**Lab Report**

**Course Tittle:** Object Oriented Programming Lab

**Course Code:** CSE 215

**Experiment No:** 07

**Experiment Name:** Implementation of Abstraction

**Submitted To:**

**Name:** Mst. Umme Ayman

**Designation:** Lecturer

**Department of CSE**

**Daffodil International University**

**Submitted By**

**Name:** Md Raduan Ahamed

**ID:** 0242220005101839

**Section:** 63\_O

**Department of CSE**

**Daffodil International University**

**Submission Date:** 14-11-2023

**Problem: 01**

You can create an abstract Shape class with methods like area() and perimeter(). Then, you can create concrete subclasses like Circle and Rectangle that provide specific implementations of these methods.

abstract class shape {

*//use abstract class*

public abstract double area();  
 public abstract double perimeter();  
}

class circle extends shape {

private double r ;   
 circle(double r){ *//constructor* this.r=r;

}  
 public double area(){

return 3.1416\*r\*r;  
 }  
 public double perimeter(){

return 2\*3.1416\*r;  
 }  
}

class rectangle extends shape {

private double h,l ;  
 rectangle(double h,double l){ *//constractor* this.h=h;  
 this.l=l;  
 }

public double area(){  
 return h\*l;  
 }  
 public double perimeter(){  
 return 2\*(h+l);  
 }  
}

public class lab\_r {

public static void main(String[] args) {

circle c= new circle(9); *//create object for circle* System.*out*.println("Area of Circle :"+c.area());  
 System.*out*.println("Perimeter of Circle :"+c.perimeter());  
  
 rectangle re =new rectangle(6, 8); *//create object for rectangle* System.*out*.println("Area of Rectangle :"+re.area());  
 System.*out*.println("Perimeter of Rectangle : "+re.perimeter());  
 }  
}

**Output:**

Area of Circle : 153.9384

Perimeter of Circle : 56.5488

Area of Rectangle : 48.0

Perimeter of Rectangle : 28.0

**Problem: 02**

In a banking system, you can use abstraction to model bank accounts. Create an abstract class BankAccount with methods like deposit(), withdraw(), and getBalance(). Then, implement concrete classes for different types of accounts like SavingsAccount and CheckingAccount.

package b;

abstract class bankaccount { //create abstract class

public abstract void setBalance(double balance);

public abstract void deposit(double amount); //create abstract method

public abstract void withdraw(double amount);

public abstract double getBalance();

}

class saving\_account extends bankaccount {

private double balance;

@Override

public void setBalance(double balance) { //use setter

this.balance = balance;

}

@Override

public void deposit(double amount) {

balance += amount;

}

@Override

public void withdraw(double amount) {

balance -= amount;

}

@Override

public double getBalance() {

return balance;

}

}

class checking\_account extends bankaccount {

private double balance;

@Override

public void setBalance(double balance) { //use setter

this.balance = balance;

}

@Override

public void deposit(double amount) {

balance += amount;

}

@Override

public void withdraw(double amount) {

balance -= amount;

}

@Override

public double getBalance() {

return balance;

}

}

public class bank {

public static void main(String[] args) {

saving\_account save = new saving\_account(); // create object

save.setBalance(10000);

save.deposit(500);

save.withdraw(1000);

System.out.println("Current Savings Account Balance: " +

save.getBalance());

checking\_account check = new checking\_account(); // create object

check.setBalance(15000);

check.deposit(1000);

check.withdraw(500);

System.out.println("Current Checking Account Balance: " +

check.getBalance());

}

}

**Output:**

Current Savings Account Balance: 9500.0

Current Checking Account Balance: 16000.0