Experiment no. 5

Title: Implement program on Encapsulation and function overloading

Theory:

Introduction:

Encapsulation and function Overloading are important concepts in object-oriented programming that enables developers to write more efficient, robust, and reusable code. Encapsulation is a mechanism that binds data and functions that manipulate that data together into a single unit. It helps to prevent unauthorized access to data and ensures that data is accessed and modified only through the defined methods. function overloading is a feature that allows a class to have multiple methods with same name but different parameters.

Encapsulation:-

Encapsulation is one of function fundamental principles of object oriented-programming that promotes data hiding and abstraction. It refers to process of binding data and function that manipulate that data together into a single unit, called a class. The data is kept private within the class, and function are define as public methods that allows access to data.

Encapsulation is achieved in C# by using access modifiers. The encapsulation data can be accessed and modified only through public methods defined in the class. The public access modifier allows access to data and functions from anywhere in program, while private access modifiers restricts access to only within the class.

DATE : / /

Syntax : e.g.

```
class Myclass.
{ private int myVar;          // private data member
  public int MyProperty;      // private property.
  { get { return myVar; }
    set { myVar = value; }
  }
  public void MyMethod()      // public method.
  { // code to manipulate myVar }
}
```

function Overloading :

function overloading is a feature in C# that allows a class to have multiple methods with same name but different parameters. This means that two or more methods in class can have same name, but they must differ in number, order, or type of their parameters.

function overloading helps to make code more readable and reusable by allowing developer to use same method name for similar operation. It also helps to avoid naming conflicts and reduces no of method names in a program.

Syntax : e.g.

```
class Myclass {
  public void MyMethod(int x)
  { // code to manipulate integer x }
  public void MyMethod(string s)
  { // code to manipulate strings }
}
```

SIGN. :

J.J.M.C.O.E.

Conclusion:

In this practical we have learn how Encapsulation and function Overloading are important in OOP that help developer to write more efficient, robust and reusable code.