**Interface programe**

**6.You are give two classes Person and Student, where Person is the base class and student is the derived class complete code for Person and a declaration for Student are provide for you is the editor observe that Student inherits all the properties of person comlete the student class by writing the following a student class constructer, which has paramenter.**

**Astring;**

**astring;**

**an integer;**

**in integer array(or vector) of test scores**

**a char calculater() method that calculating a student object average and return the grade charecter representative of their calculate average.**

**Grading scale**

**letter Average**

**O 90<=a<=100**

**E 80<=a<90**

**P 70<=a<=80**

**A 55<=a<70**

**D 40<=a<55**

**T a<40**

PROGRAME:

public class Person {

// Person class code here

}

public class Student extends Person {

private String name;

private String id;

private int age;

private int[] testScores;

public Student(String name, String id, int age, int[] testScores) {

this.name = name;

this.id = id;

this.age = age;

this.testScores = testScores;

}

public char calculateGrade() {

int average = calculateAverage();

if (average >= 90 && average <= 100) {

return 'O';

} else if (average >= 80 && average < 90) {

return 'E';

} else if (average >= 70 && average <= 80) {

return 'P';

} else if (average >= 55 && average < 70) {

return 'A';

} else if (average >= 40 && average < 55) {

return 'D';

} else {

return 'T';

}

}

private int calculateAverage() {

int sum = 0;

for (int score : testScores) {

sum += score;

}

return sum / testScores.length;

}

}

1. **WRITE A JAVA PRIOGRAME TO CREAT AN INTERFACE SHAPE WITH THE getArea() method. Create three classes Rectangle, circle and triangle that implement the Shape interface. Imlent the getArea() method for each of three classes.**

PROGRAME:

interface Shape {

double getArea();

}

class Rectangle implements Shape {

private double length;

private double width;

public Rectangle(double length, double width) {

this.length = length;

this.width = width;

}

public double getArea() {

return length \* width;

}

}

class Circle implements Shape {

private double radius;

public Circle(double radius) {

this.radius = radius;

}

public double getArea() {

return Math.PI \* radius \* radius;

}

}

class Triangle implements Shape {

private double base;

private double height;

public Triangle(double base, double height) {

this.base = base;

this.height = height;

}

public double getArea() {

return 0.5 \* base \* height;

}

}

public class Main {

public static void main(String[] args) {

Rectangle rectangle = new Rectangle(5, 3);

Circle circle = new Circle(4);

Triangle triangle = new Triangle(6, 2);

System.out.println("Area of Rectangle: " + rectangle.getArea());

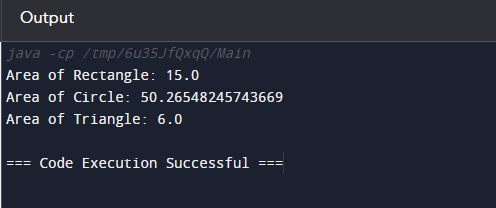
System.out.println("Area of Circle: " + circle.getArea());

System.out.println("Area of Triangle: " + triangle.getArea());

}

}

**OUTPUT:**



**8.write a java programe to creat a Animal inteface with a method called bark() that takes o argument and return void. Creat a dog class that implement Animal and override Speak() to print "Dog is barking".**

PROGRAME:

interface Animal {

void bark();

}

class Dog implements Animal {

@Override

public void bark() {

System.out.println("Dog is barking");

}

}

public class Main {

public static void main(String[] args) {

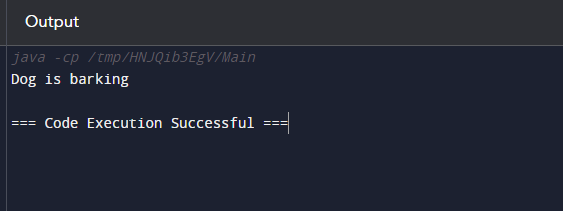
Dog dog = new Dog();

dog.bark();

}

}

**OUTPUT:**



**9.Write a java programming to create a banking system with three class Bankaccount, Savingaccount and currentaccount. The bank should have a list of account and method for adding them. Accounts should be an interface with method to deposit, withdraw, calculate interest and view balance. savingaccount and currentaccount should implement the account interface and have their own unique method.**

Programe:

import java.util.ArrayList;

import java.util.List;

interface Account {

void deposit(double amount);

void withdraw(double amount);

double calculateInterest();

double viewBalance();

}

class BankAccount {

List<Account> accounts = new ArrayList<>();

public void addAccount(Account account) {

accounts.add(account);

}

}

class SavingsAccount implements Account {

private double balance;

public void deposit(double amount) {

balance += amount;

}

public void withdraw(double amount) {

balance -= amount;

}

public double calculateInterest() {

return balance \* 0.05; // 5% interest rate

}

public double viewBalance() {

return balance;

}

// Unique method for Savings Account

public void applyBonus() {

balance += 100; // Bonus amount

}

}

class CurrentAccount implements Account {

private double balance;

public void deposit(double amount) {

balance += amount;

}

public void withdraw(double amount) {

balance -= amount;

}

public double calculateInterest() {

return 0; // Current account does not earn interest

}

public double viewBalance() {

return balance;

}

// Unique method for Current Account

public void issueOverdraft() {

balance -= 50; // Overdraft fee

}

}

public class Main {

public static void main(String[] args) {

BankAccount bank = new BankAccount();

SavingsAccount savingsAcc = new SavingsAccount();

CurrentAccount currentAcc = new CurrentAccount();

bank.addAccount(savingsAcc);

bank.addAccount(currentAcc);

savingsAcc.deposit(500);

currentAcc.deposit(1000);

System.out.println("Savings Account Balance: $" + savingsAcc.viewBalance());

System.out.println("Current Account Balance: $" + currentAcc.viewBalance());

System.out.println("Interest on Savings Account: $" + savingsAcc.calculateInterest());

}

}

**OUTPUT:**

