| Code: |
| using System;  using System.Collections.Generic;  using System.Linq;  using System.Text;  using System.Threading.Tasks;  namespace Day11\_project2  {    /\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  \* Author:Bhanu Rama Krishna Prakash Jakkamsetti  \* purpose:creating interface class  \* \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/  interface Ishape  {  /// <summary>  /// creating abstract method for parameter  /// </summary>  /// <returns></returns>  int Calculateparameter();  /// <summary>  /// creating abstract method for area  /// </summary>  /// <returns></returns>  int Calculatearea();  }  class Circle : Ishape  {  int radius;  /// <summary>  /// to read data  /// </summary>  public void Readdradius()  {  Console.WriteLine("enter radius");  radius = Convert.ToInt32(Console.ReadLine());  }  /// <summary>  /// calculate area  /// </summary>  /// <returns>area</returns>  public int Calculatearea()  {  return 22 \* radius \* radius / 7;  }  /// <summary>  /// calculate parameter  /// </summary>  /// <returns>parameter</returns>  public int Calculateparameter()  {  return 2 \* 22 \* radius / 7;  }  }  class Square : Ishape  {  int sides;  /// <summary>  /// to read data  /// </summary>  public void Readsides()  {  Console.WriteLine("enter sides value");  sides = Convert.ToInt32(Console.ReadLine());  }  /// <summary>  /// calculate area  /// </summary>  /// <returns>area</returns>  public int Calculatearea()  {  return 4 \* sides;  }  /// <summary>  /// calculate parameter  /// </summary>  /// <returns>parameter</returns>  public int Calculateparameter()  {  return sides \* sides;  }  }  class IsoscelesTriangle : Ishape  {  int a, b;  /// <summary>  /// to read data  /// </summary>  public void Readaandb()  {  Console.WriteLine("inter a");  a = Convert.ToInt32(Console.ReadLine());  Console.WriteLine("inter b");  b = Convert.ToInt32(Console.ReadLine());  }  /// <summary>  /// calculate area  /// </summary>  /// <returns>area</returns>  public int Calculatearea()  {  int A = a \* a;  int B = b \* b;  int area= (int)(0.5 \* b \* Math.Sqrt(A - B / 2));  return area;  }  /// <summary>  /// calculate parameter  /// </summary>  /// <returns>parameter</returns>  public int Calculateparameter()  {  return (2 \* a + b);  }  }  class Rectangle : Ishape  {  int length, breadth;  /// <summary>  /// to read data  /// </summary>  public void Readlandb()  {  Console.WriteLine("enter length");  length = Convert.ToInt32(Console.ReadLine());  Console.WriteLine("enter breadth");  breadth = Convert.ToInt32(Console.ReadLine());  }  /// <summary>  /// calculate area  /// </summary>  /// <returns>area</returns>  public int Calculatearea()  {  return 2 \* (length + breadth);  }  /// <summary>  /// calculate parameter  /// </summary>  /// <returns>parameter</returns>  public int Calculateparameter()  {  return length \* breadth;  }  }  internal class Program  {  static void Main(string[] args)  {  Circle c = new Circle();  c.Readdradius();  Console.WriteLine(c.Calculatearea());  Console.WriteLine(c.Calculateparameter());  Square s = new Square();  s.Readsides();  Console.WriteLine(s.Calculatearea());  Console.WriteLine(s.Calculateparameter());  IsoscelesTriangle t = new IsoscelesTriangle();  t.Readaandb();  Console.WriteLine(c.Calculatearea());  Console.WriteLine(c.Calculateparameter());  Rectangle r = new Rectangle();  r.Readlandb();  Console.WriteLine(r.Calculatearea());  Console.WriteLine(r.Calculateparameter());  Console.ReadLine();  }  }  } |
| Output: |
|  |

4. Write the 7 points discussed about properties.

1.Properties are almost same as class variables with get; and set;

2.A Property with only get is read-only.

3.A Property with only set is write only.

4.A Property with get and set => you can read value and assign the value.

5.History of properties:

a. Properties are introduced to deal with private variables.

b. A very simple example for properties are:

Class Employee

{

Private int id;

Private string name;

Private string designation;

Public int Id

{

get{return id;}

set{id=value;}

}

}

|  |
| --- |
| 5.Write sample code to illustrate properties as discussed in class—  Id-get, set  Name-get, set  Designation-set  Salary-get(with some functionality) |
| Code: |
| using System;  using System.Collections.Generic;  using System.Linq;  using System.Text;  using System.Threading.Tasks;  namespace Day11\_project3  {  /\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  \* Author:Bhanu Rama Krishna Prakash Jakkamsetti  \* purpose:creating properties  \* \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/  class Employee  {  private int id;  private string name;  private string disignation;  private int salary;  /// <summary>  /// creating get and set  /// </summary>  public int Id  {  get { return id; }  set { id = value; }  }  /// <summary>  /// creating get and set  /// </summary>  public string Name  {  get { return name; }  set { name = value; }  }  /// <summary>  /// creating get and set  /// </summary>  public string Disignation  {  set { disignation = value; }  }  /// <summary>  /// creating get  /// </summary>  public int Salary  {  get  {  salary = (disignation == "M") ? 3000 : 6000;  return salary;  }    }  }  internal class Program  {  static void Main(string[] args)  {  Employee e = new Employee();  e.Disignation = "M";  Console.WriteLine(e.Salary);  }  }  } |
| Output: |
|  |

|  |
| --- |
| 6.Create a class employee with only properties. |
| Code: |
| using System;  using System.Collections.Generic;  using System.Linq;  using System.Text;  using System.Threading.Tasks;  namespace Day11\_project4  {  /\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  \* Author:Bhanu Rama Krishna Prakash Jakkamsetti  \* purpose:creating only properties  \* \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/  class employee  {  public int id { get; set; }  public string name { get; set; }  public string disignation { get; set; }  public int salary { get; set; }  }  internal class Program  {  static void Main(string[] args)  {  }  }  } |

|  |
| --- |
| 7.Create mathematics class and add 3 static methods and call the methods in main method. |
| Code: |
| using System;  using System.Collections.Generic;  using System.Linq;  using System.Text;  using System.Threading.Tasks;  namespace Day11\_project5  {  /\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  \* Author:Bhanu Rama Krishna Prakash Jakkamsetti  \* purpose:creating static methods  \* \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/  class Employee  {  public int Id;  public string Name;  public static string company = "NationsBenifits";  /// <summary>  /// for to read data  /// </summary>  public void Readdata()  {  Console.WriteLine("enter id");  Id=Convert.ToInt32(Console.ReadLine());  Console.WriteLine("enter name");  Name=Console.ReadLine();  }  /// <summary>  /// to print name  /// </summary>  public void Printname()  {  Console.WriteLine(Name);  }  /// <summary>  /// print hello  /// </summary>  public static void Hello()  {  Console.WriteLine("hello");  }  }  class Mathimatics  {  /// <summary>  /// add  /// </summary>  /// <param name="a"></param>  /// <param name="b"></param>  /// <returns>a+b</returns>  public static int Add(int a,int b)  {  return a + b;  }  /// <summary>  /// multiplication  /// </summary>  /// <param name="a"></param>  /// <param name="b"></param>  /// <returns>a\*b</returns>  public static int Mul(int a,int b)  {  return a\*b;  }  }  internal class Program  {  static void Main(string[] args)  {  Console.WriteLine(Employee.company);  Employee e = new Employee();  Console.WriteLine(Mathimatics.Add(5,10));  Console.WriteLine(Math.Pow(5, 2));  }  }  } |
| Output: |
|  |