|  |
| --- |
| Day10 Morning Assignment  By  Anusha Bellala |

|  |
| --- |
| 1. Write the two points discussed about inheritance in the class. |
| Inheritance is the process of re-using base class methods in the derived class.   * Inheritance main goal is: Re-usability * Remove duplicate code. |

|  |
| --- |
| 2.)Write Example code:  i)Single Inheritance |
| Code: |
| using System;  using System.Collections.Generic;  using System.Linq;  using System.Text;  using System.Threading.Tasks;  namespace Inheritance  {  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(tm.Add(4,5));  Console.WriteLine(tm.Sub(10,5));  Console.WriteLine(tm.Mul(4,5));  Console.ReadLine();  }  }  } |
| Output: |

|  |
| --- |
| ii)Multilevel Inheritance |
| Code: |
| using System;  using System.Collections.Generic;  using System.Linq;  using System.Text;  using System.Threading.Tasks;  namespace Inheritance  {  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 AllSubjects : TotalMaths  {  public string Methane()  {  return"CH4";  }  }  internal class Program  {  static void Main(string[] args)  {  AllSubjects obj=new AllSubjects();  Console.WriteLine (obj.Add(4, 5));  Console.WriteLine(obj.Sub(9,7));  Console.WriteLine(obj.Mul(6,8));  Console.WriteLine(obj.Methane());  Console.ReadLine();  }  }  } |
| Output: |

|  |
| --- |
| 3. Pictorially represent 3 types of inheritance discussed in the class. |
| C:\Users\ANUSHA\OneDrive\Desktop\typesofinheritance.jpg |

|  |
| --- |
| 4. Why multiple inheritance is not supported for classes in C#. |
| * C# compiler is designed not to support multiple inheritance because it causes ambiguity of methods from base class. This is cause by Diamond shape problem of two classes. * C# does not support multiple inheritance , because they reasoned that adding multiple inheritance added too much complexity.   http://3.bp.blogspot.com/-pDOmliP7miA/Td6r5G94IsI/AAAAAAAAAGc/S69vr0wPUSc/s1600/Multiple+Class+Inheritance.png  As shown in the image above: **1.** I have 2 classes - **ClassB** and **ClassC.** **2.** Both of these classes inherit from **ClassA.** **3.** Now, we have another class, **ClassD** which inherits from both **ClassB** and **ClassC.**  So, if a method in **ClassD** calls a method defined in **ClassA** and **ClassD**has not overriden the invoked method. But both **ClassB**and **ClassC**have overridden the same method differently. Now, the ambiguity is, from which class does, **ClassD**inherit the invoked method: **ClassB**, or **ClassC**?  **In order not to have these problems, C# does not support multiple class inheritance.** |

|  |
| --- |
| 5. What is polymorphism. |
| Polymorphism is the ability of an object to take on many forms.  Two types of polymorphism:  1.Method Overloading.  2.Method Overriding. |

|  |
| --- |
| 6. Write sample code for method overloading. |
| Code: |
| using System;  using System.Collections.Generic;  using System.Linq;  using System.Text;  using System.Threading.Tasks;  namespace Polymorphism  {  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 a = new Algebra();  Console.WriteLine(a.Add(1, 2));  Console.WriteLine(a.Add(2, 3, 4));  Console.WriteLine(a.Add(3, 4, 5, 6));  Console.ReadLine();  }  }  }  Output: |
|  |

|  |
| --- |
| 7. 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 MethodOverriding  {  class EnglishMessage  {  public void PrintHi()  {    Console.WriteLine("Hi");  }  public void PrintHello()  {  Console.WriteLine("Hello");  }  public void PrintGM()  {  Console.WriteLine("Good Morning");  }  }  class TeluguMessage : EnglishMessage  {    public new void PrintGM()  {  Console.WriteLine("Subodhayam");  }  }  internal class Program  {  static void Main(string[] args)  {  tm.PrintGM();  tm.PrintHi();  tm.PrintHello();  Console.ReadLine();  }  }  } |
| Ouput: |

|  |
| --- |
| 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;  namespace VirtualKeyword  {  // Base Class  class Parent  {  public virtual void SayHello()  {  Console.WriteLine("Hello from Parent class Function!!!");  }  }  class Child : Parent  {  public override void SayHello()  {  Console.WriteLine("Hello from Child class function!!!");    }  }  internal class Program  {  static void Main(string[] args)  {  Parent pObj = new Parent();  pObj.SayHello();  Child cObj = new Child();  cObj.SayHello();  Parent pcObj = new Child();  pcObj.SayHello();    Console.ReadLine();  }  }  } |
| Output: |