# DAY10 MORNING ASSIGNMENT
# BY
# CH. PAVAN KUMAR REDDY (04-02-2022)

**Q1). Write the two points discussed about inheritance in the class.**

**ANSWER:**

1) It is a process of reusing Base class method in the Derived class or Parent class in the Child class.

2) Re-usability is the main concept of Inheritance and to remove the duplicate code.

3) Three types of inheritance. They are.
- SINGLE INHERITANCE
- MULTI-LEVEL INHERTANCE
- MULTIPLE INHERITANCE (Not used in c#)

**Q2). Write example code for: a. Single inheritance b. Multi level inheritance**

**ANSWER:**

**SINGLE INHERITANCE:**

When only one Child class inherits one Parent class, it is known as Single Inheritance.

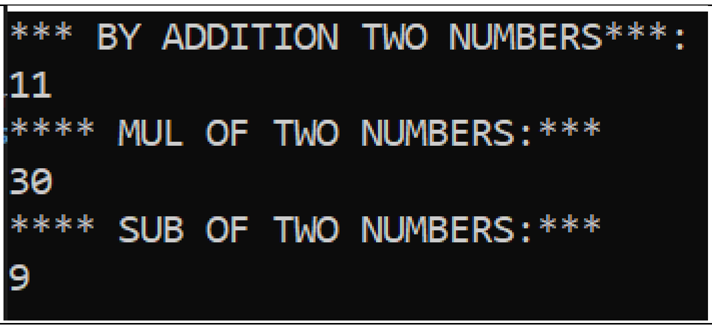
**CODE:**

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

namespace Day10Project1
{
    /// <summary>
    /// SINGLE INHERITANCE
    /// DONE BY: PAVAN
    /// </summary>
    class Algebra
    {
        public int Add(int a, int b)
        {
            return a + b;
        }
        public int sub(int a, int b)
        {
            return a - b;
        }
    }

    class TotalMaths : Algebra
    {
        public int mul(int a, int b)
        {
            return a * b;
        }
    internal class Program
    {
            static void Main(string[] args)
            {
                TotalMaths tm = new TotalMaths();
                Console.WriteLine("*** BY ADDITION TWO NUMBERS***:");
                Console.WriteLine(tm.Add(5, 6));
                Console.WriteLine("**** MUL OF TWO NUMBERS: ***");
                Console.WriteLine(tm.mul(5,6));
                Console.WriteLine("**** SUB OF TWO NUMBERS: ***");
                Console.WriteLine(tm.sub(15, 6));
                Console.ReadLine();
            }
        }
    }
}
```

**OUTPUT:**



```
*** BY ADDITION TWO NUMBERS***:
11
**** MUL OF TWO NUMBERS:***
30
**** SUB OF TWO NUMBERS:***
9
```

**MULTI-LEVEL INHERITANCE:**

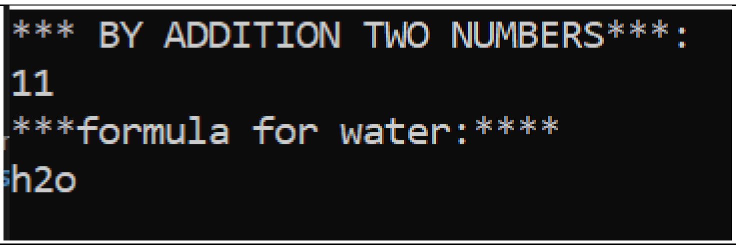
When a derived class is inherited by base class and this base class is again inherited by another derived class, it is known as Multi-Level Inheritance.

**CODE:**

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

namespace Day10Project2
{
    class Algebra
    {
        public int Add(int a, int b)
        {
            return a + b;
        }
        public int sub(int a, int b)
        {
            return a - b;
        }
    }
    class TotalMaths : Algebra
    {
        public int mul(int a, int b)
        {
            return a * b;
        }
    }
    class AllOperations : TotalMaths
    {
        public string water()
        {
            return "h2o";
        }
    }
    internal class Program
    {
        static void Main(string[] args)
        {
            AllOperations obj = new AllOperations();
            Console.WriteLine("*** BY ADDITION TWO NUMBERS***:");
            Console.WriteLine(obj.Add(5, 6));
            Console.WriteLine("***formula for water: ****");
            Console.WriteLine(obj.water());
            Console.ReadLine();
        }
    }
}
```

**OUTPUT:**

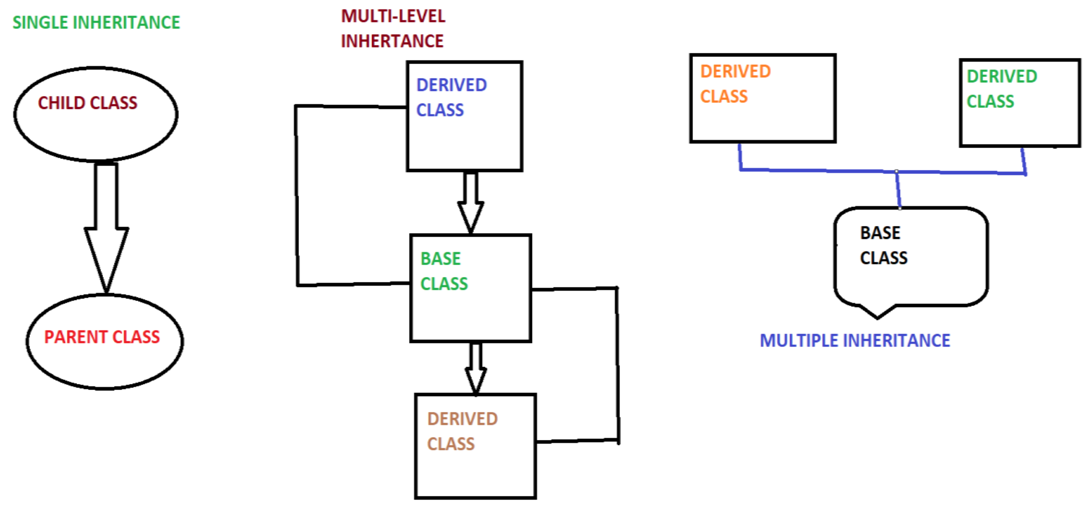


```
*** BY ADDITION TWO NUMBERS***:
11
***formula for water:****
h2o
```

**Q3). Pictorially represents 3 types of inheritance discussed in the class.**

There are 3 types of Inheritance. They are.

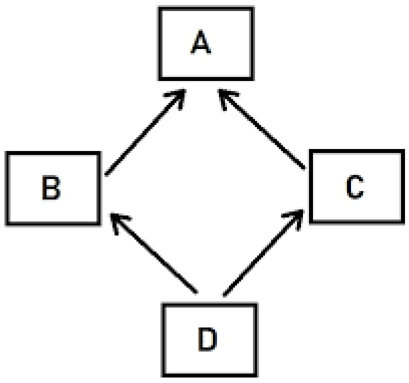
- SINGLE INHERITANCE
- MULTI-LEVEL INHERITANCE
- MULTIPLE INHERITANCE



**Q4). Why multiple inheritance is not supported for classes in C#.**

- We don't consider Multiple Inheritance in c# because it causes ambiguity of methods from different base class.
- The problem is that the compiler/runtime cannot figure out what to do if we have same parameters in the like int (int a, int b) and float (int a, int b) .
- This Multiple Inheritance causes **DIAMOND PROBLEMS.**
- The diamond problem is an ambiguity that arises when two classes B and C inherit from A, and class D inherits from both B and C. ... It is called the diamond problem.

- To overcome this multiple inheritance ambiguity in c# we use INTERFACE



concept.

Q5). What is polymorphism?

- The ability of an object to take many forms is defined as **POLYMORPHISM.**
- These are of two types. They are
  A) METHOD OVERLOADING
  B) METHOD OVERRIDING

❑ METHOD OVERLOADING:

Method overloading is to use multiple methods within the same class with different parameters irrespective of return type.

```
        public class Method overloading

    public int add (int a, int b)
{
        return a + b;
}
    public int add(int a, int b,int c)
```

```
{
        return a + b + c;
}
```

❑ METHOD OVERRIDING:

Method overriding is used to modify or re-write the data in the same class when it is inherited.

- Method overriding is only possible in derived classes, not within the same class where the method is declared
- Base class must use the NEW keywords to declare a method. Then only can a method be overridden.

```
        public class Account
{
public int balance()
{
  return 10;
}
}
   public class Amount: Account
{
      public new int balance()
 {
         return 500;
            }
```

**Q6). Write sample code for method overloading**

**CODE:**

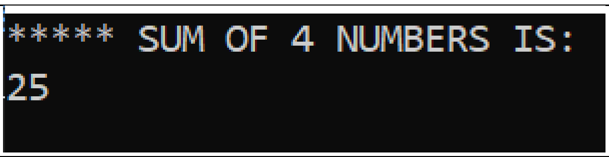
```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace Day10Project3
{
/// <summary>
   /// METHOD OVERLOADING
   /// DONE BY: PAVAN
   /// </summary>
   class Algebra
   {
      public int Add(int a, int b)
```

```
        {
            return a + b;
        }
        public int Add(int a, int b, int c)
        {
            return a + b + c;
        }
        public int Add(int a, int b, int c, int d)
        {
            return a + b + c + d;
        }
    }

    internal class Program
    {
        static void Main (string [] args)
        {
            Algebra obj = new Algebra();
            Console.WriteLine(obj.Add(4 ,6,7,8));
            Console.ReadLine();
        }
    }
}
```

**OUTPUT:**



```
***** SUM OF 4 NUMBERS IS:
25
```

**Q7). Write sample code for method overriding [using new key word]**

**CODE:**

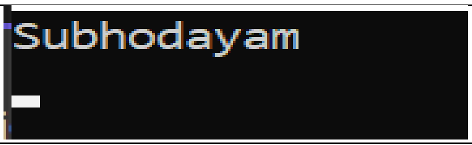
```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace Day10Project4
{
/// <summary>
/// METHOD OVERRIDING//
/// DONE BY: PAVAN
/// </summary>
    class ENGLISHMESSAGE
    {
        public void PrintHI()
        {
           Console.WriteLine("HI");
              }
        public void PrintPavan()
        {
           Console.WriteLine("Pavan");
        }
        public void PrintGM()
        {
           Console.WriteLine("GOOD MORNING");
        }
        }
    class TELUGUMESSAGE : ENGLISHMESSAGE
    {
        public new void PrintGM()
        {
           Console.WriteLine("Subhodayam");

        }
    }
        internal class Program
    {
        static void Main(string[] args)
        {
           TELUGUMESSAGE obj = new TELUGUMESSAGE();
          obj.PrintGM();
           Console.ReadLine();

        }
    }
}
```

**OUTPUT:**



```
Subhodayam
```

**Q8). Research and write sample code for method overriding using virual, override keyword.**

**CODE:**

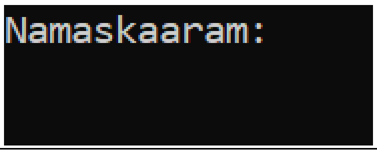
```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace Day10Project5
{
  /// <summary>
  /// OVERRIDING METHOD USING VIRTUAL KEYS//
  /// DONE BY: PAVAN
  /// </summary>
  class ENGLISHMESSAGE
  {
    public virtual void PrintHI()
    {
      Console.WriteLine("HI");
    }
    public virtual void PrintGM()
    {
      Console.WriteLine("GOOD MORNING");

    }
  }
  class TELUGU : ENGLISHMESSAGE
  {
    public override void PrintGM()
    {
      Console.WriteLine("Namaskaaram:");
    }
  }
    internal class Program
    {
      static void Main(string[] args)
      {
      TELUGU obj = new TELUGU();
      obj.PrintGM();
      Console.ReadLine();

      }
    }
  }
```

**OUTPUT:**



```
Namaskaaram:
```