Day10Assignments
(4th Feb 2022)

By
RamCharan

1.Write about Inheritance.

- Inheritance is the process of reusing base class methods in the derived class.
- In other words, Inheritance is the process of inheriting the properties of parent class to a child class.
- The main goal of using Inheritance is reusability and to remove dupliacte code.

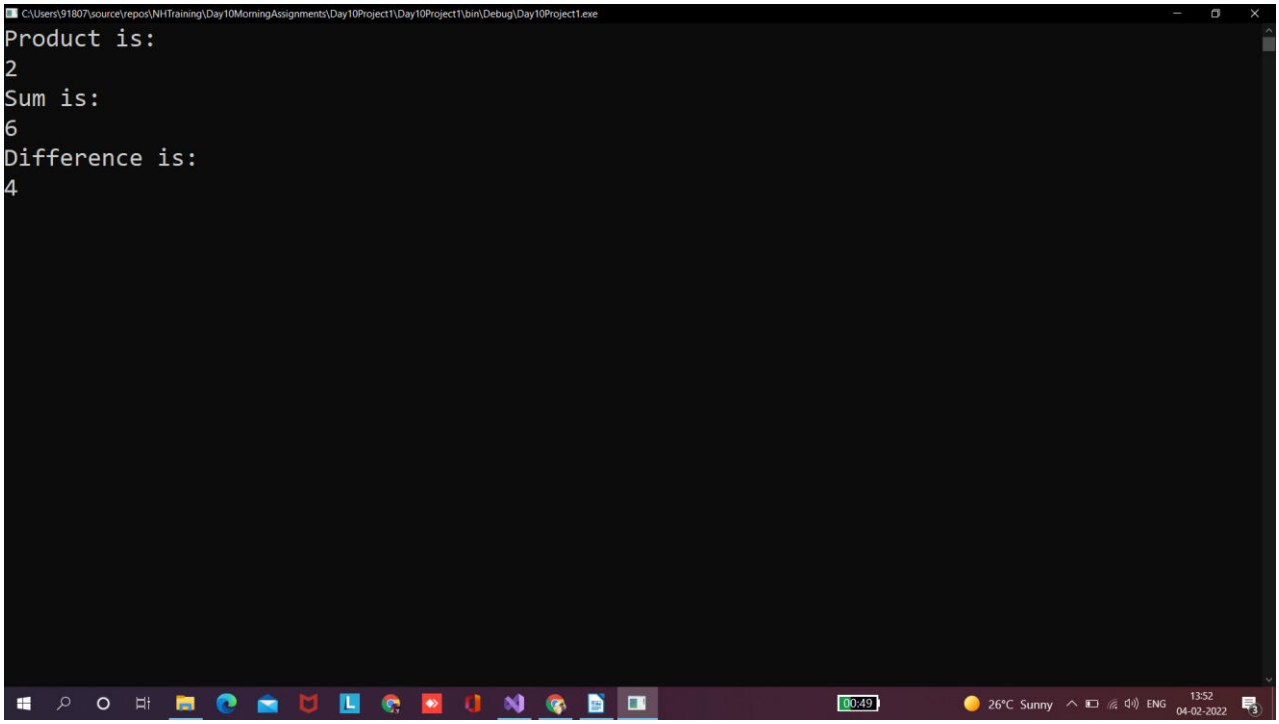
2. a.)Write a code for Single Inheritance

Code:

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace Day10Project1
{
   //Author :Rc
   /*Purpose : Single inheritance */
   class A //parent class
   {
      /// <summary>
      /// This method finds sum of 2 numbers
      /// </summary>
      /// <param name="a"></param>
      /// <param name="b"></param>
      /// <returns>Sum</returns>
```

```
        public int Sum(int a,int b)
        {
            return a + b;
        }
        /// <summary>
        /// This method find the Difference
        /// </summary>
        /// <param name="a"></param>
        /// <param name="b"></param>
        /// <returns>Difference</returns>
        public int Diff(int a, int b)
        {
            return a - b;
        }
    }
    //child class
    class B : A //B is childclass and A is parent class
    {
        /// <summary>
        /// This method find Product
        /// </summary>
        /// <param name="a"></param>
        /// <param name="b"></param>
        /// <returns>Product</returns>
        public int Product(int a, int b)
        {
            return a* b;
        }
    }
    internal class Program
    {
        static void Main(string[] args)
        {
            // Object creation
            B b = new B();
            Console.WriteLine("Product is:");
            Console.WriteLine(b.Product(1,2));
            Console.WriteLine("Sum is:");
            Console.WriteLine(b.Sum(2, 4));
            Console.WriteLine("Difference is:");
            Console.WriteLine(b.Diff(5,1));

            Console.ReadLine();
        }
    }
}
```

Output:



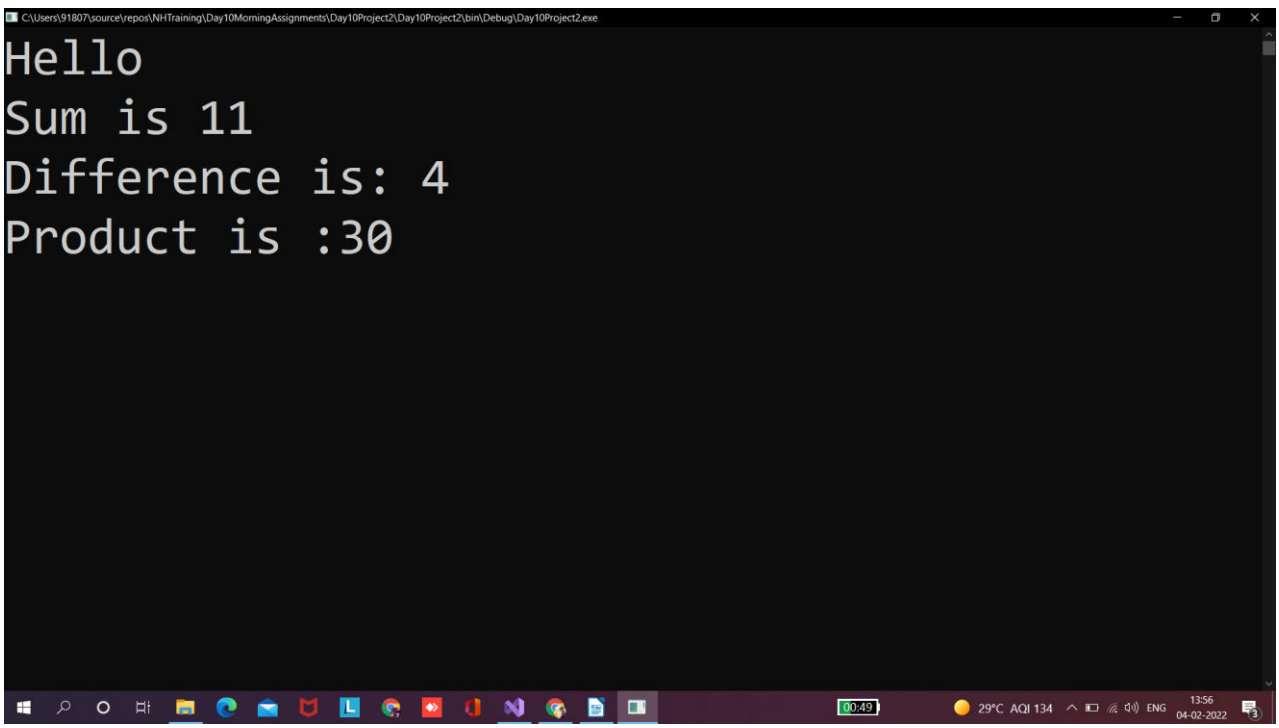
b.)Multi-level Inheritance

Code:

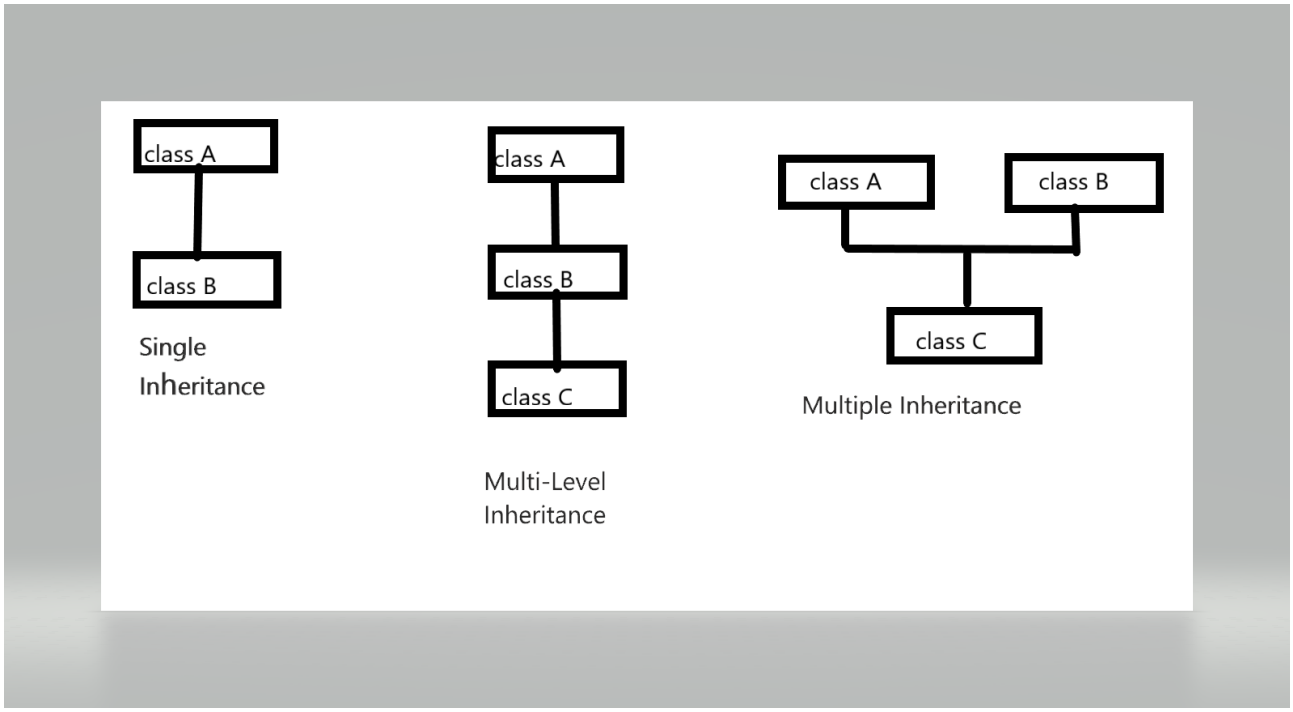
```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace Day10Project2
{
   //Author :Rc
   /* Multilevel Inheritance*/
   class A //parent class
   {
      /// <summary>
      /// This method find sum
      /// </summary>
      /// <param name="a"></param>
      /// <param name="b"></param>
      public void Sum(int a,int b)
      {
         Console.WriteLine("Sum is {0} ",a+b);
      }
      /// <summary>
      /// This method find Difference
      /// </summary>
      /// <param name="a"></param>
      /// <param name="b"></param>
```

```
        public void Dif(int a,int b)
        {

            Console.WriteLine("Difference is: {0}",a-b);

        }
    }
    class B : A //B is child class and A is parent class
    {
        /// <summary>
        /// This method Find Product
        /// </summary>
        /// <param name="a"></param>
        /// <param name="b"></param>
        public void Product(int a,int b)
        {

            Console.WriteLine("Product is :{0}",a*b);

        }
    }
    class C : B //C is child class and B is parent class
    {
        /// <summary>
        /// This method used to print the data
        /// </summary>
        public void PrintHi()
        {
            Console.WriteLine("Hello");
        }
    }
    internal class Program
    {
        static void Main(string[] args)
        {
            //object creation
            C c = new C();
            c.PrintHi();
            c.Sum(5, 6);
            c.Dif(6,2);
            c.Product(5, 6);

            Console.ReadLine();
        }
    }
}
```

Output:



3.Pictorial view of Types of Inheritance.



4.Why Multiple Inheritance is not supported for classes in C#.

- Multiple Inheritance is not supported by the classes in C#, because it causes ambiguity of methods from different parent classes.
- If still we need to implement Multiple inheritance then we atleast one interface as a base class.
- Multiple inheritance will increase complexity and developer unable to find which method is from which class.

5.What is Polymorphism.

- Polymorphism is an ability of an object to take many forms.
- Types : There are two types of Polymorphism.

1. Method Overloading
2. Method Overriding

6.Write code for Method Overloading.

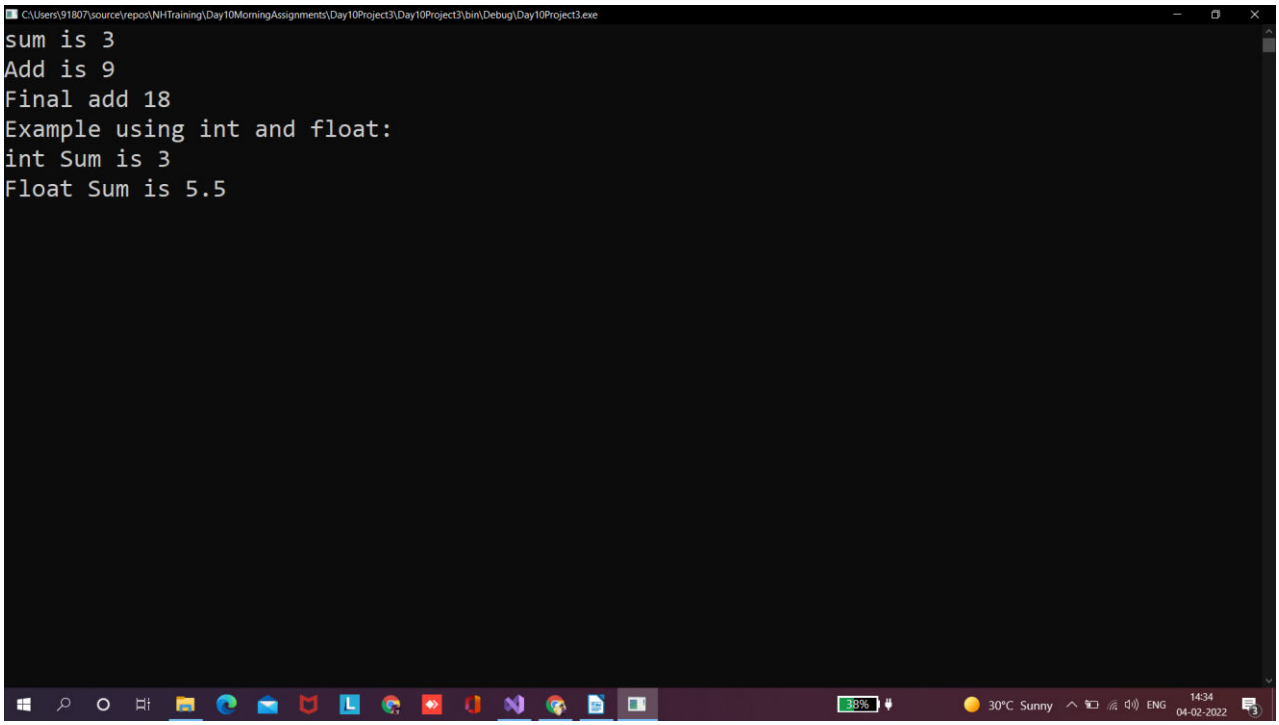
Code:

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace Day10Project3
{
   //Author:Rc
   /***Purpose:Method Overloading**/
   class MethodOverloading //class
   {
      /// <summary>
      /// This method is to find sum
      /// </summary>
      /// <param name="a"></param>
      /// <param name="b"></param>
      /// <returns>sum</returns>
      public int Add(int a, int b)
      {
         return a + b;
      }
      /// <summary>
      ///  This method is to find sum
```

```
    /// </summary>
    /// <param name="a"></param>
    /// <param name="b"></param>
    /// <param name="c"></param>
    /// <returns>sum</returns>
    public int Add(int a, int b,int c)
    {
      return a + b + c;
    }
    /// <summary>
    ///  This method is to find sum
    /// </summary>
    /// <param name="a"></param>
    /// <param name="b"></param>
    /// <param name="c"></param>
    /// <param name="d"></param>
    /// <returns>sum</returns>
    public int Add(int a, int b,int c,int d)
    {
       return a + b +c + d;
    }
}
class MethodOverloading2//class 2 creation
{
    /// <summary>
    ///  This method is to find sum
    /// </summary>
    /// <param name="a"></param>
    /// <param name="b"></param>
    /// <returns>sum</returns>
    public int Add(int a,int b)
    {
       return a + b;
    }
    /// <summary>
    ///  This method is to find sum
    /// </summary>
    /// <param name="a"></param>
    /// <param name="b"></param>
    /// <returns>sum</returns>
    public float Add(float a,float b)
    {
       return a + b;
    }
}
internal class Program
{
    static void Main(string[] args)
    {
       //Object creation for class1
       MethodOverloading m= new MethodOverloading();
       Console.WriteLine("sum is {0}",m.Add(1, 2));
```

```
            Console.WriteLine("Add is {0}",m.Add(2, 3,4));
            Console.WriteLine("Final add {0}",m.Add(3, 4,5,6));
        //class2 object creation
            MethodOverloading2 m2= new MethodOverloading2();
            Console.WriteLine("Example using int and float:");
            Console.WriteLine("int Sum is {0}",m2.Add(1,2));
            Console.WriteLine("Float Sum is {0}",m2.Add(2.5f, 3.0f));

            Console.ReadLine();
        }
    }
}
```

Output:



7.Write code for Method Overriding using new Keyword.

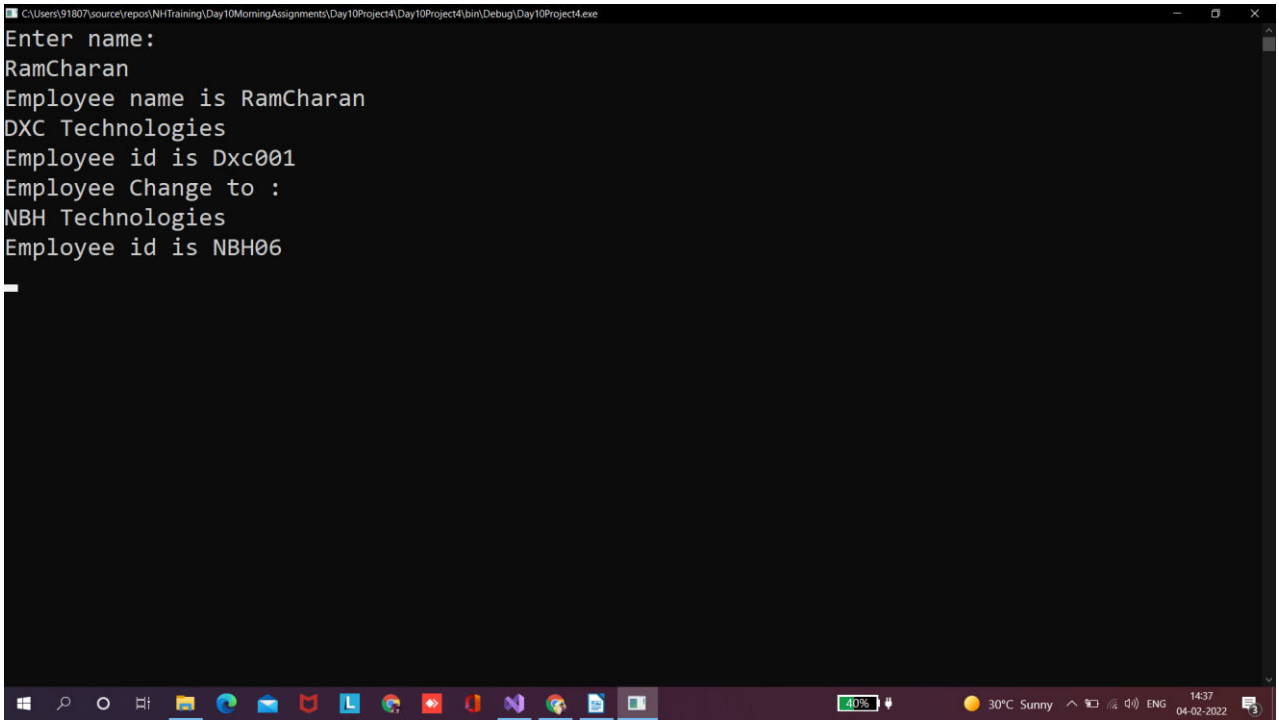
Code:

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace Day10Project4
{
    //Author: Rc
    /*****Purpose:Method Overriding*****/
    class A //parent class
```

```
{
    string name;
    /// <summary>
    /// This method reads the data
    /// </summary>

    public void Reademp()
    {

        Console.WriteLine("Enter name:");
        name=Console.ReadLine();
    }
    /// <summary>
    /// This method prints data
    /// </summary>
    public void Printemp()
    {
        Console.WriteLine("Employee name is {0}",name);
    }
    /// <summary>
    /// This method is name of the company
    /// </summary>
    public void Company()
    {
        Console.WriteLine("DXC Technologies");
    }
    /// <summary>
    /// This method is Id of the employee
    /// </summary>
    public void ID()
    {
        string id ="Dxc001";
        Console.WriteLine("Employee id is {0}",id);
    }
}
class B : A//B is sub class and A is super class
{
    /// <summary>
    /// This method is name of the company
    /// </summary>
    public new void Company()
    {
        Console.WriteLine("NBH Technologies");
    }
    /// <summary>
    /// This method is Id of the employee
    /// </summary>
    public new void ID()
    {
        string id = "NBH06";
        Console.WriteLine("Employee id is {0}", id);
    }
```

```
    }
    internal class Program
    {
        static void Main(string[] args)
        {
            //object creation
            A a = new A();
            a.Reademp();
            a.Printemp();
            a.Company();
            a.ID();
            //object creation
            B b = new B();
            Console.WriteLine("Employee Change to :");
            b.Company();
            b.ID();

            Console.ReadLine();
        }
    }
}
```

Output:



8.Write Method Overriding using virtual and override keyword.

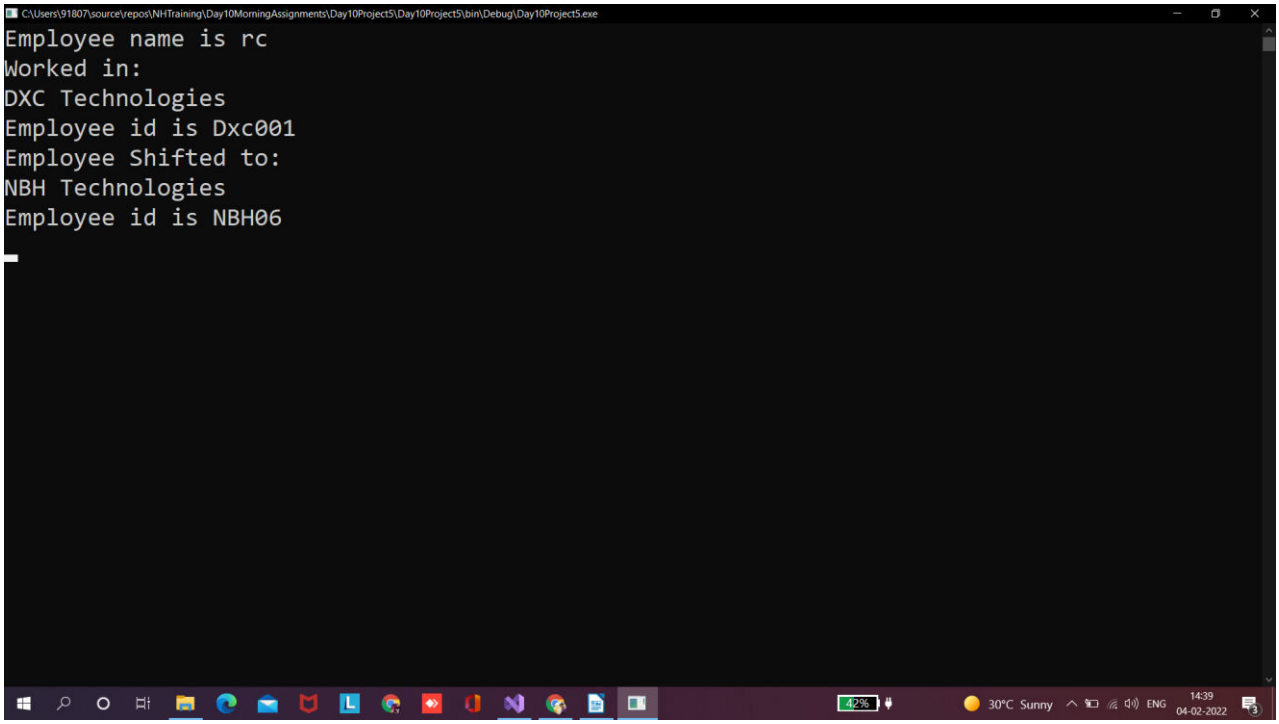
Code:

```
using System;
```

```
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace Day10Project5
{
   //Author: Rc
   /*****Purpose:Method Overriding using virtual and override *****/
   class A //parent class
   {
      /// <summary>
      /// This method is details of employee
      /// </summary>
      public void Employee()
      {
         string name = "rc";
         Console.WriteLine("Employee name is {0}",name);
      }
      /// <summary>
      /// This method is name of the company
      /// </summary>
      public virtual void Company()
      {
         Console.WriteLine("DXC Technologies");
      }
      /// <summary>
      /// This method is Id of the employee
      /// </summary>
      public virtual void ID()
      {
         string id = "Dxc001";
         Console.WriteLine("Employee id is {0}", id);
      }
   }
   class B : A//B is sub class and A is super class
   {
      /// <summary>
      /// This method is name of the company
      /// </summary>
      public override void Company()
      {
         Console.WriteLine("NBH Technologies");
      }
      /// <summary>
      /// This method is Id of the employee
      /// </summary>
      public override void ID()
      {
         string id = "NBH06";
         Console.WriteLine("Employee id is {0}", id);
      }
```

```
    }
    internal class Program
    {
        static void Main(string[] args)
        {
            //object creation
            A a = new A();
            a.Employee();
            Console.WriteLine("Worked in:");
            a.Company();
            a.ID();
            //object creation
            B b = new B();
            Console.WriteLine("Employee Shifted to:");
            b.Company();
            b.ID();

            Console.ReadLine();
        }
    }
}
```

Output:



END