**PRACTICAL # 10**

**OBJECT:**

Understanding Polymorphism in C#

**THEORY:**

Polymorphism is one of the core concepts of object oriented programming. Polymorphism provides an ability for the classes to implement different methods that are called through the same name and it also provides an ability to invoke the methods of derived class through base class reference during runtime. C# has two different kinds of polymorphism:

• Compile Time Polymorphism

• Run Time Polymorphism

Compile Time Polymorphism

Compile Time Polymorphism, also known as early binding or static binding, defines multiple methods with same name but with different parameters. By using compile time polymorphism, we can perform different tasks with same method name by passing different parameters. The compile time polymorphism is achieved using method overloading.

Example:

*using System;*

*namespace Program*

*{*

*public class Calculate*

*{*

*public void AddNumbers(int a, int b)*

*{*

*Console.WriteLine("a + b = {0}", a + b);*

*}*

*public void AddNumbers(int a, int b, int c)*

*{*

*Console.WriteLine("a + b + c = {0}", a + b + c);*

*}*

*}*

*class Program*

*{*

*static void Main(string[] args)*

*{*

*Calculate c = new Calculate();*

*c.AddNumbers(1, 2);*

*c.AddNumbers(1, 2, 3);*

*Console.WriteLine("\nPress Enter Key to Exit..");*

*Console.ReadLine();*

*}*

*}*

*}*

Output:

*8*

*100*

*90*

Run Time Polymorphism

Dynamic or runtime polymorphism is also known as late binding. Here, the method name and the method signature (number of parameters and parameter type must be the same and may have a different implementation). Method overriding is used to achieve dynamic polymorphism. Method overriding can be done using inheritance. With method overriding it is possible for the base class and derived class to have the same method name with different implementation.

Example:

*using System;*

*public class Animal{*

*public virtual void eat(){*

*Console.WriteLine("Eating...");*

*}*

*}*

*public class Buf: Animal*

*{*

*public override void eat()*

*{*

*Console.WriteLine("Eating grass...");*

*}*

*}*

*public class TestOverriding*

*{*

*public static void Main()*

*{*

*Bug bf = new Buf();*

*bf.eat();*

*}*

*}*

Output:

*Eating grass…*

**ACTIVITIES**

**Activity 1**

Design an application with base class vehicle and child classes car and truck. Use polymorphism to define a method called drive in each class with different implementation.

**REVIEW QUESTIONS**

1. What is polymorphism?
2. Differentiate between compile-time and Runtime Polymorphism in C#?
3. What is method overriding?