

**St. Francis Institute of Technology**

**Borivli (West), Mumbai-400103**

**Department of Information Technology**

**Experiment – 3**

**Program:-**

1. WAP in java to create Box class with parameterized constructor with an object argument to initialize length, breadth and height also create a function volume which returns the volume of the box and print it in main method.

Program:

```
import java.util.Scanner;

//Class box
class Box{

    //Private data members of class Box
    private float length, breadth, height;

    //Parameterized constructor of class Box
    public Box(float length, float breadth, float height){
        this.length = length;
        this.breadth = breadth;
        this.height = height;
    }

    //Public function to get volume
    public float volume(){
        float volume = length*breadth*height;
        return volume;
    }
}

public class Exp3_1 {

    //Main method
    public static void main(String[] args) {

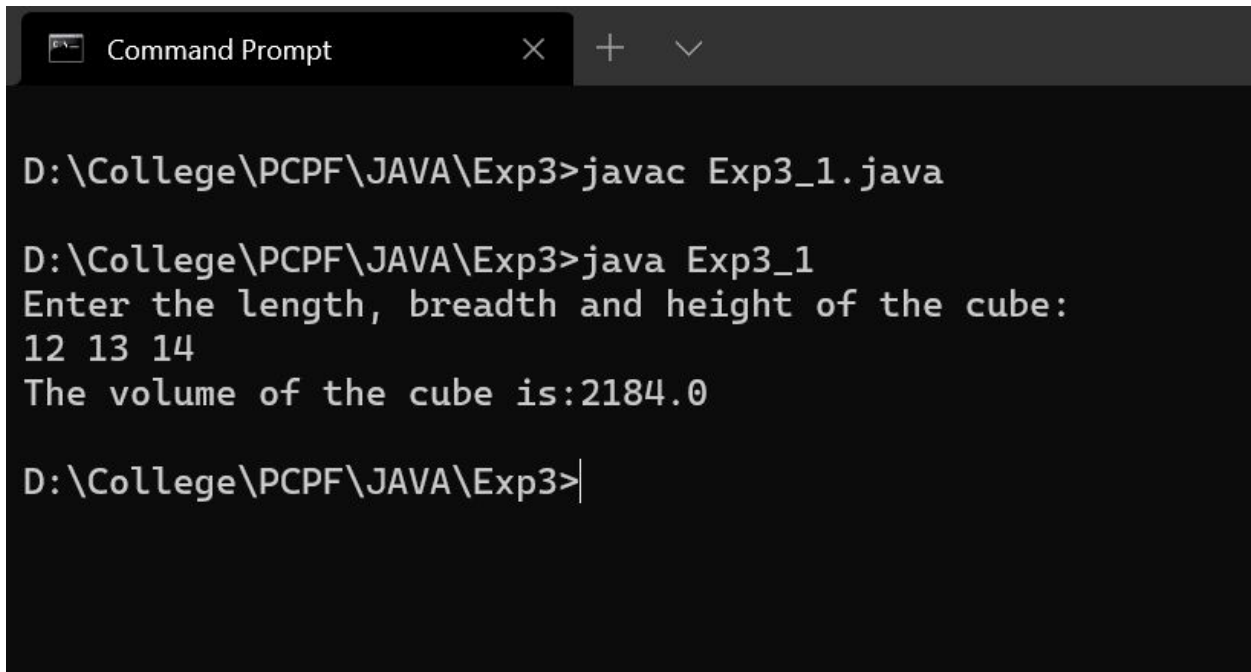
        float length, breadth, height;
```

```
Scanner scanner = new Scanner(System.in);
System.out.println("Enter the length, breadth and height of the
cube:");
length = scanner.nextFloat();
breadth = scanner.nextFloat();
height = scanner.nextFloat();

//Instantiating object of class and passing parameters
Box cube = new Box(length,breadth,height);
System.out.println("The volume of the cube is:"+cube.volume());

scanner.close();
}
}
```

Output:



```
Command Prompt
D:\College\PCPF\JAVA\Exp3>javac Exp3_1.java
D:\College\PCPF\JAVA\Exp3>java Exp3_1
Enter the length, breadth and height of the cube:
12 13 14
The volume of the cube is:2184.0
D:\College\PCPF\JAVA\Exp3>
```

From this output we can infer that we can pass parameters through a parameterized constructor.

2. Write a program to create a class circle. Create two public methods to take value and show value for area and circumference of circle. The radius should be declared as private. Also create a package p1 to store the class circle. Create another package p2 to access the area of circle from package p1.

Program:

```
//Creating package p1
package p1;

//Public declaration of class circle
public class circle {

    //Private data members
    private double radius;
    private double area, circumference;

    //Public Functions
    public double getRadius() {
        return radius;
    }

    public void setRadius(double radius) {
        this.radius = radius;
    }

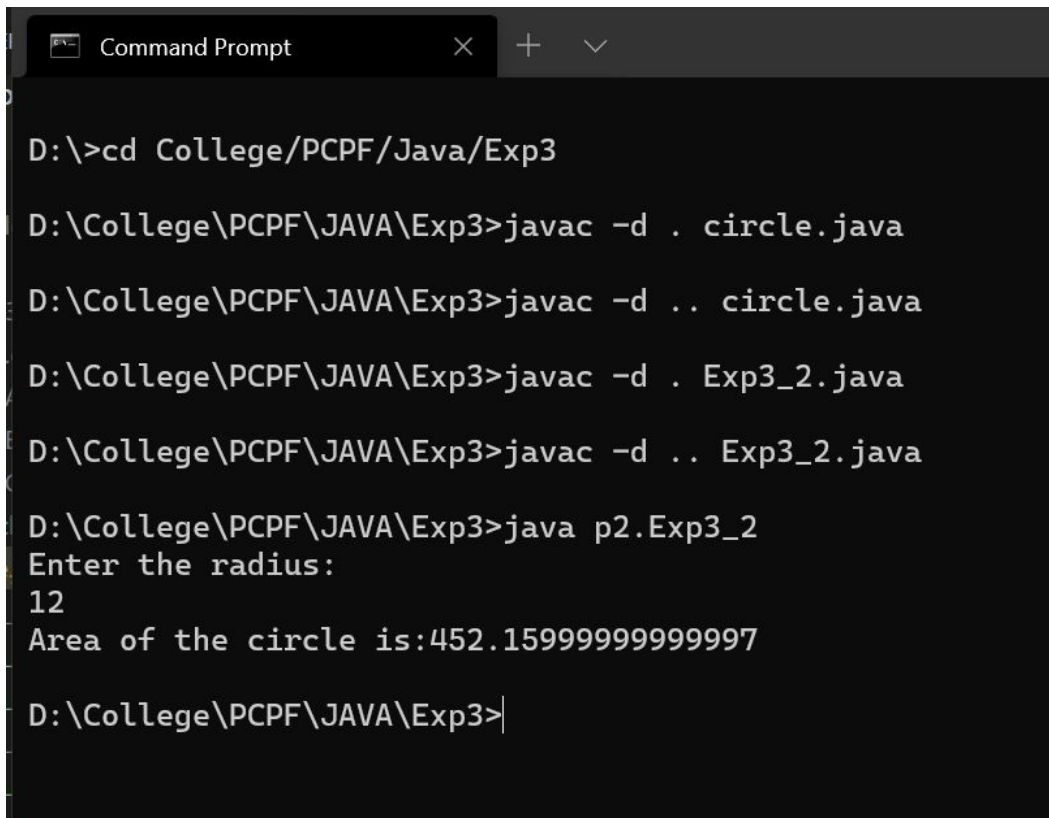
    public double getArea() {
        area = 3.14*radius*radius;
        return area;
    }

    public double getCircumference() {
        circumference = 2*3.14*radius;
        return circumference;
    }
}
```

```
//Creating package p2
package p2;

import java.util.Scanner;
//importing package p1
import p1.*;
public class Exp3_2 {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        double radius;
        System.out.println("Enter the radius:");
        radius=scanner.nextDouble();
        //Instantiating object of class circle
        circle c = new circle();
        c.setRadius(radius);
        System.out.println("Area of the circle is:"+ c.getArea());
        scanner.close();
    }
}
```

Output:



```
Command Prompt
D:\>cd College/PCPF/Java/Exp3
D:\College\PCPF\JAVA\Exp3>javac -d . circle.java
D:\College\PCPF\JAVA\Exp3>javac -d .. circle.java
D:\College\PCPF\JAVA\Exp3>javac -d . Exp3_2.java
D:\College\PCPF\JAVA\Exp3>javac -d .. Exp3_2.java
D:\College\PCPF\JAVA\Exp3>java p2.Exp3_2
Enter the radius:
12
Area of the circle is:452.15999999999997
D:\College\PCPF\JAVA\Exp3>
```

From this program we infer that we can only access the classes of another package if their visibility access is public.

3. Create an abstract class 'Bank' with an abstract method 'getBalance'. \$100, \$150 and \$200 are deposited in banks A, B and C respectively. 'BankA', 'BankB' and 'BankC' are subclasses of class 'Bank', each having a method named 'getBalance'. Call this method by creating an object of each of the three classes.

```
//Abstract class Bank
abstract class Bank{
    //Abstract method get balance
    abstract float getbalance();
}

//Class BankA extending abstract class Bank
class BankA extends Bank{
    private float balance;
    public void setBalance(float balance){
        this.balance=balance;
    }
    //Definition of function getBalance
    public float getbalance(){
        return balance;
    }
}

//Class BankB extending abstract class Bank
class BankB extends Bank{
    private float balance;

    public void setBalance(float balance){
        this.balance=balance;
    }
    //Definition of function getBalance
    public float getbalance(){
        return balance;
    }
}

//Class BankC extending abstract class Bank
class BankC extends Bank{
    private float balance;
    public void setBalance(float balance){
        this.balance=balance;
    }
}
```

```
//Definition of function getBalance
public float getbalance() {
    return balance;
}
}
public class Exp3_3 {

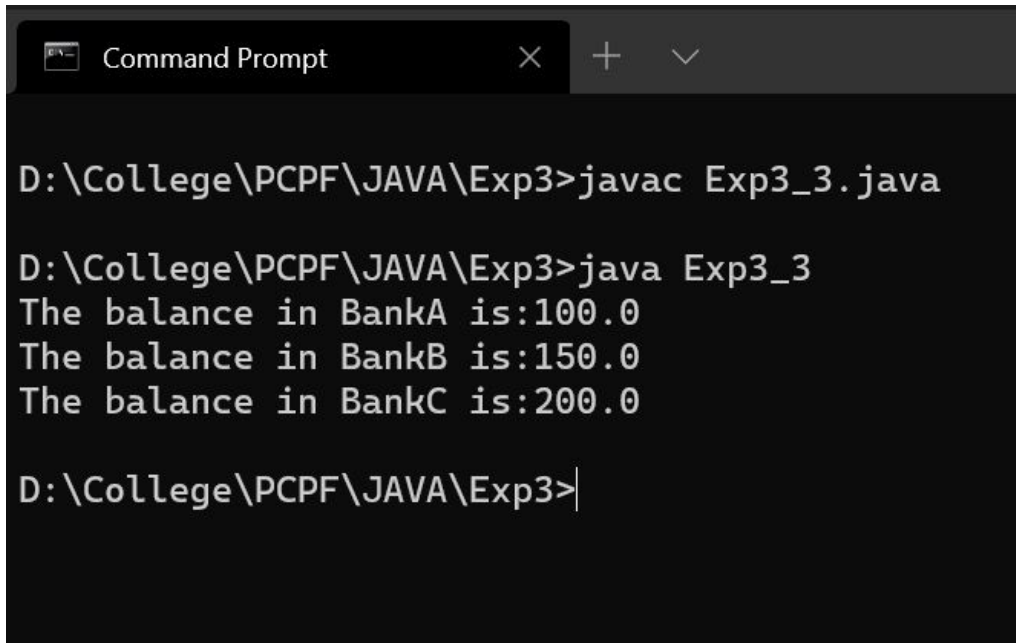
    //main method
    public static void main(String[] args) {

        //Object of BankA
        BankA a = new BankA();
        a.setBalance(100);
        System.out.println("The balance in BankA is:"+a.getbalance());

        //Object of BankB
        BankB b = new BankB();
        b.setBalance(150);
        System.out.println("The balance in BankB is:"+b.getbalance());

        //Object of BankC
        BankC c = new BankC();
        c.setBalance(200);
        System.out.println("The balance in BankC is:"+c.getbalance());
    }
}
```

Output:



```
D:\College\PCPF\JAVA\Exp3>javac Exp3_3.java

D:\College\PCPF\JAVA\Exp3>java Exp3_3
The balance in BankA is:100.0
The balance in BankB is:150.0
The balance in BankC is:200.0

D:\College\PCPF\JAVA\Exp3>|
```

From this program we infer that we can use abstract classes for data abstraction. Object Abstract classes cannot be instantiated.

### Post Experimental Exercise-

#### Questions:

#### 1. What is the significance encapsulation with respect to data visibility?

Data Encapsulation is an Object Oriented Programming concept that binds a group of related properties, functions, and other members are treated as a single unit. It is sometimes referred to as data hiding that prevents the user to access the implementation details. Encapsulation therefore guarantees the integrity of the data contained in the Object. Data Encapsulation is implemented by using access specifiers (Access Modifiers) and it defines the scope and visibility of a class member. The most common type of access specifiers are public, private and protected.

#### 2. What is an interface? When is an interface used in imperative programming style?

An interface is a programming structure/syntax that allows the computer to enforce certain properties on an object (class). It is similar to an abstract class. Interfaces allow us to define common behavior that can be implemented by any class, regardless of its inheritance. For example, say we have a car class and a scooter class and a truck class. Each of these three classes should have a start\_engine() action. How the "engine is started" for each vehicle is left to each particular class, but the fact that they must have a start\_engine action is the domain of the interface.

Methods that take an interface as an argument type are going to be more future-proof than methods that take a class as an argument type. The reason is that it doesn't tie your method to a particular inheritance tree thus making the program more flexible.

Polymorphism that's free from inheritance is free from the restrictions that inheritance imposes. Most notably, languages like Java and C# do not allow you to inherit from more than one class. But since interfaces are not tied to inheritance, a class can implement any number of interfaces.

**3. WAP to calculate the area of a rectangle, a square and a circle. Create an abstract class 'Shape' with three abstract methods namely 'RectangleArea' taking two parameters, 'SquareArea' and 'CircleArea' taking one parameter each. The parameters of 'RectangleArea' are its length and breadth, that of 'SquareArea' is its side and that of 'CircleArea' is its radius. Now create another class 'Area' containing all the three methods 'RectangleArea', 'SquareArea' and 'CircleArea' for printing the area of rectangle, square and circle respectively. Create an object of class 'Area' and call all the three methods.**

```
//Declaration of abstract class shape
abstract class Shape{
    //Abstract methods
    abstract float RectangleArea(float length, float breadth);
    abstract float SquareArea(float side);
    abstract double CircleArea(double radius);
}

//Class area extends abstract class Shape
class Area extends Shape{

    //Function definition of abstract method RectangleArea
    float RectangleArea(float length, float breadth){

        return length*breadth;
    }
    //Function definition of abstract method SquareArea
    float SquareArea(float side){

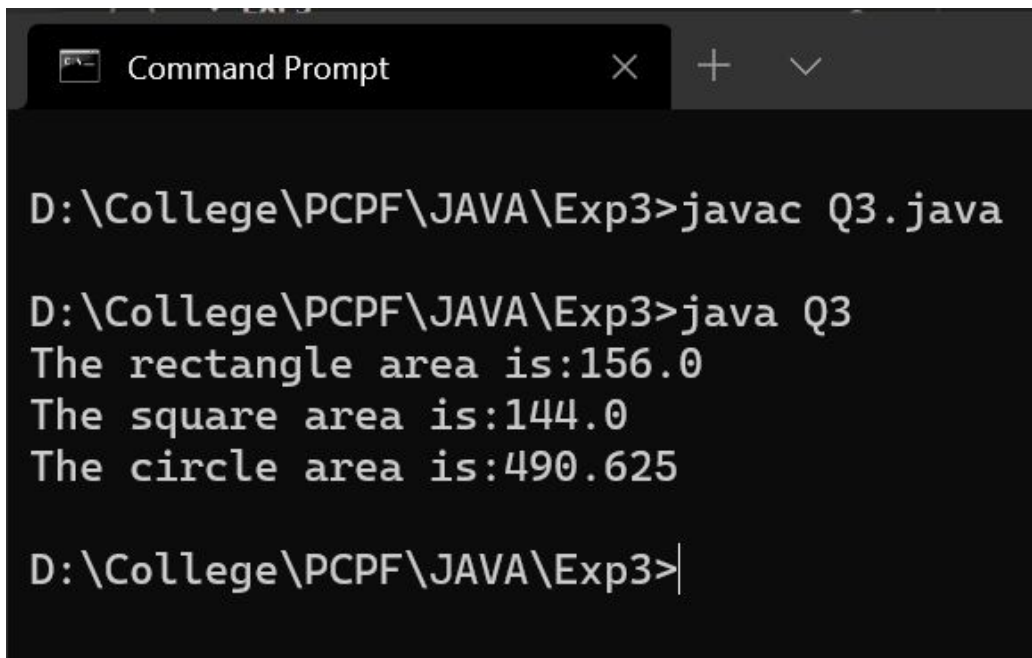
        return side*side;
    }
    //Function definition of abstract method CircleArea
    double CircleArea(double radius){

        return 3.14*radius*radius;
    }
}

public class Q3 {
```



```
public static void main(String[] args) {  
  
    //Object of class Area  
    Area a = new Area();  
    System.out.println("The rectangle area is:" + a.RectangleArea(12,  
13));  
    System.out.println("The square area is:" + a.SquareArea(12));  
    System.out.println("The circle area is:" + a.CircleArea(12.5));  
}  
}
```



```
D:\College\PCPF\JAVA\Exp3>javac Q3.java  
  
D:\College\PCPF\JAVA\Exp3>java Q3  
The rectangle area is:156.0  
The square area is:144.0  
The circle area is:490.625  
  
D:\College\PCPF\JAVA\Exp3>|
```

**4. Create an interface Animal, with two methods called animal Sound() and animalHome(). Implement the interface to define their “Home” and “Cries”. Access the interface methods from the main class.**

```
//Declaring interface animal  
interface Animal {  
    //members of interface are public and abstract by default  
    void Sound();  
    void animalHome();  
}  
  
//class cow implements interface animal
```

```
class Cow implements Animal{

    //Definition of function sound
    public void Sound(){
        System.out.println("The cow says moo");
    }
    //Definition of function animalHome
    public void animalHome(){
        System.out.println("The cow lives in a barn");
    }
}

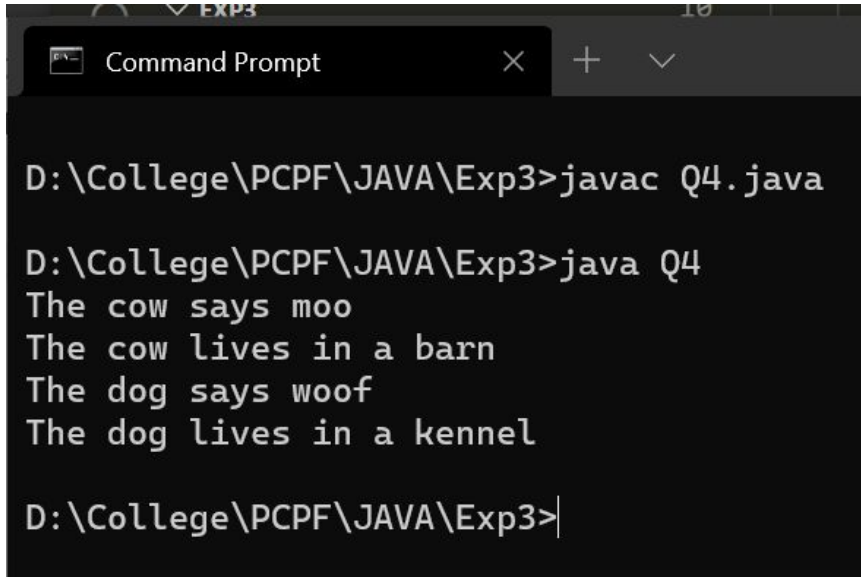
//class dog implements interface animal
class Dog implements Animal{

    //Definition of function sound
    public void Sound(){
        System.out.println("The dog says woof");
    }
    //Definition of function animalHome
    public void animalHome(){
        System.out.println("The dog lives in a kennel");
    }
}

public class Q4{

    public static void main(String[] args) {

        //object of class cow
        Cow c = new Cow();
        c.Sound();
        c.animalHome();
        //object of class dog
        Dog d = new Dog();
        d.Sound();
        d.animalHome();
    }
}
```



```
D:\College\PCPF\JAVA\Exp3>javac Q4.java

D:\College\PCPF\JAVA\Exp3>java Q4
The cow says moo
The cow lives in a barn
The dog says woof
The dog lives in a kennel

D:\College\PCPF\JAVA\Exp3>|
```

### Conclusion:

Object Oriented programming (OOP) is a programming paradigm that relies on the concept of classes and objects. In this experiment we have written programs in Java to implement concepts of Data Abstraction and Encapsulation.

To perform this experiment Visual Studio Code was used as a code editing software and Java 15 language was in all the codes.

From this experiment we can infer that Encapsulation is a process of wrapping the data and the code that operates on the data into a single entity. You can assume it as a protective wrapper that stops random access of code defined outside that wrapper. Abstraction is a concept that focuses only on relevant data of an object. It hides the background details and emphasizes the essential data points for reducing the complexity and increasing efficiency.

### References:

- [1] E Balaguruswamy, "Object Oriented Programming with C++", second edition Tata McGraw Hill
- [2] E Balaguruswamy, " Programming with Java-A Primer", third edition, Tata McGraw Hill