**Task 01**

**CODE**

public class TimesMedico{

    static double overallRevenue=2500.99;

    public static void main(String[] args) {

        System.out.println("\n--------Shah Hunain(23K-3032)--------\n");

        Tablet[] t = new Tablet[2];

        t[0]=new Tablet("Panadol", "abcd", 100, "23-3-24", "29-7-24", 0.449);

        t[1]=new Tablet("Disprin", "abcdef", 100, "23-3-24", "29-7-24", 0.379);

        Capsule[] c = new Capsule[2];

        c[0]=new Capsule("Evion", "xyz", 45.99, "1-5-24", "1-8-24", 59.99);

        c[1]=new Capsule("Callo", "vwxyz", 39.00, "10-2-24", "11-8-24", 59.99);

        Syrup[] s = new Syrup[2];

        s[0]=new Syrup("Calpol", "abxy", 250.0,"3-3-24", "5-6-24");

        s[1]=new Syrup("Actified-P", "defghi", 300.99,"9-9-23", "10-7-24");

        Counter counter = new Counter();

        boolean counterFound = counter.searchMedicine(t, c, s);

        if(!counterFound){

            Pharmacist p = new Pharmacist();

            p.searchMedicine(t, c, s);

        }

    }

}

public class Medicine {

    private String name;

    private String formula;

    private double retailPrice;

    private String manufactureDate;

    private String expirationDate;

    Medicine(){

        this.name="null";

        this.formula="null";

        this.retailPrice=0.0;

        this.manufactureDate="null";

        this.expirationDate="null";

    }

    Medicine(String name, String formula, double retailPrice, String manufactureDate, String expirationDate){

        this.name=name;

        this.formula=formula;

        this.retailPrice=retailPrice;

        this.manufactureDate=manufactureDate;

        this.expirationDate=expirationDate;

    }

    public String getName() {

        return name;

    }

    public void setName(String name) {

        this.name = name;

    }

    public String getFormula() {

        return formula;

    }

    public void setFormula(String formula) {

        this.formula = formula;

    }

    public double getRetailPrice() {

        return retailPrice;

    }

    public void setRetailPrice(double retailPrice) {

        this.retailPrice = retailPrice;

    }

    public String getManufactureDate() {

        return manufactureDate;

    }

    public void setManufactureDate(String manufactureDate) {

        this.manufactureDate = manufactureDate;

    }

    public String getExpirationDate() {

        return expirationDate;

    }

    public void setExpirationDate(String expirationDate) {

        this.expirationDate = expirationDate;

    }

    public void printMedicineDetails(){

        System.out.println("\t"+getClass()+" details");

        System.out.println("Name: "+this.name);

        System.out.println("Formula: "+this.formula);

        System.out.println("Retail Price: "+this.retailPrice);

        System.out.println("Manufacture Date: "+this.manufactureDate);

        System.out.println("Expiration Date: "+this.expirationDate);

    }

}

public class Tablet extends Medicine{

    double sucroseLevel;

    Tablet(String name, String formula, double retailPrice, String manufactureDate, String expirationDate, double sucroseLevel){

        super(name, formula, retailPrice, manufactureDate, expirationDate);

        if(sucroseLevel>=0.0 && sucroseLevel<=1.0){

            this.sucroseLevel=sucroseLevel;

        }

    }

    public void setSucroseLevel(double sucroseLevel){

        if(sucroseLevel>=0.0 && sucroseLevel<=1.0){

            this.sucroseLevel=sucroseLevel;

        }

    }

    public double getSucroseLevel(){

        return this.sucroseLevel;

    }

    public void printMedicineDetails(){

        //Or super.printMedicineDetails

        System.out.println("\t"+getClass()+" details");

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

        System.out.println("Formula: "+getFormula());

        System.out.println("Retail Price: "+getRetailPrice());

        System.out.println("Manufacture Date: "+getManufactureDate());

        System.out.println("Expiration Date: "+getExpirationDate());

        System.out.println("Sucrose level: "+this.sucroseLevel);

    }

}

public class Capsule extends Medicine{

    double absorptionPercentage;

    Capsule(String name, String formula, double retailPrice, String manufactureDate, String expirationDate, double absorptionPercentage){

        super(name, formula, retailPrice, manufactureDate, expirationDate);

        if(absorptionPercentage>=1.0 && absorptionPercentage<=100.0){

            this.absorptionPercentage=absorptionPercentage;

        }

    }

    public void setAbsorptionPercentage(double absorptionPercentage){

        if(absorptionPercentage>=1.0 && absorptionPercentage<=100.0){

            this.absorptionPercentage=absorptionPercentage;

        }

    }

    public double getAbsorptionPercentage(){

        return this.absorptionPercentage;

    }

    public void printMedicineDetails(){

        System.out.println("\t"+getClass()+" details");

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

        System.out.println("Formula: "+getFormula());

        System.out.println("Retail Price: "+getRetailPrice());

        System.out.println("Manufacture Date: "+getManufactureDate());

        System.out.println("Expiration Date: "+getExpirationDate());

        System.out.println("Sucrose level: "+this.absorptionPercentage);

    }

}

public class Syrup extends Medicine{

    Syrup(String name, String formula, double retailPrice, String manufactureDate, String expirationDate){

        super(name, formula, retailPrice, manufactureDate, expirationDate);

    }

    public void printMedicineDetails(){

        System.out.println("\t"+getClass()+" details");

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

        System.out.println("Formula: "+getFormula());

        System.out.println("Retail Price: "+getRetailPrice());

        System.out.println("Manufacture Date: "+getManufactureDate());

        System.out.println("Expiration Date: "+getExpirationDate());

    }

}

import java.util.Scanner;

public class Pharmacist {

    private Counter counter=new Counter();

    public boolean searchMedicine(Tablet[] t, Capsule[] c, Syrup[] s){

        Scanner obj = new Scanner(System.in);

        boolean found = false;

        System.out.println("Enter formula of medicine to search for: ");

        String formula = obj.nextLine();

        for(int i=0; i<2; i++){

            if(formula.compareTo(t[i].getFormula())==0){

                found=true;

                System.out.println("\tTablet found with formula: "+formula);

                System.out.println("Printing Details: ");

                t[i].printMedicineDetails();

                counter.updateRevenue(t[i].getRetailPrice());

            }

            if(formula.compareTo(c[i].getFormula())==0){

                found=true;

                System.out.println("\tMatch found");

                System.out.println("Printing Details: ");

                System.out.println("Tablet found with formula: "+formula);

                c[i].printMedicineDetails();

                counter.updateRevenue(c[i].getRetailPrice());

            }

            if(formula.compareTo(s[i].getFormula())==0){

                found=true;

                System.out.println("\tMatch found");

                System.out.println("Printing Details: ");

                System.out.println("Tablet found with formula: "+formula);

                s[i].printMedicineDetails();

                counter.updateRevenue(s[i].getRetailPrice());

            }

        }

        if(!found){

            System.out.println("Medicine with formula: "+formula+" not found");

            return false;

        }

        else{

            return true;

        }

    }

}

import java.util.Scanner;

public class Counter {

    public boolean searchMedicine(Tablet[] t, Capsule[] c, Syrup[] s){

        Scanner obj = new Scanner(System.in);

        boolean found = false;

        System.out.println("Enter name of medicine to search for: ");

        String name = obj.nextLine();

        for(int i=0; i<2; i++){

            if(name.compareTo(t[i].getName())==0){

                found=true;

                System.out.println("\tTablet found with name: "+name);

                System.out.println("Printing Details: ");

                t[i].printMedicineDetails();

                updateRevenue(t[i].getRetailPrice());

            }

            if(name.compareTo(c[i].getName())==0){

                found=true;

                System.out.println("\tMatch found");

                System.out.println("Printing Details: ");

                System.out.println("Tablet found with name: "+name);

                c[i].printMedicineDetails();

                updateRevenue(c[i].getRetailPrice());

            }

            if(name.compareTo(s[i].getName())==0){

                found=true;

                System.out.println("\tMatch found");

                System.out.println("Printing Details: ");

                System.out.println("Tablet found with name: "+name);

                s[i].printMedicineDetails();

                updateRevenue(s[i].getRetailPrice());

            }

        }

        if(!found){

            System.out.println("Medicine with name: "+name+" not found");

            return false;

        }

        else{

            return true;

        }

    }

    public void updateRevenue(double price){

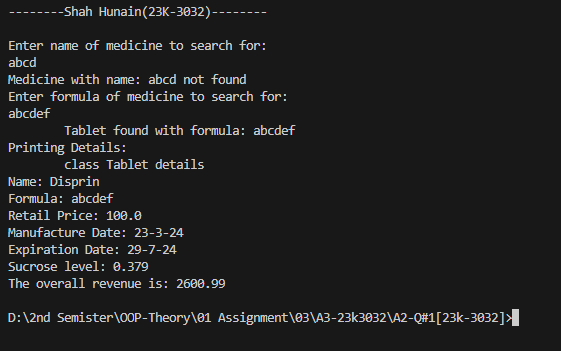
        TimesMedico.overallRevenue=TimesMedico.overallRevenue+price;

        System.out.println("The overall revenue is: "+TimesMedico.overallRevenue);

    }

}

**OUTPUT**

****

**Task 02**

**CODE**

public class CollectionOfPets{

    public static void main(String[] args) {

        System.out.println("\n--------Shah Hunain(23K-3032)--------\n");

        Cat<String,Integer> c = new Cat<String,Integer>("Dora", 5);

        Dog<String,Integer> d = new Dog<String,Integer>("Tommy", 7);

        Bird<String,Integer> b = new Bird<String,Integer>("Humming", 2);

        c.printDetails();

        c.makeSound();

        System.out.println();

        d.printDetails();

        d.makeSound();

        System.out.println();

        b.printDetails();

        b.makeSound();

        System.out.println();

    }

}

public abstract class Pet <T,U> {

    private T name;

    private U age;

    Pet(T name, U age){

        this.name=name;

        this.age=age;

    }

    abstract void makeSound();

    public T getName(){

        return this.name;

    }

    public U getAge(){

        return this.age;

    }

}

public class Cat <T,U> extends Pet <T,U> {

    Cat(T name, U age){

        super(name,age);

    }

    void makeSound(){

        System.out.println(getClass()+" meows");

    }

    public void printDetails(){

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

        System.out.println("Age: "+getAge());

    }

}

public class Dog <T,U> extends Pet <T,U>{

    Dog(T name, U age){

        super(name,age);

    }

    void makeSound(){

        System.out.println(getClass()+" woofs");

    }

    public void printDetails(){

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

        System.out.println("Age: "+getAge());

    }

}

public class Bird <T,U> extends Pet <T,U>{

    Bird(T name, U age){

        super(name,age);

    }

    void makeSound(){

        System.out.println(getClass()+" chirps");

    }

    public void printDetails(){

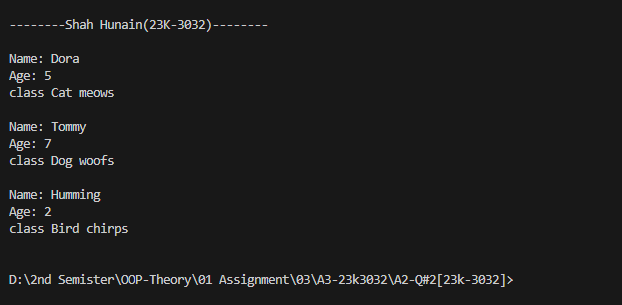
        System.out.println("Name: "+getName());

        System.out.println("Age: "+getAge());

    }

}

**OUTPUT**



**Task 03**

**CODE**

public class Main{

    public static void main(String[] args) {

        System.out.println("\n--------Shah Hunain(23K-3032)--------\n");

        Integer m1[][]={

                {1,2},

                {3,4},

                {5,6}

            };

        Matrix m = new Matrix<>(3, 2);

        m.setM(m1);

        m.display();

        Integer m2[][]={

            {0,1},

            {3,2},

            {5,4}

        };

        m.sum(m2, 3, 2);

        m.difference(m2, 3, 2);

        m.multiply(m2, 3, 2);

        Integer m3[][]={

            {1,2},

            {3,4},

        };

        Matrix<Integer> M = new Matrix<>(2, 2);

        M.setM(m3);

        Integer m4[][]={

            {0,1},

            {3,2},

        };

        M.multiply(m4, 2, 2);

    }

}

public class Matrix <T extends Number> {

    int rows;

    int columns;

    T[][] m;

    Matrix(int rows, int columns) {

        this.rows = rows;

        this.columns = columns;

        this.m= (T[][]) new Number[rows][columns];

    }

    public void setM(T[][] m) {

        this.m=m;

    }

    public T getMElement(int indexR, int indexC){

        return m[indexR][indexC];

    }

    public void setMElement(T value, int indexR, int indexC){

        this.m[indexR][indexC]=value;

    }

    public void display(){

        System.out.println("The matrix is: ");

        for(int i=0; i<rows; i++){

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

                System.out.print(" "+m[i][j]+"\t");

            }

            System.out.println();

        }

    }

    public void displayResultMatrix(T[][] matrix){

        System.out.println("The matrix is: ");

        for(int i=0; i<rows; i++){

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

                System.out.print(" "+matrix[i][j]+"\t");

            }

            System.out.println();

        }

    }

    public void sum(T[][] matrix , int r, int c){

        T[][] result = (T[][]) new Number[rows][columns];

        if(this.rows==r){

            if(this.columns==c){

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

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

                        result[i][j]= (T) (Double.valueOf(m[i][j].doubleValue()+matrix[i][j].doubleValue()));

                    }

                }

                displayResultMatrix(result);

            }

            else{

                System.out.println("ROWS and COLUMNS both should be same for addition for matrices");

        }

        }

        else{

            System.out.println("ROWS and COLUMNS both should be same for addition for matrices");

    }

    }

    public void difference(T[][] matrix , int r, int c){

        T[][] result = (T[][]) new Number[rows][columns];

        if(this.rows==r){

            if(this.columns==c){

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

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

                        result[i][j]= (T) (Double.valueOf(m[i][j].doubleValue()-matrix[i][j].doubleValue()));

                    }

                }

                displayResultMatrix(result);

            }

            else{

                System.out.println("ROWS and COLUMNS both should be same for subtraction of matrices");

            }

        }

        else{

                System.out.println("ROWS and COLUMNS both should be same for subtraction of matrices");

        }

    }

    public void multiply(T[][] matrix, int r, int c) {

        if (this.columns != r){

            System.out.println("Number of columns in the first matrix must be equal to the number of rows in the second matrix for multiplication.");

        }

        else{

            T[][] result = (T[][]) new Number[this.rows][this.columns];

            for (int i = 0; i < this.rows; i++) {

                for (int j = 0; j < matrix[0].length; j++) {

                    T sum = (T) Double.valueOf(0);

                    for (int k = 0; k < this.columns; k++) {

                        sum = (T) Double.valueOf(sum.doubleValue() + this.m[i][k].doubleValue() \* matrix[k][j].doubleValue());

                    }

                    result[i][j] = sum;

                }

            }

            displayResultMatrix(result);

        }

    }

}

public class IntMatrix extends Matrix <Integer> {

    IntMatrix(int noOfRows, int noOfColumns){

        super(noOfRows, noOfColumns);

    }

    public void display(){

        System.out.println(getClass()+"");

        super.display();

    }

}

public class DoubleMatrix extends Matrix <Double> {

    DoubleMatrix(int noOfRows, int noOfColumns){

        super(noOfRows, noOfColumns);

    }

    public void display(){

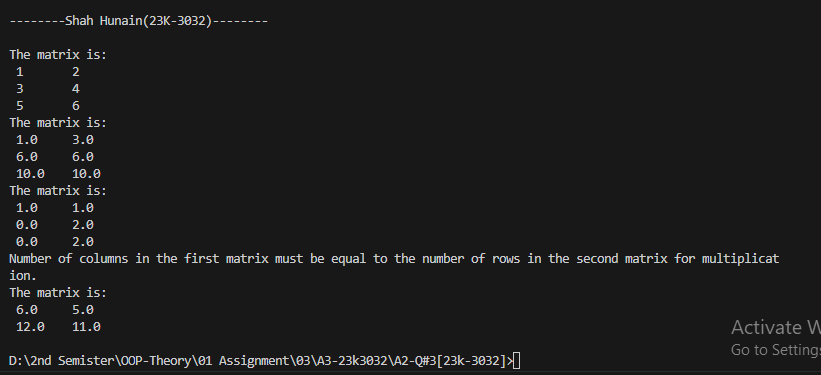
        System.out.println(getClass()+"");

        super.display();

    }

}

**OUTPUT**

****

**Task 04**

**CODE**

public class AutonomousDrones{

    public static void main(String[] args) {

        System.out.println("\n--------Shah Hunain(23K-3032)--------\n");

        try{

            ReconDrone d = new ReconDrone(40, 74, 100, 30, 16, 10);

            d.takeoff();

            d.navigateTo(46, 94, 150);

            d.scanArea(100);

            d.land();

        }

        catch(Exception e){

            System.out.println("Exception occurred");

            System.out.println(e);

        }

    }

}

public abstract class Drone {

    float latitude;

    float longitude;

    float altitude;

    float speed;

    Drone(float latitude, float longitude, float altitude, float speed){

        this.latitude=latitude;

        this.longitude=longitude;

        this.altitude=altitude;

        this.speed=speed;

    }

    abstract void adjustAltitude(float meters);

    abstract void setSpeed(float speed);

}

public interface Flyable {

    public void takeoff();

    public void land();

    public void navigateTo(float latitude, float longitude, float altitude);

}

public interface Scannable {

    public void scanArea(float radius);

}

public class ReconDrone extends Drone implements Flyable, Scannable{

    int cameraResolution;

    int maxFlightTime;

    ReconDrone(float latitude, float longitude, float altitude, float speed, int cameraResolution, int maxFlightTime){

        super(latitude, longitude, altitude, speed);

        this.cameraResolution=cameraResolution;

        this.maxFlightTime=maxFlightTime;

    }

    public void land(){

        System.out.println("ReconDrone is landing");

    }

    public void takeoff(){

        System.out.println("ReconDrone is taking off");

    }

    public void adjustAltitude(float meters){

        this.altitude=this.altitude+meters;

        System.out.println("Altitude: "+altitude);

    }

    public void setSpeed(float speed){

        this.speed=speed;

        System.out.println("Speed: "+speed);

    }

    public void navigateTo(float latitude, float longitude, float altitude){

        System.out.println("Navigating to latitude: "+latitude+" longitude: "+longitude+" altitude: "+altitude);

        float differentLat=this.latitude-latitude;

        float differentLon=this.longitude-longitude;

        float differentAlt=this.altitude-altitude;

        float distance = differentAlt + differentLat + differentLon;

        distance=Math.abs(distance);

        float time= distance/this.speed;

        time=Math.abs(time);

        System.out.println("Time to reach: "+time);

        this.latitude=latitude;

        this.longitude=longitude;

        this.altitude=altitude;

    }

    public void scanArea(float radius){

        System.out.println("Radius: "+radius+"is geeting scanned");

        int detectedObjects=3;

        System.out.println("Number of objects detected: "+detectedObjects);

        float lat=18;

        float lon=28;

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

            System.out.println((i+1)+" object detected at latitude: "+lat+" and longitude: "+lon);

            lat+=9;

            lon+=5;

        }

    }

}

**OUTPUT**

