ABHISHEK SINGH

(ASET taazaa)

***Question Define virtual , override and new keyword and uses.***

**Virtual** :- We use virtual keyword with method and properties of parent class so that child of parent class can overrides the properties and methods of parent class.

For Example -

public class ClassA

{

public **virtual** void Aclassmethod() {

Console.WriteLine("A class Method is called");

}

}

public class ClassB : ClassA

{

public **override** void Aclassmethod()

{

Console.WriteLine("B class method overrides class A method");

}}

**Override** – By the use of override in child class (with properties and methods)

we can rewrite the parent class virtual properties and methods.

For Example - ( ***// Implementation of both virtual and override keyword***)

public class ClassA

{

public **virtual** void Aclassmethod() {

Console.WriteLine("A class Method is called");

}

}

public class ClassB : ClassA

{

public ***override*** void Aclassmethod()

{

Console.WriteLine("B class method overrides class A method");

}

}

class **Program** **//Implementation of virtual and override**

{

static void Main(string[] args)

{

ClassA A1 = new ClassA(); **// (return) A class Method is called**

A1.Aclassmethod();

ClassB B1 = new ClassB(); **// B class method overrides class A method**

B1.Aclassmethod();

ClassA A2 = new ClassB(); /**/ B class method overrides class A method**

A2.Aclassmethod();

}

}

**New** - If we use new keyword when we override a method of parent class

then new keyword breaks the chain and create new method of the same name in class.

For Example -

class ClassC : ClassB

{

public **new** void Aclassmethod()

{

Console.WriteLine("C has own Aclassmethod");

}

}

**// *then new will break the chain and create new Aclassmethod() in class C.***

***Question. Define Constructor and its types***.

**Constructor** :- It is use to initialize the property of object and also use to create objects. By default, there is always a default constructor of a class present in it which is empty constructor.

There are five types of constructor present in C#.

1. **Default constructor**
2. **Static constructor**
3. **Copy constructor**
4. **Private constructor**
5. **Parameterized constructor**

**Default constructor -** By default , a constructor is present inside the class .

when we create the object of the class then default constructor is invoked and constructor have no arguments.

For Example -

public Human() { **// rewrite the default constructor**

Console.WriteLine("This is default constructor of Human class");

}

static void Main(string[] args)

{

Human h1 = new Human(); ***// (return)This is default constructor of Human class***

}

**Static constructor :-**

**1.**Static constructor are those which is call before object create.

2. Static constructor must be parameter less.

3. You cannot excess non-static field or method inside static constructor.

For Example -

public class Human

{

static int number = 121;

static string str = "Static Member"

static Human() { **//Static constructor**

Console.WriteLine(number);

Console.WriteLine(str);

Console.WriteLine("This is default constructor of Human class");

}

}

**Parameterized constructor :-** In parameterized constructor we pass arguments to the constructor.

public class Human

{

public string Name;

public int Age;

public Human(string name, int age) {

this.Name = name;

this.Age = age;

}

public void display() {

Console.WriteLine("Name : " + Name);

Console.WriteLine("Age : " + Age);

}

***Human h1 = new Human(“Abhi”,21); // create object***

***h1.display(); // display name and age***

***Copy constructor :-*** By the use of copy constructor we can copy value of one object to another object but it create new object not reference of object.

For Example -

public Human (Human p){

Name = p.Name;

Age = p.Age;

}

***// Human h1 = new Human();***

***// Human h2 = new Human(h1);***

**Private constructor :-** by the use of private constructor we can restrict a class to make instance of that class and only contain static members only.

For Example -

private Human() {

Console.WriteLine("Private Constructor");

}

Throw error -

***Error*** ***CS0122*** ***'Human.Human()' is inaccessible due to its protection level***

***Question. Define property initializer in C#.***

In properties we use the get and set methods. Get is used to read the value and is called getter and set is used to assign a value and is called setter.

If we want to make a method read-only then for that we need to remove the set method.

There are three types of property initializer

1. **Object property initializer**
2. **Automatic property initializer**
3. **Inline property initializer**

**Object property initializer :-** we use object property initializer to define objects with properties.

For Example – Adult a1 = new adult(){ Name : “Koo", Age :21}

**Automatic property initializer :-** In this property automatic initialize with a predefined variable in get and set.

For Example - public age {get ; set;}

**Inline property initializer :-** In inline property the name is define by the user for getter and setter.

For Example-

Public string name;

public string name { get => name; set => name = value ;}