**COMP471 JAVA PROGRAMMING**

**HOMEWORK**

**Spring 2020-21**

**DUE (3/6/2021 23:55PM)**

174101 – Kristina Grigoryeva

In this java project I use additional referenced libraries:

**import** com.github.lgooddatepicker.components.TimePicker;

**import** net.sourceforge.jdatepicker.impl.JDatePanelImpl;

**import** net.sourceforge.jdatepicker.impl.JDatePickerImpl;

**import** net.sourceforge.jdatepicker.impl.UtilDateModel;

**Please state how you differentiate apple and orange objects since you just select fruit. Do you need to do anything about that?**

In order to differentiate apple from orange I created 2 classes Apple and Orange. In the **class** pan I created function ss(). When user will choose category Fruit in the panel he will need to write name of the fruit.

**void** ss(ArrayList<Plant> wat,String sss){

String s = "";

System.***out***.println(sss);

**if**(sss.equals("Fruit")){

**for** (Plant x : wat){

Then, getClass() method will return an object and getSimpleName() returns class which equals Apple or Citrus.

**if**(x.getClass().getSimpleName().equals("Apple")||x.getClass().getSimpleName().equals("Citrus") ){

s = s + x.toString()+'\n';

System.***out***.println(s);

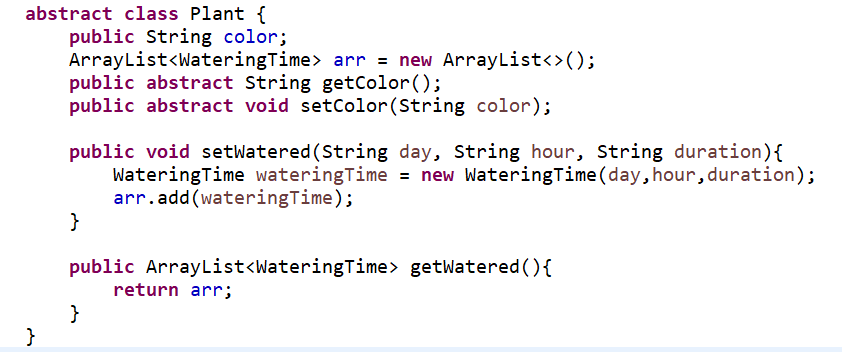
}

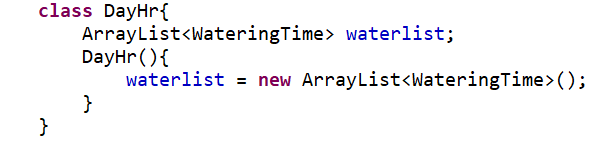
}

}

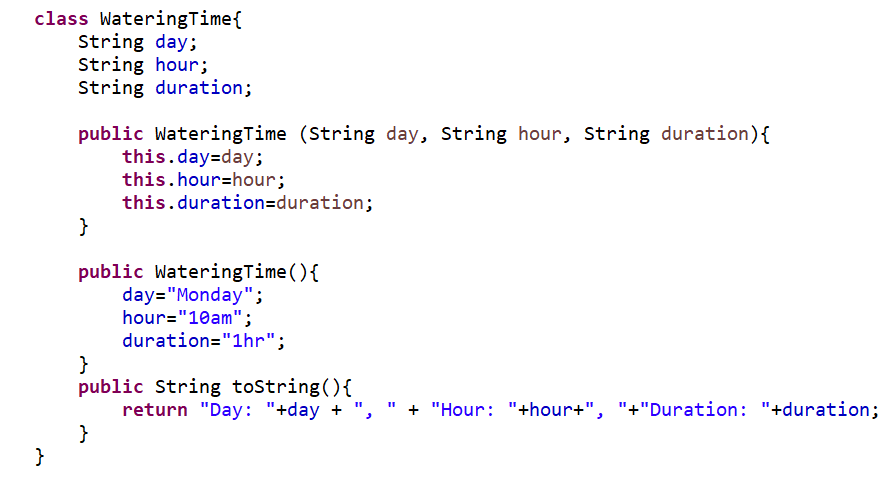
1. Create an abstract Plant class, which is the abstract superclass of Fruit, Vegetable and Flower classes.
2. implement setWatered and getWatered fuctiontions for the plant. Note that DayHr is a class which holds the list of watering time for a week( in terms of day and hour and duration) the plant should be watered. The default plan is set for 10 am on Mondays and the duration is 1 hr.
3. Define an abstract fuctions setColor and getColor which they are used to set and get color of plant

1) Here I created an abstract class Plant with 2 functions: setWatered() and getWatered(). In order to output information about watering time we use ArrayList which can resize. After we input data it will go to the function setWatered(), then creates new object and data from WateringTime will push to ArrayList. When the function getWatered starts to work it will output array converted to string + new WateringTime().toString();





Also, I added WateringTime class which will contain default plan for 10 am on Mondays and the duration is 1 hr.



When class Flower extends class Plant we need to use class DayHr to collect data from Flower\_Panel flower.setWatered(datePicker.getJFormattedTextField().getText(),timePicker1.getTimeStringOrEmptyString(),(dur.getText() + " min"));

DayHr d = **new** DayHr();

@Override

**public** **void** setWatered(String day, String hour, String duration) {

d.waterlist.add(**new** WateringTime(day,hour,duration));

}

@Override

**public** ArrayList<WateringTime> getWatered() {

**return** d.waterlist;

}

2) There are 2 functions in Plant class

**public** **abstract** String getColor();

**public** **abstract** **void** setColor(String color);

Later we will use these functions in Flower, Citrus, Apple and Tomato classes:

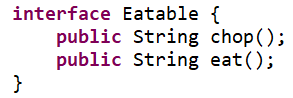
**public** String getColor() {

**return** **super**.color;}

@Override

**public** **void** setColor(String color) {**super**.color = color;}

1. Create an interface Eatable with chop and eat functions



1. Create Flower subclass of a Plant class.
   1. The Flower class has a name attribute of a class and a smell intensity which intensity should be enumerated from 1 to 5 as very strong, strong, middle, weak, very weak
   2. Create set and get methods for intensity

1) **class** Flower **extends** Plant {

String name;

intensivety smell;

**public** Flower(String name) {

**this**.name = name;

}

}

I created Plant class and for intensivety I use enum class with constant variables and then I push this value as smell in the Panel.

**num** intensivety {

***VERYSTRONG***,

***STRONG***,

***MIDDLE***,

***WEAK***,

***VERYWEAK***

}

2). @Override

**public** String toString() {

**return** "Flower{" +

"name='" + name + '\'' +

", color='" + color + '\'' +

", smell=" + smell +

", d=" + d.waterlist.toString() +

'}';

}

1. Create an abstract Fruit class.
   1. Write set and get price functions for a fruit.
   2. Implement chop function and write a string “ don’t chop the fruit byte it”
   3. Implement eat function of a class and write string “ you should eat fruits raw”

**abstract** **class** Fruit **extends** Plant{

**int** price;

1)**public** **void** setPrice(**int** price){

**this**.price=price;

}

**public** **int** getPrice(){

**return** **this**.price;

}

2)**public** String chop(){

**return** "Don't chop the fruit byte it";

}

3)**public** String eat(){

**return** "you should eat fruits raw";

}

}

1. Create an abstract Vegetable class.
   1. Write set and get weight functions for a vegetable.
   2. Implement chop function and write a string “ chop the vegetable don’t byte it”
   3. Implement eat function of a class and write string “ you should eat vegetables cooked”

**abstract** **class** Vegetable **extends** Plant **implements** Eatable {

1. **int** Weight;

**public** **void** setWeight(**int** weight){

**this**.Weight=weight;

}

**public** **int** getWeight(){

**return** **this**.Weight;

}

2) **public** String chop() {

**return** "chop the vegetable don't byte it";

}

3) **public** String eat(){

**return** "you should eat vegetable cooked";

}

}

1. Create a citrus class which is a subclass of a Fruit
   1. By citrus function set the name of the citrus
   2. Set and get the color of the citrus

**class** Citrus **extends** Fruit{

1. String cname;

Citrus(String cname){

**this**.cname=cname;

}

1. **public** **void** setColor(String color){

**this**.color=color;

}

**public** String getColor(){

**return** **this**.color;

}

}

1. Create an apple class which is a subclass of a Fruit
   1. By apple function set the name of the apple
   2. Set and get the color of the apple

**class** Apple **extends** Fruit{

1. String aname;

Apple(String aname){

**this**.aname=aname;

}

1. **public** **void** setColor(String color){

**this**.color=color;

}

**public** String getColor(){

**return** **this**.color;

}

}

1. Create a subclass Tomato which is a subclass of Vegetable
   1. In tomato function determine the type of tomato.
   2. Override setwatered and getwatered functions and schedule a plan for Monday, Wednesday and Friday from 9am to 9.30 am
   3. Make a list of products which we can make from tomato ( ex: tomato soup, tomato paste, ketcup)

**class** Tomato **extends** Vegetable{

1. String kind;

Tomato(String kind){

**this**.kind=kind;

}

3)String[] products = {"tomato soup", "ketchup"} ;

**public** ArrayList<String> products(String product){

ArrayList<String>arr = **new** ArrayList<>();

**return** arr;

}

String day= "Monday, Wednesday, Friday";

String hour="from 9am to 9.30 am";

String duration="30 min";

ArrayList<WateringTime>ar = **new** ArrayList<>();

@Override

**public** **void** setWatered(String day, String hour, String duration){

ar.add(**new** WateringTime(day, hour, duration));

}

@Override

**public** ArrayList<WateringTime> getWatered(){

**return** ar;

}

}

package kristina;

import java.awt.event.ActionEvent;

import java.awt.event.ActionListener;

import java.util.ArrayList;

import java.util.Arrays;

import java.util.Locale;

import javax.swing.border.LineBorder;

import com.github.lgooddatepicker.components.TimePicker;

import net.sourceforge.jdatepicker.impl.JDatePanelImpl;

import net.sourceforge.jdatepicker.impl.JDatePickerImpl;

import net.sourceforge.jdatepicker.impl.UtilDateModel;

import javax.swing.\*;

import java.awt.\*;

    interface Eatable {

        public String chop();

        public String eat();

    }

    abstract class Plant {

        public String color;

        ArrayList<WateringTime> arr = new ArrayList<>();

        public abstract String getColor();

        public abstract void setColor(String color);

        public void setWatered(String day, String hour, String duration){

            WateringTime wateringTime = new WateringTime(day,hour,duration);

            arr.add(wateringTime);

        }

        public ArrayList<WateringTime> getWatered(){

            return arr;

        }

    }

    class DayHr{

        ArrayList<WateringTime> waterlist;

        DayHr(){

            waterlist = new ArrayList<WateringTime>();

        }

    }

    enum intensivety {

        VERYSTRONG,

        STRONG,

        MIDDLE,

        WEAK,

        VERYWEAK

    }

    class WateringTime{

        String day;

        String hour;

        String duration;

        public WateringTime (String day, String hour, String duration){

            this.day=day;

            this.hour=hour;

            this.duration=duration;

        }

        public WateringTime(){

            day="Monday";

            hour="10am";

            duration="1hr";

        }

        public String toString(){

            return "Day: "+day + ", " + "Hour: "+hour+", "+"Duration: "+duration;

        }

    }

    class Flower extends Plant{

        String name;

        DayHr d = new DayHr();

        intensivety smell;

        public Flower(String name) {

            this.name = name;

        }

        public String getColor() {

            return super.color;

        }

        @Override

        public void setColor(String color) {

            super.color = color;

        }

        @Override

        public void setWatered(String day, String hour, String duration) {

            d.waterlist.add(new WateringTime(day,hour,duration));

        }

        @Override

        public ArrayList<WateringTime> getWatered() {

            return d.waterlist;

        }

        @Override

        public String toString() {

            return "Flower{" +

                    "name='" + name + '\'' +

                    ", color='" + color + '\'' +

                    ", smell=" + smell +

                    ", d=" + d.waterlist.toString() +

                    '}';

        }

    }

     class Apple extends Fruit{

        String aname;

        Apple(String aname){

            this.aname=aname;

        }

        public void setColor(String color){

            this.color=color;

        }

        public String getColor(){

            return this.color;

        }

         @Override

         public String toString() {

             return "Apple{" +

                     "name='" + aname + '\'' +

                     ", color='" + color + '\'' +

                     '}' + "\r\n" + new WateringTime().toString();

         }

     }

     class Citrus extends Fruit{

        String cname;

        Citrus(String cname){

            this.cname=cname;

        }

        public void setColor(String color){

            this.color=color;

        }

        public String getColor(){

            return this.color;

        }

         @Override

         public String toString() {

             return "Citrus{" +

                     "name='" + cname + '\'' +

                     ", color='" + color + '\'' +

                     '}'  + "\r\n" + new WateringTime().toString();

         }

     }

     abstract class Vegetable extends Plant implements Eatable {

        int Weight;

        public void setWeight(int weight){

            this.Weight=weight;

        }

        public int getWeight(){

            return this.Weight;

        }

        public String chop() {

            return "chop the vegetable don't byte it";

        }

        public String eat(){

            return "you should eat vegetable cooked";

        }

    }

     abstract class Fruit extends Plant implements Eatable  {

        int price;

        public void setPrice(int price){

            this.price=price;

        }

        public int getPrice(){

            return this.price;

        }

        public String chop(){

            return "Don't chop the fruit byte it";

        }

        public String eat(){

            return "you should eat fruits raw";

        }

    }

     class Tomato extends Vegetable{

        String kind;

        String[] products = {"tomato soup", "ketchup"} ;

        String day= "Monday, Wednesday, Friday";

        String hour="from 9am to 9.30 am";

        String duration="30 min";

        ArrayList<WateringTime>ar = new ArrayList<>();

        Tomato(String kind){

            this.kind=kind;

        }

        public ArrayList<String> products(String product){

            ArrayList<String>arr = new ArrayList<>();

           return arr;

        }

         @Override

         public String getColor() {

             return super.color;

         }

         @Override

         public void setColor(String color) {

                this.color = color;

         }

         @Override

         public void setWatered(String day, String hour, String duration){

            ar.add(new WateringTime(day, hour, duration));

        }

         @Override

        public ArrayList<WateringTime> getWatered(){

            return ar;

        }

         @Override

         public String toString() {

             return "Tomato{" +

                     "color='" + color + '\'' +

                     ", kind='" + kind + '\'' +

                     ", products='" + Arrays.toString(products) + '\'' +

                     '}' + "\r\n" + new WateringTime(day,hour,duration).toString();

         }

     }

    class Flower\_Panel  extends JPanel{

        /\*\*

         \*

         \*/

        private static final long serialVersionUID = 1L;

        JLabel name = new JLabel("Enter flower name");

        JTextField nametf = new JTextField();

        JLabel color = new JLabel("Enter colour name");

        JTextField colortf = new JTextField();

        JLabel smell = new JLabel("Smell intensity");

        JComboBox<String> c = new JComboBox<String>();

        UtilDateModel model = new UtilDateModel();

        JDatePanelImpl datePanel = new JDatePanelImpl(model);

        JDatePickerImpl datePicker = new JDatePickerImpl(datePanel);

        TimePicker timePicker1 = new TimePicker();

        JLabel duration = new JLabel("Duration");

        JLabel datetime = new JLabel("Date/Time");

        JTextField dur = new JTextField();

        JButton submit1 = new JButton("Submit");

        public Flower\_Panel(ArrayList<Plant> wat) {

            name.setBounds(20,20,150,40);

            nametf.setBounds(150,20,150,40);

            color.setBounds(20,80,150,40);

            colortf.setBounds(150,80,150,40);

            smell.setBounds(20,150,150,40);

            c.setBounds(150,150,150,40);

            datetime.setBounds(20,250,100,40);

            datePicker.setToolTipText("date");

            datePicker.setBounds(120,250,150,60);

            timePicker1.setBounds(280,250,150,30);

            timePicker1.setText("time");

            duration.setBounds(440,250,100,30);

            dur.setBounds(550,250,100,30);

            submit1.setBounds(50,450,100,40);

            add(submit1);

            add(datetime);

            add(duration);

            add(name);

            add(nametf);

            add(colortf);

            add(color);

            add(smell);

            add(c);

            add(datePicker);

            add(timePicker1);

            add(dur);

             String[] description = { "VERYSTRONG",

                     "    STRONG" ,

                     "    MIDDLE" ,

                     "    WEAK"

                     ,"VERYWEAK"

             };

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

                c.addItem(description[i]);

            }

            LineBorder border = new LineBorder(Color.BLACK, 2);

            submit1.addActionListener(new ActionListener() {

                @Override

                public void actionPerformed(ActionEvent e) {

                    String s = "";

                    s = s + nametf.getText() + " " +  colortf.getText() + " " + c.getSelectedItem().toString() + " " + datePicker.getJFormattedTextField().getText() + " " + timePicker1.getTimeStringOrEmptyString()+" "+ " "+dur.getText()+" min";

                    System.out.println(s);

                    Flower flower = new Flower(nametf.getText());

                    flower.setColor(colortf.getText());

                    if(c.getSelectedItem().equals("VERYWEAK")){

                        flower.smell = intensivety.VERYWEAK;

                    }

                    else if(c.getSelectedItem().equals("MIDDLE")){

                        flower.smell = intensivety.MIDDLE;

                    }

                    else if(c.getSelectedItem().equals("STRONG")){

                        flower.smell = intensivety.STRONG;

                    }

                    else if(c.getSelectedItem().equals("VERYSTRONG")){

                        flower.smell = intensivety.VERYSTRONG;

                    }

                    else {

                        flower.smell = intensivety.WEAK;

                    }

                    flower.setColor(colortf.getText());

                    flower.setWatered(datePicker.getJFormattedTextField().getText(),timePicker1.getTimeStringOrEmptyString(),(dur.getText() + " min"));

                    wat.add(flower);

                    nametf.setText("");

                    colortf.setText("");

                    setVisible(false);

                }

            });

            setLayout(null);

            setBorder(border);

            setVisible(false);

        }

    }

    class pan extends JPanel{

        /\*\*

         \*

         \*/

        private static final long serialVersionUID = 1L;

        JTextArea textArea = new JTextArea();

        void ss(ArrayList<Plant> wat,String sss){

            String s = "";

            System.out.println(sss);

            if(sss.equals("Fruit")){

                for (Plant x : wat){

                    if(x.getClass().getSimpleName().equals("Apple")||x.getClass().getSimpleName().equals("Citrus") ){

                        s = s + x.toString()+'\n';

                        System.out.println(s);

                    }

                }

            }

            if(sss.equals("Vegetables")){

                for (Plant x : wat){

                    if(x.getClass().getSimpleName().equals("Tomato") ){

                        s = s + x.toString()+'\n';

                        System.out.println(s);

                    }

                }

            }

            if(sss.equals("Flower")){

                for (Plant x : wat){

                    if(x.getClass().getSimpleName().equals("Flower") ){

                        s = s + x.toString()+'\n';

                        System.out.println(s);

                    }

                }

            }

            textArea.setText(s);

            setVisible(true);

        }

        public pan(ArrayList<Plant> wat) {

            textArea.setBounds(20,20,600,600);

            String s = "";

            for(Plant x : wat){

                s = s + x.toString();

            }

            textArea.setText(s);

            add(textArea);

            setBorder(new LineBorder(Color.BLACK,2));

            setBounds(450,50,700,700);

            setLayout(null);

            setVisible(false);

        }

    }

    class FruitP extends JPanel{

        /\*\*

         \*

         \*/

        private static final long serialVersionUID = 1L;

        JLabel name = new JLabel("Enter name of Fruit");

        JTextField nametf = new JTextField();

        JLabel color = new JLabel("Enter colour of Fruit  ");

        JTextField colortf = new JTextField();

        JButton submit1 = new JButton("Submit");

        public FruitP(ArrayList<Plant> wat) {

            name.setBounds(20,20,170,40);

            nametf.setBounds(150,20,170,40);

            color.setBounds(20,80,170,40);

            colortf.setBounds(150,80,170,40);

            submit1.setBounds(50,450,100,40);

            add(submit1);

            add(name);

            add(nametf);

            add(colortf);

            add(color);

            setBorder(new LineBorder(Color.BLACK,2));

            setBounds(450,50,700,700);

            submit1.addActionListener(new ActionListener() {

                @Override

                public void actionPerformed(ActionEvent e) {

                    Plant plant;

                    if(nametf.getText().toLowerCase(Locale.ROOT).equals("apple")){

                        plant = new Apple(nametf.getText());

                    }else{

                        plant = new Citrus(nametf.getText());

                    }

                    plant.setColor(colortf.getText());

                    wat.add(plant);

                    nametf.setText("");

                    colortf.setText("");

                    setVisible(false);

                }

            });

            setLayout(null);

            setVisible(false);

        }

    }

    class Veget extends JPanel{

        /\*\*

         \*

         \*/

        private static final long serialVersionUID = 1L;

        JLabel name = new JLabel("Enter name of Vegetable");

        JTextField nametf = new JTextField();

        JLabel color = new JLabel("Enter colour of Vegetable  ");

        JTextField colortf = new JTextField();

        JButton submit1 = new JButton("Submit");

        public Veget(ArrayList<Plant> wat) {

            name.setBounds(20,20,170,40);

            nametf.setBounds(200,20,170,40);

            color.setBounds(20,80,170,40);

            colortf.setBounds(200,80,170,40);

            submit1.setBounds(50,450,100,40);

            add(submit1);

            add(name);

            add(nametf);

            add(colortf);

            add(color);

            setBorder(new LineBorder(Color.BLACK,2));

            setBounds(450,50,700,700);

            submit1.addActionListener(new ActionListener() {

                @Override

                public void actionPerformed(ActionEvent e) {

                    Vegetable vegetable = new Tomato(nametf.getText());

                    vegetable.setColor(colortf.getText());

                    wat.add(vegetable);

                    nametf.setText("");

                    colortf.setText("");

                    setVisible(false);

                }

            });

            setLayout(null);

            setVisible(false);

        }

    }

    class Frame extends JFrame{

        /\*\*

         \*

         \*/

        private static final long serialVersionUID = 1L;

        JButton add1;

        JButton search;

        JLabel jLabel;

        Flower\_Panel flower\_panel;

        FruitP ff;

        Veget vv;

        DefaultListModel<String> l1 = new DefaultListModel<>();

        JComboBox<String> c = new JComboBox<String>();

        pan p;

        private String[] description = { "Flower", "Fruit", "Vegetables"};

        public Frame(ArrayList<Plant> wat) throws HeadlessException {

            flower\_panel = new Flower\_Panel(wat);

            p = new pan(wat);

            ff = new FruitP(wat);

            vv = new Veget(wat);

            setSize(1100, 700);

            add1 = new JButton("Add");

            search = new JButton("Search");

            jLabel = new JLabel("Select Plant");

            for (int i = 0; i < 3; i++){

                c.addItem(description[i]);

            }

            add1.setBounds(50,400,150,50);

            search.setBounds(200,400,150,50);

            jLabel.setBounds(50,100,150,50);

            c.setBounds(150,100,150,50);

            flower\_panel.setBounds(350,50,700,700);

            add(flower\_panel);

            p.setBounds(350,50,700,700);

            vv.setBounds(350,50,700,700);

            ff.setBounds(350,50,700,700);

            add(add1);

            add(search);

            add(jLabel);

            add(c);

            add(p);

            add(ff);

            add(vv);

            add(flower\_panel);

            add1.addActionListener(new ActionListener() {

                @Override

                public void actionPerformed(ActionEvent e) {

                    if(c.getSelectedItem().toString().equals("Fruit")){

                        ff.setVisible(true);

                        flower\_panel.setVisible(false);

                        p.setVisible(false);

                        System.out.println("fruit");

                        vv.setVisible(false);

                    }

                    if(c.getSelectedItem().toString().equals("Flower")){

                        flower\_panel.setVisible(true);

                        ff.setVisible(false);

                        p.setVisible(false);

                        System.out.println("flower");

                        vv.setVisible(false);

                    }

                    if(c.getSelectedItem().toString().equals("Vegetables")){

                        vv.setVisible(true);

                        ff.setVisible(false);

                        flower\_panel.setVisible(false);

                        p.setVisible(false);

                        System.out.println("vegetable");

                    }

                }

            });

            search.addActionListener(new ActionListener() {

                @Override

                public void actionPerformed(ActionEvent e) {

                    flower\_panel.setVisible(false);

                    ff.setVisible(false);

                    p.ss(wat,c.getSelectedItem().toString());

                }

            });

            setLayout(null);

            setVisible(true);

        }

    }

    public class kristina {

        public static void main(String[] args) {

            ArrayList<Plant> wat = new ArrayList<Plant>();

            new Frame(wat);

        }

    }