Feb 4th Morning Assignment

By Surya Teja Chandolu

1. **Write the two points discussed about inheritance in the class.**

- Inheritance is the process of reusing parent class methods in the child class.
- Inheritance will remove duplicate code.
- Reusability

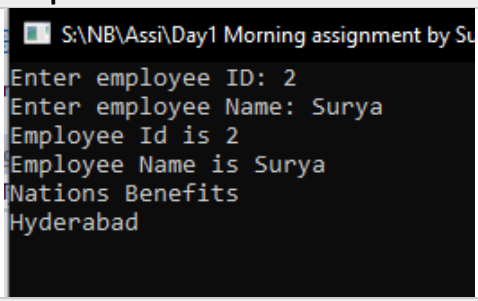
2. **Write example code for:**
   a. **Single inheritance**
   b. **Multi level inheritance**

Code Single Inheritance:

```
using System;

/*************************************************************************
* Author: Surya Teja
* Purpose: Write example code for:
       a. Single inheritance
       b. Multi level inheritance
* ************************************************************************/


namespace TypesOfInheritance
{
    /// <summary>
    /// MultiLevel Inheritance
    /// </summary>
    class Company
    {
        /// <summary>
        /// Company Name
        /// </summary>
        public void CompanyName()
        {
            Console.WriteLine("Nations Benefits");
        }
        /// <summary>
        /// Company City
        /// </summary>
        public void CompanyCity()
        {
            Console.WriteLine("Hyderabad");
        }
    }
    /// <summary>
    /// Employee class inherit Company class
    /// </summary>
    class Employee : Company
    {
        private int id;
        private string name;
        /// <summary>
```

```
        /// Read Input from user
        /// </summary>
        public void ReadEmployee()
        {
            Console.Write("Enter employee ID: ");
            id = Convert.ToInt32(Console.ReadLine());

            Console.Write("Enter employee Name: ");
            name = Console.ReadLine();
        }
        /// <summary>
        /// Employee Id
        /// </summary>
        public void EmployeeId()
        {
            Console.WriteLine($"Employee Id is {id}");
        }
        /// <summary>
        /// Employee Name
        /// </summary>
        public void EmployeeName()
        {
            Console.WriteLine($"Employee Name is {name}");
        }
    }
    internal class Program
    {
        static void Main(string[] args)
        {
            Employee emp = new Employee();
            emp.ReadEmployee();
            emp.EmployeeId();
            emp.EmployeeName();
            emp.CompanyName();
            emp.CompanyCity();

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

Output:



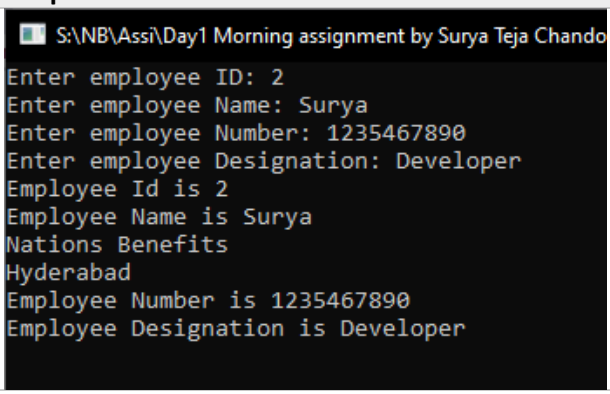
Code MultiLevel Inheritance:

```
using System;

/*************************************************************************
* Author: Surya Teja
* Purpose: Write example code for:
       a. Single inheritance
       b. Multi level inheritance
* ************************************************************************/
```

```
namespace TypesOfInheritance
{
    /// <summary>
    /// MultiLevel Inheritance
    /// </summary>
    class Company
    {
        /// <summary>
        /// Company Name
        /// </summary>
        public void CompanyName()
        {
            Console.WriteLine("Nations Benefits");
        }
        /// <summary>
        /// Company City
        /// </summary>
        public void CompanyCity()
        {
            Console.WriteLine("Hyderabad");
        }
    }
    /// <summary>
    /// Employee class inherit Company class
    /// </summary>
    class Employee : Company
    {
        private int id;
        private string name;
        /// <summary>
        /// Read Input from user
        /// </summary>
        public void ReadEmployee()
        {
            Console.Write("Enter employee ID: ");
            id = Convert.ToInt32(Console.ReadLine());

            Console.Write("Enter employee Name: ");
            name = Console.ReadLine();
        }
        /// <summary>
        /// Employee Id
        /// </summary>
        public void EmployeeId()
        {
            Console.WriteLine($"Employee Id is {id}");
        }
        /// <summary>
        /// Employee Name
        /// </summary>
        public void EmployeeName()
        {
            Console.WriteLine($"Employee Name is {name}");
        }
    }
    /// <summary>
    /// EmployeeDetails class inherit Employee class
    /// </summary>
    class EmployeeDetails : Employee
    {
        private int number;
        private string designation;
        /// <summary>
```

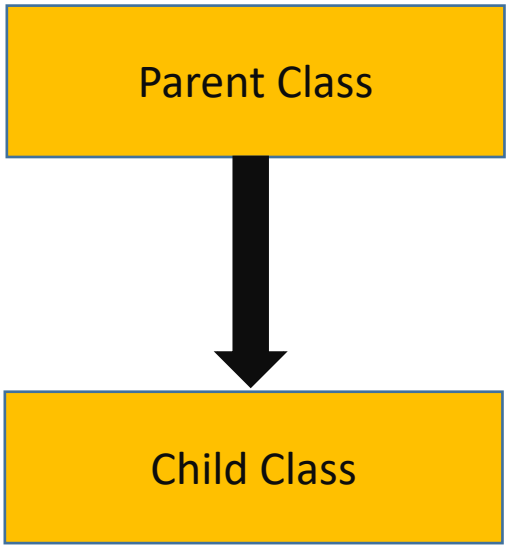
```
        /// Read Input from user
        /// </summary>
        public void ReadEmployeeD()
        {
            Console.Write("Enter employee Number: ");
            number = Convert.ToInt32(Console.ReadLine());

            Console.Write("Enter employee Designation: ");
            designation = Console.ReadLine();
        }
        /// <summary>
        /// Employee Number
        /// </summary>
        public void EmployeeNumber()
        {
            Console.WriteLine($"Employee Number is {number}");
        }
        /// <summary>
        /// Employee Designation
        /// </summary>
        public void EmployeeDesignation()
        {
            Console.WriteLine($"Employee Designation is {designation}");
        }
    }
    internal class Program
    {
        static void Main(string[] args)
        {
            Employee emp = new Employee();
            EmployeeDetails empD = new EmployeeDetails();

            emp.ReadEmployee();
            empD.ReadEmployeeD();

            emp.EmployeeId();
            emp.EmployeeName();
            emp.CompanyName();
            emp.CompanyCity();

            empD.EmployeeNumber();
            empD.EmployeeDesignation();

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

Output:

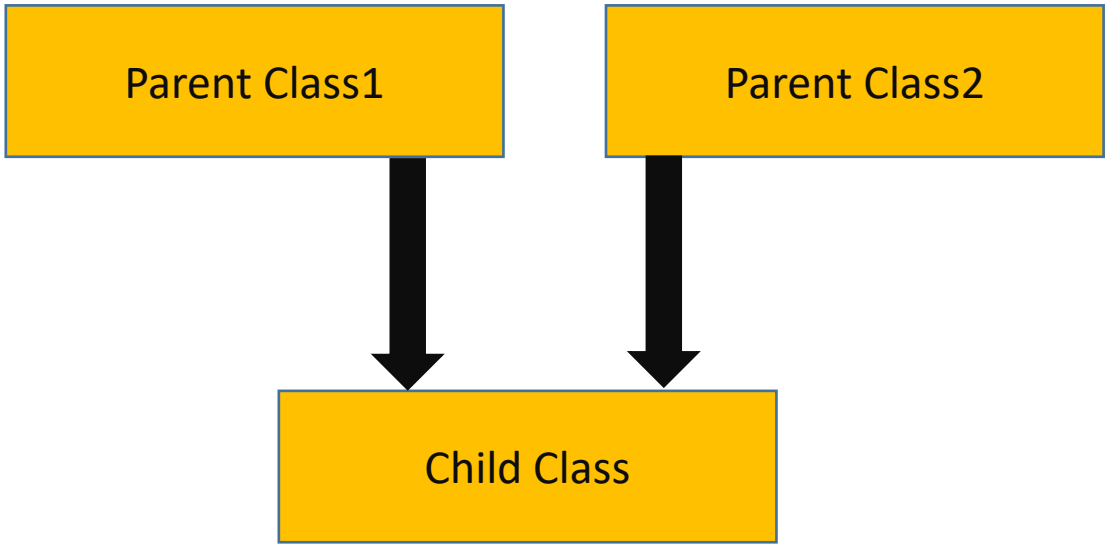


3. Pictorially represent three types of inheritance discussed in the class.

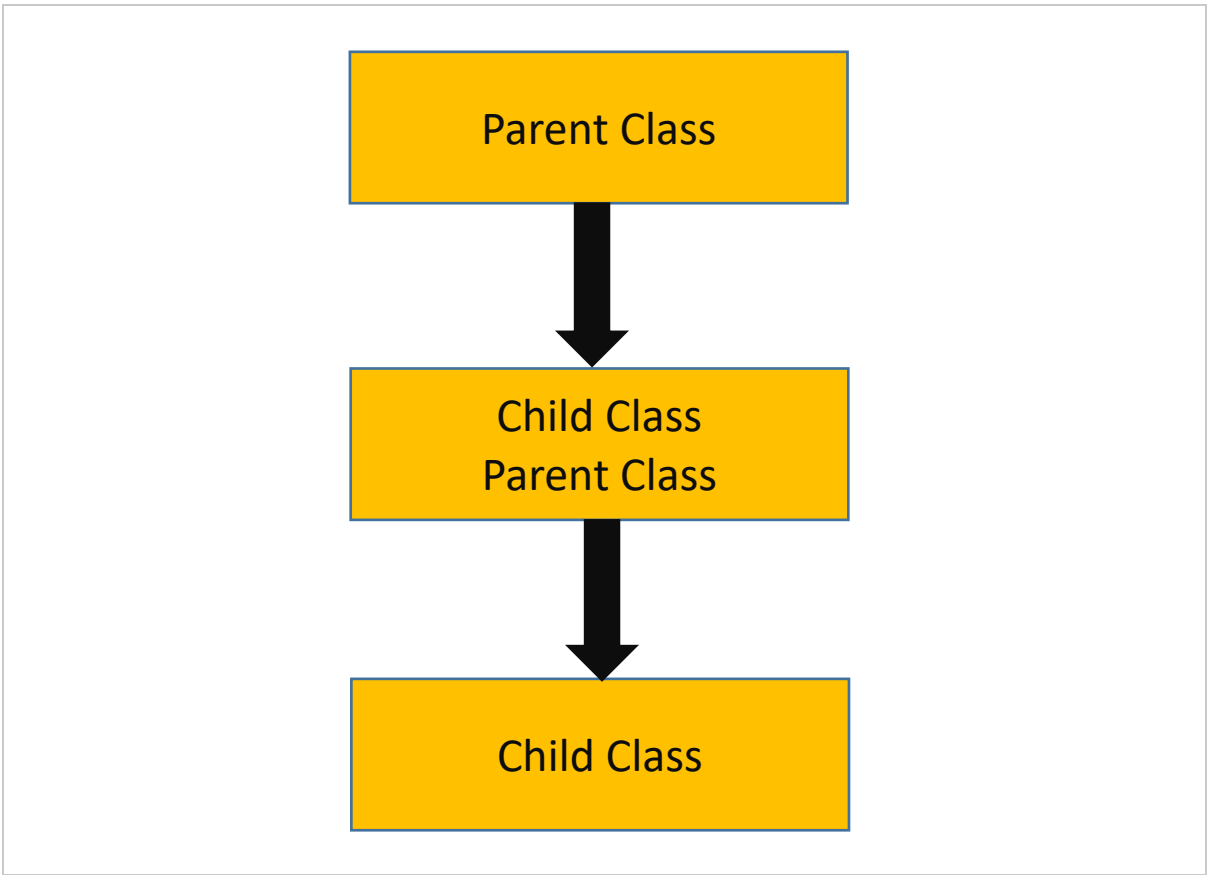
Single Inheritance



Multiple Inheritance



MultiLevel Inheritance



4. Why multiple inheritance is not supported for classes in C#

- Multiple inheritance is not supported by the C# compiler because it leads to ambiguity in methods from distinct base classes. This is due to two types of diamond shape issues. If two classes B and C are descended from A, and class D is descended from both B and C, As a result, multiple inheritance in C# is not conceivable.

5. What is polymorphism.

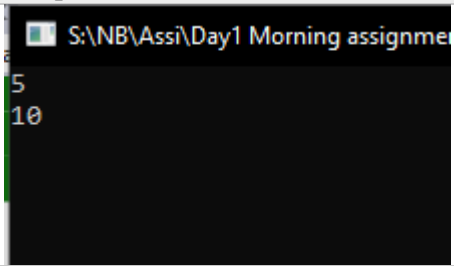
- Polymorphism is the ability of an abject to take on many forms
  Method OverLoading
  Method OverRiding
- **Method OverLoading:** Method over loading support when parameters with different size and parameters with different type irrespective of return type.
- **Method OverRiding:** If a child class to provide a specific implementation of a method that is already provided by parent classes.
- It allows the hiding of an inherited property or method. This is done using **new** keyword.

6. Write sample code for method overloading

Code:

```
using System;

/*************************************************************************
* Author: Surya Teja
* Purpose: Write sample code for method overloading
* ***********************************************************************/

namespace MethodOverLoading
{
    /// <summary>
    /// Method Over Loading
    /// </summary>
    class Operators
    {
        /// <summary>
        /// Method for adding Two numbers
        /// </summary>
        /// <param name="a"></param>
        /// <param name="b"></param>
        /// <returns></returns>
        public int Add(int a, int b)
        {
            return a + b;
        }
        /// <summary>
        /// Method for adding Three numbers
        /// </summary>
        /// <param name="a"></param>
        /// <param name="b"></param>
        /// <param name="c"></param>
        /// <returns></returns>
        public int Add(int a, int b, int c)
        {
            return a + b + c;
        }
    }
    internal class Program
    {
        static void Main(string[] args)
        {
            Operators op = new Operators();
            Console.WriteLine(op.Add(2,3));
            Console.WriteLine(op.Add(2, 3, 5));

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

Output:

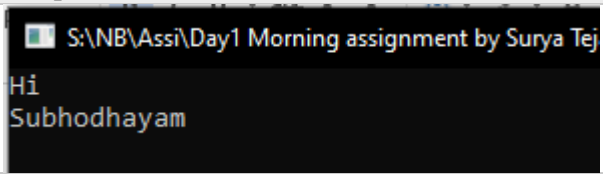


7. Write sample code for method overriding [using new key word]

Code:

```
using System;

/*************************************************************************
* Author: Surya Teja
* Purpose: Write sample code for method overriding [ using new key word ]
* ***********************************************************************/

namespace MethodOverRidingUsingNew
{
    /// <summary>
    /// Creating class English
    /// </summary>
    class English
    {
        /// <summary>
        /// Print Hi Message
        /// </summary>
        public void PrintHi()
        {
            Console.WriteLine("Hi");
        }
        /// <summary>
        /// Print Good Morning Message
        /// </summary>
        public void PrintGM()
        {
            Console.WriteLine("Good Morning");
        }
    }
    /// <summary>
    /// Creating class Telugu
    /// </summary>
    class Telugu : English
    {
        /// <summary>
        /// Print Subhodhayam Message
        /// </summary>
        public new void PrintGM()
        {
            Console.WriteLine("Subhodhayam");
        }
    }
    internal class Program
    {
        static void Main(string[] args)
        {
            Telugu t = new Telugu();
            t.PrintHi();
            t.PrintGM();

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

Output:



8. Research and write sample code for method overriding using virual, override keyword.

Code:

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

/*********************************************************************
* Author: Surya Teja
* Purpose: Research and write sample code for method overriding using virual,
override keyword.
* ********************************************************************/

namespace MethodOverRidingUsingVirtualAndOverride
{
    internal class Program
    {
        /// <summary>
        /// Creating class English
        /// </summary>
        class English
        {
            /// <summary>
            /// Print Hi Message
            /// </summary>
            public void PrintHi()
            {
                Console.WriteLine("Hi");
            }
            /// <summary>
            /// Print Good Morning Message
            /// </summary>
            public virtual void PrintGM()
            {
                Console.WriteLine("Good Morning");
            }
        }
        /// <summary>
        /// Creating class Telugu
        /// </summary>
        class Telugu : English
        {
            /// <summary>
            /// Print Subhodhayam Message
            /// </summary>
            public override void PrintGM()
            {
                Console.WriteLine("Subhodhayam");
            }
        }
        static void Main(string[] args)
        {
            Telugu t = new Telugu();
            t.PrintHi();
            t.PrintGM();

            Console.ReadLine();
        }
    }
```

```
}
```

Output: