Class & Object

Lecture - 7

Tokey Ahmmed

Lecturer, Dept. of CSE

Varendra University

tokey@vu.edu.bd

Class

- **Class** - A class can be defined as a template/blue print that describes the behaviors/states that object of its type support
- Class defines structure and behavior (data & code) that will be shared by a set of objects

Example

```
class MyClass { }
```

Object

- An object is a region of storage that defines both state & behavior.
  - State is represented by a set of variables & the values they contain.
  - Behavior is represented by a set of methods & the logic they implement.
- Thus, an object is a combination of a data & the code that acts upon it.
- Objects are the basic runtime entities in an object-oriented system.
- Objects are instance of a class.

Example:

```
person p1,p2;
p1 = new person();
p2 = new person();
```

Constructors

- A constructor in Java is a special method that is used to initialize objects. The constructor is called when an object of a class is created.
  - A constructor with no parameters is referred to as a *no-arg constructor*.
  - Constructors must have the same name as the class itself.
  - Constructors do not have a return type - not even void.
  - Constructors are invoked using the new operator when an object is created.
  - Constructors play the role of initializing objects.

```
class Person{
        private int Name;
        private int Age;

        public Person(String name, int age){
             Name = name;
             System.out.println ("Name is: "+ name );
        }
        public void setAge(int age ){
                Age = age;
        }
        public int getAge(){
                return Age;
        }
}
```

```
public class MainClass {
    public static void main(String[] args) {
              /* Object creation */
              Person p1 =new Person(“John");
              p1.setAge(25);
              System.out.println("Age is: "+p1.getAge() );
     }
}
```

```
Output:
Name is: John
Age is: 25
```

Constructor Overloading

- Constructors are methods that can be overloaded, just like any other method in a class.
- The constructor overloading can be defined as the concept of having more than one constructor with different parameters so that every constructor can perform a different task.
- The compiler differentiates these constructors by taking into account the number of parameters in the list and their type.

```
public class MyClass
{
    int x;
    MyClass(){
        System.out.println("Inside MyClass()
        constructor.");
        x=0;
    }

    MyClass(int i){
        System.out.println("Inside MyClass(int)
        constructor.");
        x=i;
    }

    MyClass(double d){
        System.out.println("Inside MyClass(double)
        constructor.");
        x=(int)d;
    }

    void getXvalue()
    {
        System.out.println("The value of the instance
             variable of the object  is “ +x +".");
    }
}
```

```
public class MyClassTest
{

    public static void main(String[] args)
    {
        MyClass first=new MyClass();
        MyClass second=new MyClass(52);
        MyClass third=new MyClass(13.6);
        first.getXvalue();
        second.getXvalue();
        third.getXvalue();
    }
}
```

```
Inside MyClass() constructor.
Inside MyClass(int) constructor.
Inside MyClass(double) constructor.
The value of the instance variable of the object is 0 .
The value of the instance variable of the object is 52.
The value of the instance variable of the object is 13.
```