Lab Number:	4
Student Name:	**Kshitij Prasad Bare**
Roll No :	**14**

Title:

4.1 Write a Java program to Create a class Student with two method getData() and printData(). getData() to get the value from the user and display the data in printData(). Create the two objects s1 ,s2 to declare and access the values from class StudentTest.

4.2 Write a Java program for Basic bank Management System

Learning Objective:

- Students will be able to write C++ and java program for using classes and objects.

Learning Outcome:

- Ability to execute a simple C++ and Java program by accepting and displaying values using functions
- Understanding the classes and objects concept in C++ and Java.

Course Outcome:

ECL304.1	Understand object-oriented programming concepts and implement using C++ and Java

Theory:

Q)Explain about Constructor?

SOLUTION: - In Java, a constructor is a block of codes similar to the method. It is called when an instance of the class is created. At the time of calling constructor, memory for the object is allocated in the memory.

It is a special type of method which is used to initialize the object.

Every time an object is created using the new () keyword, at least one constructor is called.

It calls a default constructor if there is no constructor available in the class. In such case, Java compiler provides a default constructor by default.

Faculty: Ms. Deepali Kayande

Q)Explain about classes and objects in Java?

SOLUTION: - A class is a user defined blueprint or prototype from which objects are created. It represents the set of properties or methods that are common to all objects of one type. In general, class declarations can include these components, in order:

1. **Modifiers**: A class can be public or has default access
2. **Class keyword:** class keyword is used to create a class.
3. **Class name:** The name should begin with an initial letter (capitalized by convention).
4. **Superclass (if any):** The name of the class's parent (superclass), if any, preceded by the keyword extends. A class can only extend (subclass) one parent.
5. **Interfaces (if any):** A comma-separated list of interfaces implemented by the class, if any, preceded by the keyword implements. A class can implement more than one interface.
6. **Body:** The class body surrounded by braces, { }.

Object is a basic unit of Object-Oriented Programming and represents the real life entities. A typical Java program creates many objects, which as you know, interact by invoking methods. An object consists of :

1. **State**: It is represented by attributes of an object. It also reflects the properties of an object.
2. **Behavior**: It is represented by methods of an object. It also reflects the response of an object with other objects.
3. **Identity**: It gives a unique name to an object and enables one object to interact with other objects.
Example of an object: dog

Q) How to access class attributes and methods? Explain with example?

SOLUTION: - Any variable that is bound in a class is a *class attribute.* Any function defined within a class is a *method.* Methods receive an instance of the class, conventionally called `self`, as the first argument. For example, to define some class attributes and methods, you might enter the following code:

```
class MyClass

    attr1 = 10        #class attributes

    attr2 = "hello"


    def method1(self):

        print MyClass.attr1 #reference the class attribute


    def method2(self):

        print MyClass.attr2 #reference the class attribute


    def method3(self, text):

        self.text = text        #instance attribute

        print text, self.text #print my argument and my attribute


    method4 = method3 #make an alias for method3
```

Inside a class, you should qualify all references to class attributes with the class name; for example, `MyClass.attr1`. All references to instance attributes should be qualified with the `self` variable; for example, `self.text`. Outside the class, you should qualify all references to class attributes with the class name (for example `MyClass.attr1`) or with an instance of the class (for example `x.attr1`, where `x` is an instance of the class). Outside the class, all references to instance variables should be qualified with an instance of the class; for example, `x.text`.

4.2- Input given: a=3, b=1

OUTPUT