JAVA PROGRAMMING LAB

**LAB 4**

**Submitted by:** Bharat kumar

**Enrollment no:** 12814802719

**Topic covered:** Classes & object Constructor

**LAB 4.**1,4.2,4.3,4.4,4.5

**Aim:**

* WAP that creates a class circle with instance variables for the center and the radius. Initialize and display its variables
* Modify experiment 1 to have a constructor in class circle to initialize its variables
* Modify experiment 2 to show constructor overloading.
* WAP to display the use of this keyword
* Write a program that can count the number of instances created for the class.

**Theory:**

## Class Definition in Java

A Class is a template for an object. Every object has a class which defines the structure of an object (that means what are various component in it, termed as member elements) its functional inter feces (called methods) which decide what messages the object will respond to and how. The general form of class definition is shown below.

Class Class Name

[extends SuperClassName ]

[ implements Interface ] {

[declaration of member elements ]

[ declaration of methods ]}

Creating and Initializing Objects

Take another look at how we've been creating Circle objects:

Circle c = new Circle();

# Constructor overloading in Java

In Java, we can overload constructors like methods. 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.

Consider the following java program, in which we have used different constructors in the class.

## **Instantiating a Class**

The new operator instantiates a class by allocating memory for a new object and returning a reference to that memory. The new operator also invokes the object constructor.

**Note:** The phrase "instantiating a class" means the same thing as "creating an object." When you create an object, you are creating an "instance" of a class, therefore "instantiating" a class.

**Source code:**

public class circle {

private double radius;

private String center;

private static int count=0;

circle(){

radius=-1;

center="undefined";

count++;

}

circle(double radius,String center){

this.radius=radius;

this.center=center;

count++;

}

circle(circle c){

radius=c.radius;

center=c.center;

count++;

}

void display(){

System.out.println("Radius:"+radius+" Dimension of center:"+center);

}

static int getNumOfInstance(){

return count;

}

public static void main(String[] args) {

circle c1=new circle();

circle c2=new circle(10.0,"2,2");

circle c3=new circle(15.5,"0,0");

circle c4=new circle(c2);

c1.display();

c2.display();

c3.display();

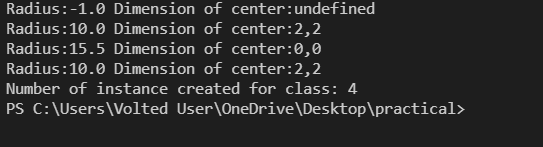
c4.display();

System.out.println("Number of instance created for class: "+getNumOfInstance());

}

}

**Output**



**LAB 4.6**

**Aim:** Java Program to get the cube of a given number using the static method

**Theory:**

## **What Does Static Method Mean?**

In Java, a static method is a method that belongs to a class rather than an instance of a class. The method is accessible to every instance of a class, but methods defined in an instance are only able to be accessed by that object of a class.

A static method is not part of the objects it creates but is part of a class definition. Unlike instance methods, a static method is referenced by the class name and can be invoked without creating an object of class.

In simpler terms, they are methods that exist even if no object has been constructed yet and that do not require an invocation object.

## **Techopedia Explains Static Method**

Java allows developers to define static methods, which are also available to every instance of a class.

In an instance of a class, static methods cannot access variables in an instance and those belonging to a class. They can only access static fields and have to use object reference.

**Source code:**

public class prog46

{

static int num=6;

static void cube() {

System.out.println("Cube of the given number is " + (num \* num \* num));

}

public static void main(String[] args) {

cube();

}

}

**Output:**



**LAB 4.7**

**Aim:** WAP that describes a class person. It should have instance variables to record name, age and salary. Create a person object. Set and display its instance variables.

**Source code:**

class person {

String name;

int age;

long salary;

public person(String name, int age, long salary) {

this.name = name;

this.age = age;

this.salary = salary;

}

public String getName() {

return name;

}

// method 2

public int getAge() {

return age;

}

// method 3

public long getSalary() {

return salary;

}

public String display() {

return ("Name: " + getName() + ".\nAge: " + getAge() + "\nSalary: " + getSalary());

}

public static void main(String[] args) {

person p = new person("Anuj", 61, 4500000);

person p1 = new person("Virat", 29, 6000000);

person p2 = new person("Varun", 40, 4200000);

System.out.println("\n"+p.display()+ "\n");

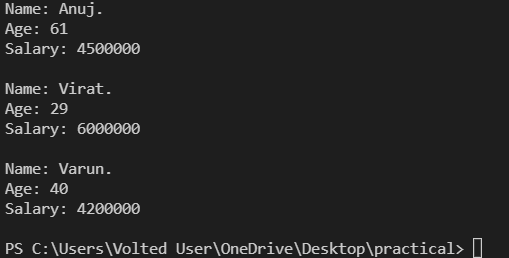
System.out.println(p1.display()+ "\n");

System.out.println(p2.display()+ "\n");

}

}

**Output:**



VIVA VOCE QUESTION

1. What is the purpose of declaring a variable as final?

Ans. A final variable's value can't be changed. final variables should be initialized before using them.

2. What is the impact of declaring a method as final?

Ans. A method declared as final can't be overridden. A sub-class can't have the same method signature with a different implementation.

3. I don't want my class to be inherited by any other class. What should i do?

Ans. You should declared your class as final. But you can't define your class as final, if it is an abstract class. A class declared as final can't be extended by any other class. 31. Can you give few examples of final classes defined in Java API? java.lang.String, java.lang.Math are final classes.

4. How is final different from finally and finalize()?

Ans. final is a modifier which can be applied to a class or a method or a variable. final class can't be inherited, final method can't be overridden and final variable can't be changed. finally is an exception handling code section which gets executed whether an exception is raised or not by the try block code segment. finalize() is a method of Object class which will be executed by the JVM just before garbage collecting object to give a final chance for resource releasing activity.

5. Can a class be declared as static?

Ans. We can not declare top level class as static, but only inner class can be declared static.

public class Test

{ static class InnerClass

{

public static void InnerMethod()

{ System.out.println("Static Inner Class!"); }

} public static void main(String args[])

{ 100 Test.InnerClass.InnerMethod(); }

}

//output: Static Inner Class!