North South University

Department of Electrical and Computer Engineering

CSE 215L (Programming Language II Lab)

Lab 5: Class and Objects

Objective:

- **Introduction to Classes and Objects:** Understand the fundamental concepts of classes and objects in Java, including the role of classes as blueprints and objects as instances of those blueprints.
- **Class Declaration and Structure:** Learn how to declare a class in Java, define class members (fields or variables, methods), and understand the structure of a class.
- **Object Instantiation:** Explore the process of creating objects from a class, emphasizing using the new keyword for instantiation.
- **Constructor Methods:** Understand the purpose of constructor methods in initializing objects during their creation and how to define various types of constructors.
- **Encapsulation:** Grasp the concept of encapsulation, which involves bundling data (fields) and methods within a class to control access and protect data integrity.
- **Instance Variables and Methods:** Learn about instance variables that store object-specific data and instance methods that perform operations on the object's data.
- **Static Members:** Understand the concept of static members (variables and methods) in a class, which are associated with the class rather than specific instances.

What is a Class:

A class is a construct that defines objects of the same type. A Java class uses variables to define data fields and methods to define behaviors. Additionally, a class provides a special type of method, known as constructors, invoked to construct objects from the class. However, every class has its own set of methods that could be used.

What is an Object:

An object has both a state and behavior. The state defines the object, and the behavior defines what the object does. The state of an object consists of a set of data fields (also known as properties) with their current values. A set of methods defines the behavior of an object. For example, for a class named TestClass, when we call the constructor for that class, we create an object of that class.

Demonstration of Class	Demonstration of Objects
`public class Circle {`	`public class Main {`

Demonstration of Class

```
public class Circle {
    private double radius;
    public Circle (){
        radius = 1;
    }

    public Circle (double r){
        radius = r;
    }

    public void setRadius(double r){
        radius = r;
    }

    public double getRadius(){
        return radius;
    }
}
```

Demonstration of Objects

```
public class Main {
    public static void main(String[] args) {
        Circle c1 = new Circle(4);
        System.out.println(c1.getRadius());

        c1.setRadius(7);
        System.out.println(c1.getRadius());
    }
}
```

Static Variables, Constants, and Methods:

Static variables are shared by all the class instances, which means these variables are not exclusive to only an INSTANCE of a class but represent the WHOLE CLASS.

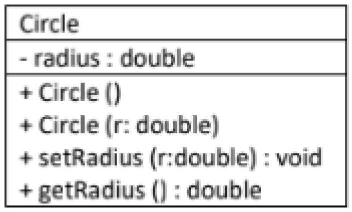
Static methods are not tied to a specific object; by this time, you already know about them as you have already used them.

Static constants are final variables shared by all the instances of the class, for this, we use final and static keywords to show that THEY CANNOT BE CHANGED.

Use the static modifier to declare static variables, constants, and methods. The UML Class diagrams represent the static variables, constants, or methods by underlines.

UML Class Diagrams:

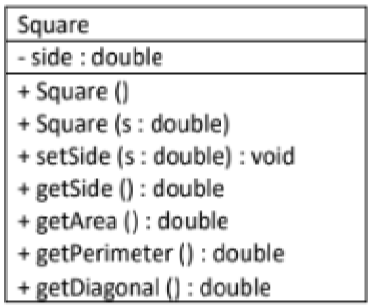
UML stands for Unified Modelling Language, and UML Class Diagrams are used to visually describe a class's data fields and methods in a rectangle. The example of a UML class diagram is as follows:



Circle
- radius : double
+ Circle ()
+ Circle (r: double)
+ setRadius (r:double) : void
+ getRadius () : double

Practice Problems:

1. Implement the following Class based on the provided UML Diagram and test the methods



```
Square
- side : double
+ Square ()
+ Square (s : double)
+ setSide (s : double) : void
+ getSide () : double
+ getArea () : double
+ getPerimeter () : double
+ getDiagonal () : double
```

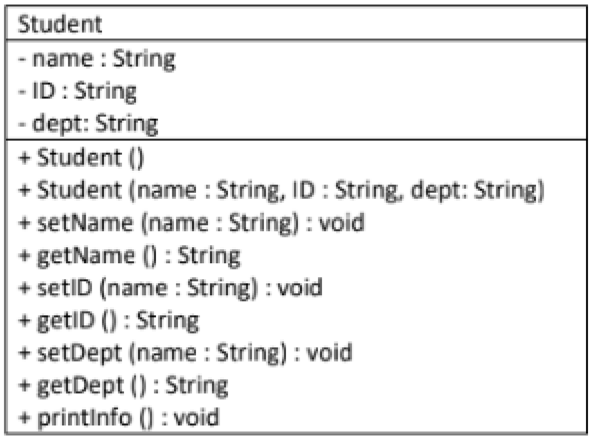
The default constructor sets the value of the square side to 1

getArea is used for finding the area of the square

getPerimeter is used for finding the perimeter of the square

getDiagonal is used for finding the length of the diagonal of the square

2. Consider the UML Class Diagram, where you must implement the following class and test the methods.



```
Student
- name : String
- ID : String
- dept: String
+ Student ()
+ Student (name : String, ID : String, dept: String)
+ setName (name : String) : void
+ getName () : String
+ setID (name : String) : void
+ getID () : String
+ setDept (name : String) : void
+ getDept () : String
+ printInfo () : void
```

3. You are tasked to make a BankAccount Class that contains the following variables:
   i. String name
   ii. String branch
   iii. double bankBalance
   iv. int numberOfAccounts (stores the number of Accounts)
   v. double totalAmount (stores the total balance of all accounts)
   vi. double avgAmount (stores the average balance of all accounts)
   a. Figure out which of the six variables should be represented by the whole class and which is exclusive to an object instance.
   b. For the BankAccount Class, create a get and set method for name, branch, and bank balance variables.
   c. Create a method that helps you get the total and average balance, however,/those methods should belong to the entire Class.
   d. Create five instances of the BankAccount Class, then print the average and total balances.
4. You are tasked to create a new Team class which has the following variables:
   i. Name of the Team
   ii. Name of the Coach
   iii. Name of the Home Country of the team
   iv. Array of Names of Players
   v. The current number of Players (hint: starts with zero upon initialization of the object)

vi. Maximum Number of Players (Must be 7)

a. You should have a constructor that takes in the names of the team, coach, and home country as parameters and the array of players gets declared with the size of a maximum number of players.
b. Then, you should have a get-and-set method for the names of the team, coach, and home country.
c. After that, you should have distinct methods to add and remove players from a team.
d. Finally, a function will be made that prints the names of the team, coach, and home country in consecutive lines, followed by printing the players' names.