TASK1:

/\*

\* To change this license header, choose License Headers in Project Properties.

\* To change this template file, choose Tools | Templates

\* and open the template in the editor.

\*/

package javaapplication2;

import java.util.Scanner;

/\*\*

\*

\* @author test23

\*/

public class JavaApplication2 {

public static void main(String[] args) {

Rectangle r = new Rectangle();

System.out.println("area of RECTANGLE :"+r.getarea());

Circle c = new Circle();

System.out.println("area of CIRCLE :"+c.getarea());

Triangle t = new Triangle();

System.out.println("area of triangle :"+t.getarea());

}

}

interface shape{

double getarea();

}

class Rectangle implements shape{

public double getarea(){

System.out.println("Enter Height and Base for rectangle:");

Scanner sc = new Scanner(System.in);

int h = sc.nextInt();

int b = sc.nextInt();

return (b \*h);

}

}

class Circle implements shape{

public double getarea(){

System.out.println("Enter Radius:");

Scanner sc = new Scanner(System.in);

int r = sc.nextInt();

return (3.14 \* r \*r);

}

}

class Triangle implements shape{

public double getarea(){

System.out.println("Enter Height and Base:of Triangle");

Scanner sc = new Scanner(System.in);

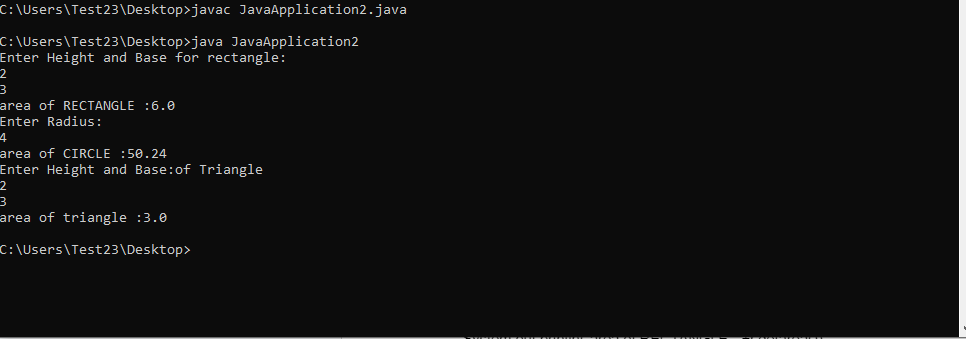
int h = sc.nextInt();

int b = sc.nextInt();

return 0.5\*b \*h;

}

}



Task2:

public class Question2 {

public static void main(String[] args) {

Dog d = new Dog();

d.bark();

}

}

interface Animal {

void bark();

}

class Dog implements Animal {

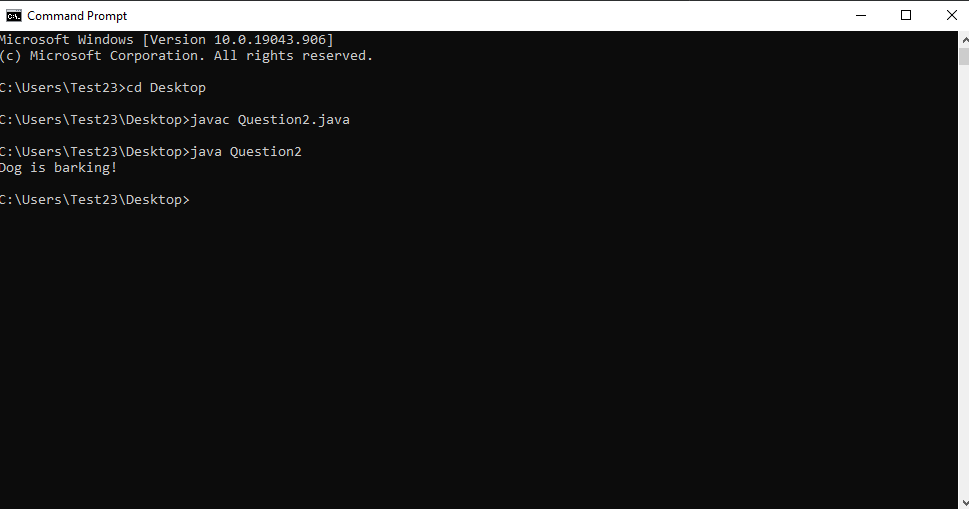
@Override

public void bark() {

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

}

}



Task3:

import java.util.ArrayList;

public class Task3 {

public static void main(String[] args) {

Bank bank = new Bank();

SavingsAccount s = new SavingsAccount(212010, 0.2);

CurrentAccount c = new CurrentAccount(350, 1500);

bank.addAccount(s);

bank.addAccount(c);

s.deposit(200);

c.withdraw(100);

System.out.println("SavingsAccount: " + s.getBalance());

System.out.println("CurrentAccount: " + c.getBalance());

}

}

interface Account {

void deposit(double amount);

void withdraw(double amount);

double calculateInterest();

double getBalance();

}

class SavingsAccount implements Account {

private double balance;

private double interestRate;

public SavingsAccount(double initBalance, double interestRate) {

this.balance = initBalance;

this.interestRate = interestRate;

}

@Override

public void deposit(double amount) {

balance = balance + amount;

}

@Override

public void withdraw(double amount) {

if (amount <= balance) {

balance = balance - amount;

} else {

System.out.println("Insufficient balance.");

}

}

@Override

public double calculateInterest() {

return balance \* interestRate;

}

@Override

public double getBalance() {

return balance;

}

}

class CurrentAccount implements Account {

private double balance;

private double overdraftLimit;

public CurrentAccount(double initBalance, double overdraftLimit) {

this.balance = initBalance;

this.overdraftLimit = overdraftLimit;

}

@Override

public void deposit(double amount) {

balance =balance + amount;

}

@Override

public void withdraw(double amount) {

if (amount <= balance + overdraftLimit) {

balance = balance - amount;

} else {

System.out.println("Withdrawal exceeds overdraft limit.");

}

}

@Override

public double calculateInterest() {

// As Current accounts don't have interest

return 0;

}

@Override

public double getBalance() {

return balance;

}

}

class Bank {

private ArrayList<Account> accounts;

public Bank() {

accounts = new ArrayList<>();

}

public void addAccount(Account account) {

accounts.add(account);

}

}



Task4:

public class Task4 {

public static void main(String[] args) {

Drawable c = new Circle();

Drawable r = new Rectangle();

Drawable t = new Triangle();

c.draw();

System.out.println();

r.draw();

System.out.println();

t.draw();

}

}

interface Drawable {

void draw();

}

class Circle implements Drawable {

@Override

public void draw() {

System.out.println("circle");

int r = 5;

int X = r;

int Y = r;

for (int i = 0; i <= 2 \* r; i++) {

for (int j = 0; j <= 2 \* r; j++) {

double distance = Math.sqrt((i - r) \* (i - r) + (j - r) \* (j - r));

if (distance > r - 0.5 && distance < r + 0.5) {

System.out.print("\* ");

} else {

System.out.print(" ");

}

}

System.out.println();

}

}

}

class Rectangle implements Drawable {

@Override

public void draw() {

System.out.println("rectangle ");

for (int i = 0; i < 5; i++) {

for (int j = 0; j < 10; j++) {

System.out.print("\* ");

}

System.out.println();

}

}

}

class Triangle implements Drawable {

@Override

public void draw() {

System.out.println("triangle");

int height = 5;

for (int i = 0; i < height; i++) {

for (int j = 0; j < height - i - 1; j++) {

System.out.print(" ");

}

for (int j = 0; j < 2 \* i + 1; j++) {

System.out.print("\*");

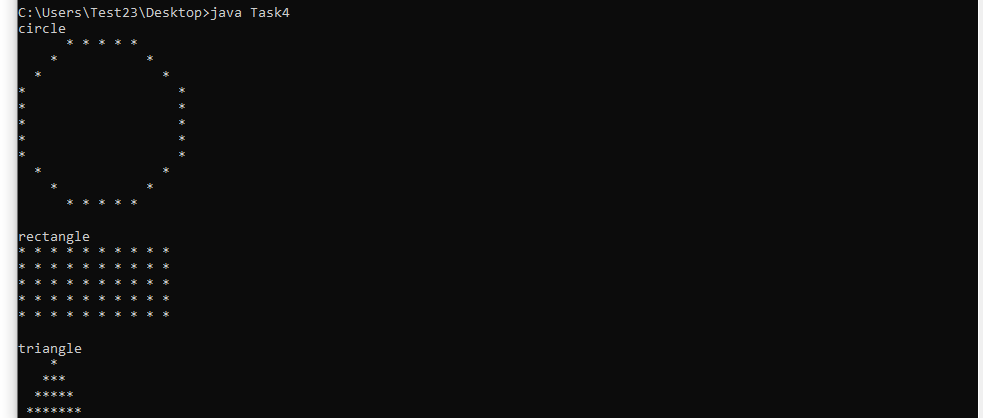
}

System.out.println();

}

}

}



Task5:

public class Task5 {

public static void main(String[] args) {

Circle c = new Circle(3);

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

System.out.println("Circle Perimeter: " + c.getPerimeter());

Rectangle r = new Rectangle(5, 7);

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

System.out.println("Rectangle Perimeter: " + r.getPerimeter());

Triangle t = new Triangle(2, 7, 3, 5, 3);

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

System.out.println("Triangle Perimeter: " + t.getPerimeter());

}

}

abstract class Shape {

abstract double getArea();

abstract double getPerimeter();

}

class Circle extends Shape {

private double radius;

public Circle(double radius) {

this.radius = radius;

}

@Override

double getArea() {

return 3.14 \* radius \* radius;

}

@Override

double getPerimeter() {

return 2 \* 3.14 \* radius;

}

}

class Rectangle extends Shape {

private double length;

private double width;

public Rectangle(double length, double width) {

this.length = length;

this.width = width;

}

@Override

double getArea() {

return length \* width;

}

@Override

double getPerimeter() {

return 2 \* (length + width);

}

}

class Triangle extends Shape {

private double side1;

private double side2;

private double side3;

private double length;

private double width;

public Triangle(double side1, double side2, double side3, double length, double width) {

this.side1 = side1;

this.side2 = side2;

this.side3 = side3;

this.length = length;

this.width = width;

}

@Override

double getArea() {

return (length\*width) / 2;

}

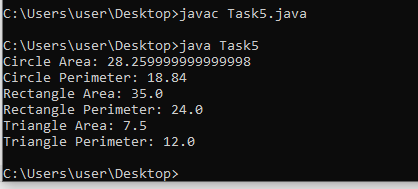
@Override

double getPerimeter() {

return side1 + side2 + side3;

}

}



Task6:

public class Task6 {

public static void main(String[] args) {

Animal a = new Animal();

System.out.println("Base Animal class :");

a.move();

a.makeSound();

Bird b = new Bird();

System.out.println("Bird :");

b.move();

b.makeSound();

Panthera p = new Panthera();

System.out.println("Panthera:");

p.move();

p.makeSound();

}

}

class Animal {

void move() {

System.out.println("The animal moves.");

}

void makeSound() {

System.out.println("The animal makes sound.");

}

}

class Bird extends Animal {

@Override

void move() {

System.out.println("The bird flies.");

}

@Override

void makeSound() {

System.out.println("The bird sings.Churrp churrp");

}

}

class Panthera extends Animal {

@Override

void move() {

System.out.println("The panthera runs.");

}

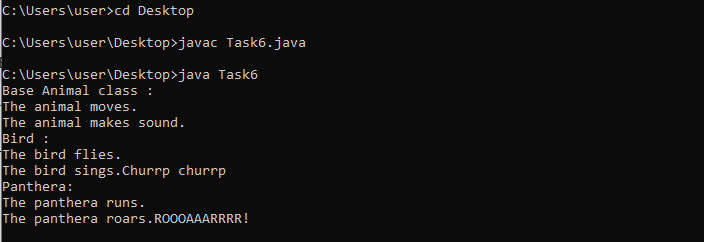
@Override

void makeSound() {

System.out.println("The panthera roars.ROOOAAARRRR!");

}

}



Task7:

public class Task7 {

public static void main(String[] args) {

Animal a = new Animal();

System.out.println("Base animal class:");

a.eat();

a.sound();

Lion l= new Lion();

System.out.println("Lion:");

l.eat();

l.sound();

Tiger t = new Tiger();

System.out.println("Tiger:");

t.eat();

t.sound();

Panther p = new Panther();

System.out.println("Panther:");

p.eat();

p.sound();

}

}

class Animal {

void eat() {

System.out.println("The animal eats.");

}

void sound() {

System.out.println("The animal makes sound.");

}

}

class Lion extends Animal {

@Override

void eat() {

System.out.println("The lion eats meat.");

}

@Override

void sound() {

System.out.println("The lion roars.ROAAAAR!");

}

}

class Tiger extends Animal {

@Override

void eat() {

System.out.println("The tiger eats meat.");

}

@Override

void sound() {

System.out.println("The tiger growls.RRRRRRRRRRR!");

}

}

class Panther extends Animal {

@Override

void eat() {

System.out.println("The panther eats meat and small animals.");

}

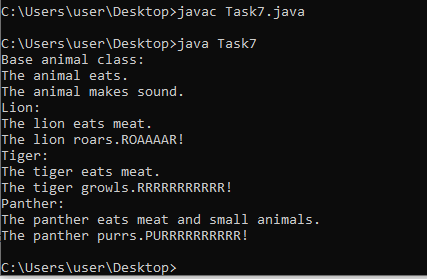
@Override

void sound() {

System.out.println("The panther purrs.PURRRRRRRRRR!");

}

}



Task8:

public class Task8 {

public static void main(String[] args) {

Manager m = new Manager("Haroon Mirza", 3821, 250000.99, 1000.0);

m.displayInfo();

Programmer p = new Programmer("Ali", 3811, 10.00, 250);

p.displayInfo();

}

}

abstract class Employee {

private String name;

private int ID;

public Employee(String name, int ID) {

this.name = name;

this.ID = ID;

}

public String getName() {

return name;

}

public int getID() {

return ID;

}

public abstract double calculateSalary();

public abstract void displayInfo();

}

class Manager extends Employee {

private double baseSalary;

private double bonus;

public Manager(String name, int ID, double baseSalary, double bonus) {

super(name, ID);

this.baseSalary = baseSalary;

this.bonus = bonus;

}

@Override

public double calculateSalary() {

return baseSalary + bonus;

}

@Override

public void displayInfo() {

System.out.println("Manager Info:");

System.out.println("Name: " + getName());

System.out.println("Employee ID: " + getID());

System.out.println("Base Salary: $ " + baseSalary);

System.out.println("Bonus:$" + bonus);

System.out.println("Total Salary:$ " + calculateSalary());

}

}

class Programmer extends Employee {

private double hourlyRate;

private int hoursWorked;

public Programmer(String name, int ID, double hourlyRate, int hoursWorked) {

super(name, ID);

this.hourlyRate = hourlyRate;

this.hoursWorked = hoursWorked;

}

@Override

public double calculateSalary() {

return hourlyRate \* hoursWorked;

}

@Override

public void displayInfo() {

System.out.println("Programmer Info:");

System.out.println("Name: " + getName());

System.out.println("Employee ID: " + getID());

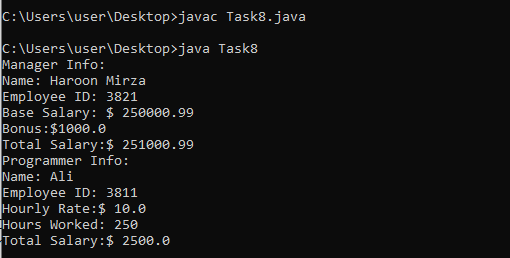
System.out.println("Hourly Rate:$ " + hourlyRate);

System.out.println("Hours Worked: " + hoursWorked);

System.out.println("Total Salary:$ " + calculateSalary());

}

}



Task9:

public class Task9 {

public static void main(String[] args) {

Triangle triangle = new Triangle(3.0, 4.0, 5.0 , 2.0, 4.0);

System.out.println("Triangle:");

System.out.println("Area: " + triangle.area());

System.out.println("Perimeter: " + triangle.perimeter());

Square square = new Square(5.0);

System.out.println("Square:");

System.out.println("Area: " + square.area());

System.out.println("Perimeter: " + square.perimeter());

}

}

abstract class GeometricShape {

public abstract double area();

public abstract double perimeter();

}

class Triangle extends GeometricShape {

private double side1;

private double side2;

private double side3;

private double length;

private double width;

public Triangle(double side1, double side2, double side3, double length , double width) {

this.side1 = side1;

this.side2 = side2;

this.side3 = side3;

this.length = length;

this.width = width;

}

@Override

public double area() {

return (length\*width) / 2;

}

@Override

public double perimeter() {

return side1 + side2 + side3;

}

}

class Square extends GeometricShape {

private double side;

public Square(double side) {

this.side = side;

}

@Override

public double area() {

return side \* side;

}

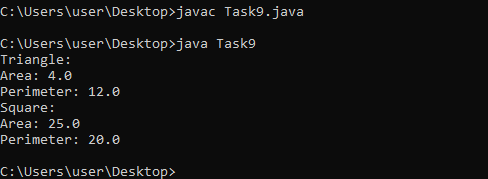
@Override

public double perimeter() {

return 4 \* side;

}

}



Task10:

public class Task10 {

public static void main(String[] args) {

Eagle eagle = new Eagle();

System.out.println("Eagle:");

eagle.fly();

eagle.makeSound();

Hawk hawk = new Hawk();

System.out.println("\nHawk:");

hawk.fly();

hawk.makeSound();

}

}

abstract class Bird {

public abstract void fly();

public abstract void makeSound();

}

class Eagle extends Bird {

@Override

public void fly() {

System.out.println("The eagle Flies above the sky .");

}

@Override

public void makeSound() {

System.out.println("The eagle makes screeching sound.");

}

}

class Hawk extends Bird {

@Override

public void fly() {

System.out.println("The hawk maintains an altitude which is perfect to capture preys.");

}

@Override

public void makeSound() {

System.out.println("The hawk makes a hoarse sound.");

}

}

