Probleme propuse: Pentru toate variantele extindeţi capacităţile clasei de bază dată folosind moştenirea:
1.Time, astfel încât să se poată deduce timpul zilei: dimineaţă, seară e.t.c
2.Time, astfel încât să se poată deduce in secunde, minute timpul curent.
3.Time, astfel încât să se poată deduce timpul trecut de la o acţiune specificata.
4.Date, astfel încât să se poată deduce anotimpurile anului: iarna ,vara e.t.c.
5.Annotation, astfel încât să se poată deduce timpul in secunde de la ultima modificare a annotaţiei.
6.Dictionary, astfel încât să se poată deduce data în zile a ultimei modificări a dicţionarului.
7.File, astfel încât să se poată deduce timpul si data creării obiectului.
8.File, astfel încât să se poată deduce mărimea obiectului după modificare.
9.Stack, astfel încât să se poată deduce timpul in minute a ultimei sesiuni a lucrului cu stiva.
10.Stack, astfel încât să se poată deduce data în zile a ultimei sesiuni a lucrului cu stiva.
11.Annotation, astfel încât să se poată deduce data in zilele de la ultima modificare a annotaţiei.
12.Annotation, astfel încât să se poată deduce timpul şi data modificării annotaţiei.

Problema 1

Time, astfel încât să se poată deduce timpul zilei: dimineaţă, seară etc.

```
import java.awt.*;
import java.awt.event.*;
import java.lang.annotation.*;
import java.lang.instrument.Instrumentation;
import java.util.*;
import javax.annotation.*;
import javax.swing.*;
public class lab_3 {
    public static void main(String[] arg){
      MyFrame ff = new MyFrame();
      ff.setVisible(true);
    }
}

class Time {
      private Date dat;
      Time() {
            this.setTime();
      }
      public void setTime() {
            dat = new Date();
      }
      public Date getTime() {
            return dat;
      }
      public String getPeriod() {
            String str = "noapte";
            int h = dat.getHours();
            if (h > 5 && h<11) str = "dimineata";
            if (h>=11 && h<16) str = "ziua";
            if (h>=16 && h<20) str = "seara";

            return str;
      }
}

class File {
      protected String com;
//  private Date dat;
      protected String user;
      protected Time dat;
      protected String dimension;
```

```
        File() {

        }

        File(String com, String user, Time dat) {
                this.com=com;
                this.user=user;
                this.dat=dat;
        }
        public String fileDimension(String com, String user, Time dat) {
                String str = "0 KB";
                int dimension;
                File f = new File(com, user, dat);

                str = "Size: " + com.length() + user.length();

                System.out.println(str);
                return str;
        }

}

class MyAnnotation extends File implements Generated {

    public MyAnnotation(){
        this.setAnnotation();
    }
    public MyAnnotation(String cCom, String uUser){
        this.setAnnotation(cCom, uUser);
    }
    public void setAnnotation(){
        com = "Introduceti comentariul!";
        user = "Introduceti numele autorul!";
        dat = new Time();
        dimension = fileDimension(com, user, dat);
    }
    public void setAnnotation(String cCom,String uUser){
        com = cCom;
        user = uUser;
        dat = new Time();
        dimension = fileDimension(com, user, dat);
    }
    public String date() {
        return dat.getTime().toString() + ", " + dat.getPeriod();
    }
    public String comments() {
        return com;
    }
    public String users(){
        return user;
    }
    public String getAnnotation(){
        String output;
        output = "Autor: " + user + "\n";
        output += "Text: " + com + "\n";
        output += "Data: " + date() + "\n";
        output += "Dimension: " + dimension + "bytes \n";
        output += "=========================\n";
        return output;
    }
    public Class <? extends Annotation>  annotationType() {
        throw new UnsupportedOperationException("Not supported yet.");
    }
```

```
      public String[] value() {
         throw new UnsupportedOperationException("Not supported yet.");
     }
}

class MyFrame extends JFrame {
      private JTextArea history;
    private JPanel left, center, right;
    private JTextField user;
    private JTextArea com;
    private JLabel dat;
    private MyAnnotation ts;

    public MyFrame(){
        this.setTitle("Laborator 3");
        this.setSize(800, 500);
        this.setLocation(300, 300);
        this.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        ts = new MyAnnotation();
        user = new JTextField(ts.users(),10);
        left = new JPanel();
        Box b1 = Box.createVerticalBox();
        JButton but = new JButton("Submit");
        but.addActionListener(new ActionListener(){
             public void actionPerformed(ActionEvent e){
                    System.out.println("aa");
                    ts.setAnnotation(com.getText(), user.getText());
                    dat.setText("Data: "+ts.date());
                    history.setText(history.getText()+ts.getAnnotation());
            }
        });
        user = new JTextField(ts.users(),5);
        dat = new JLabel("Data: "+ts.date());
        b1.add(new JLabel("Autor:"));
        b1.add(user);
        b1.add(dat);
        b1.add(but);
        left.add(b1);
        this.add(left,BorderLayout.WEST);
        center = new JPanel();
        com = new JTextArea(ts.comments(),20,20);
        center.add(com);
        center.setBackground(Color.LIGHT_GRAY);
        this.add(center,BorderLayout.CENTER);
        right = new JPanel();
        history = new JTextArea(ts.getAnnotation(),20,20);
        right.add(history);
        this.add(right,BorderLayout.EAST);
    }
}
```

Problema 1   Varianta 2

```
package laboratorul2;
import java.util.Scanner;
public class lab1 {

    public static void main(String[] args) {
        Scanner gosa = new Scanner(System.in);
        System.out.println("Introduceti Ora, minute, secundele");
        int hour = gosa.nextInt();
        int minute = gosa.nextInt();
        int seconds = gosa.nextInt();

        myTime mt1 = new myTime(hour,minute,seconds);
        //myTime mt2 = new myTime(hour,minute);
        mt1.partOfDays();
        //mt2.partOfDays();
    } }
class Time{
        Time(int hour){
        this(hour, 0);
    }
     Time(int hour, int minute){
    this(hour,minute,0);
    }
    Time(int hour, int minute, int seconds){
     this.hour = hour;
     this.minute= minute;
     this.seconds = seconds;
    }
    int hour;
    int minute;
    int seconds;
}
class myTime extends Time{
    myTime(int hour){
        super(hour);
    }
    myTime(int hour, int minute){
    super(hour, minute);
    }
    myTime(int hour, int minute, int seconds){
     super(hour,minute,seconds);
    }
    void partOfDays(){
        this.checkTime();
        if(hour >= 6 && hour <= 11){
            System.out.println("Now is Morning, Time  = " + hour + " : " + minute + " : " +
seconds);
        }
        else if(hour > 11 && hour <= 17){
            System.out.println("Now is Days, Time = " + hour + " : " + minute + " : " +
seconds);
        }
        else if(hour > 17 && hour <= 24 || hour >= 0 && hour <6){
            System.out.println("Now is Night, Time = " + hour + " : " + minute + " : " +
seconds);
        }
    }
    void checkTime(){
        while(seconds > 60 || minute > 60 || hour > 24){
        if(seconds >= 60){
            minute ++;
            seconds -= 60;
        }
        if(minute >=60){
            hour ++;
            minute -=60;
        }
        if(hour > 24){
            System.out.println("Ati introdus o ora gresita");
        }}
```

Problema 5

Annotation, astfel încât să se poată deduce timpul in secunde de la ultima modificare a annotaţiei.

```
package lab2;
import javax.annotation.Generated;
import javax.swing.*;
import java.awt.*;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;
import java.util.Date;

public class lab3 {
    public static void main(String[] arg){
        MyFrame ff = new MyFrame();
        ff.setVisible(true);
    }
}

class MyAnnotation implements Generated {
    private String com;
    private Date dat;
    private String user;

    public MyAnnotation(){
        this.setAnnotation();
    }
    public MyAnnotation(String cCom, String uUser){
        this.setAnnotation(cCom, uUser);
    }

    public void setAnnotation(){
        com = "I like quarantine!";
        user = "Mary";
        dat = new Date();
    }
    public void setAnnotation(String cCom, String uUser){
        com = cCom;
        user = uUser;
        dat = new Date();
    }

    public String date() {
        return dat.toString();
    }
    public String comments() {
        return com;
    }
    public String users(){
        return user;
    }

    public String getAnnotation(){
        String output;
        output = "Autor: " + user + "\n";
        output += "Text: " + com + "\n";
        output += "Data: " + date() + "\n";
        output += "  \n";
        return output;
    }
    public Class <? extends Annotation>  annotationType() {
        throw new UnsupportedOperationException("Not supported yet.");
    }
    public String[] value() {
        throw new UnsupportedOperationException("Not supported yet.");
    }
}
```

```java
//Clasa Annotation extinde clasa MyAnnotation, si adauga timpul de la ultima modificare a
anotatiei
class Annotation extends MyAnnotation {
    private long lStartTime;
    private long lEndTime;

    @Override
    public void setAnnotation() {
        lStartTime = new Date().getTime();
    }
    @Override
    public void setAnnotation(String cCom, String uUser) {
        lStartTime = new Date().getTime();
    }

    @Override
    public String getAnnotation(){
        lEndTime = new Date().getTime();
        String output = "Time from previous modification in sec: " + ((lEndTime -
lStartTime)/1000) + "\n";
        output += "==================================\n";
        lStartTime = new Date().getTime();
        return output;
    }

}

class MyFrame extends JFrame {
    private JTextArea history;
    private JPanel left, center, right;
    private JTextField user;
    private JTextArea com;
    private JLabel dat;
    private MyAnnotation ts;
    private Annotation tin;

    public MyFrame(){
        this.setTitle("Lab_3");
        this.setSize(600, 500);
        this.setLocation(300, 300);
        this.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        ts = new MyAnnotation();
        tin = new Annotation();
        user = new JTextField(ts.users(),10);
        left = new JPanel();
        Box b1 = Box.createVerticalBox();
        JButton but = new JButton("Submit");
        but.addActionListener(new ActionListener(){
                    public void actionPerformed(ActionEvent e){
                        System.out.println("aa");
                        ts.setAnnotation(com.getText(), user.getText());
                        dat.setText("Data: "+ts.date());
                        history.setText(history.getText()+
ts.getAnnotation()+tin.getAnnotation());
                    }  });
        user = new JTextField(ts.users(),5);
        dat = new JLabel("Data: "+ts.date());
        b1.add(new JLabel("Autor:"));
        b1.add(user);
        b1.add(dat);
        b1.add(but);
        left.add(b1);
        this.add(left,BorderLayout.WEST);
        center = new JPanel();
        com = new JTextArea(ts.comments(),20,20);
        center.add(com);
        center.setBackground(Color.red);
        this.add(center,BorderLayout.CENTER);
        right = new JPanel();
        history = new JTextArea(ts.getAnnotation(),20,20);
        right.add(history);
        this.add(right,BorderLayout.EAST);
    }
```

Problema 5 Variant 2

```
import java.util.*;


interface Myinterface{
    public void PublicateNote();
}
class Myclass implements Myinterface {

    String Author;
    String News;
    Date Date;
    @Override
    public void PublicateNote() {
        Scanner in = new Scanner(System.in);
        System.out.println("Author name: " ); Author = in.next();
        System.out.println("Text: " ); News = in.next();
        Date = new Date();
        this.ToString();
    }
    public void ToString(){
        System.out.println("Author name: " + this.Author);
        System.out.println("Text: " + this.News);
        System.out.println("Changed Time: " + Date.toString());
    }
}
public class Main {
    public static void main(String[] arg) {

        ArrayList<Myclass> program = new ArrayList<Myclass>();
        while(true) {
            Scanner in = new Scanner(System.in);
           System.out.println("Do you want to introduce a news?");
           String answer = in.next();
            answer.toLowerCase();
            switch (answer){
                case "yes":
                    Myclass obj = new Myclass();
                    obj.PublicateNote();
                    program.add(obj);
                    if (program.size() > 1)
                    {
                        System.out.printf("Time from the last change: ");
                        System.out.println((obj.Date.getTime() - program.get(program.size()-
2).Date.getTime())/1000 + "c");
                    }
                    break;
                case  "no":
                    System. exit(0);
                    break;

                default:
                    System.out.println("Check your input!");
                    break;
            }        }
    }}
}
```

Problema 8

8.File, astfel încât să se poată deduce mărimea obiectului după modificare.

```
package Lab2;

import java.io.*;
import java.util.Scanner;

class Person{
    String name;

    Person(String n){
        this.name = n;
    }
}
class Student extends Person{
    String group = "IA-191";

    Student(String n) {
        super(n);
    }
    public String getGroup(){
        return group;
    }
    public String getName(){
        return this.name;
    }
}
public class SecondLab {
    /*
    * Crearea fisierului pentru a putea deduce marimea dupa modificare
    * */
    public void startLab(){
        File f = new File("D:\\example.txt");

        System.out.println("Marimea fisierului pana la modificare:");
        System.out.println(getFileSizeBytes(f));


        System.out.println("Scimbari in fisier");
        Scanner sc = new Scanner(System.in);


        System.out.println("Introduceti numele");
        String name = sc.nextLine();


        Student me = new Student(name);


        try {
            FileWriter writer = new FileWriter("D:\\example.txt", true);
            BufferedWriter bw = new BufferedWriter(writer);
            writer.write(me.getName() + " ");
            writer.write(me.getGroup()+ " ");
            bw.newLine();
            writer.flush();
        } catch (IOException e) {
            e.printStackTrace();
        }


        if (f.exists())
        {
            System.out.println("Marimea fisierului dupa modificare:");
            System.out.println(getFileSizeBytes(f));
        }else
            System.out.println("File not found");

    }

    private static String getFileSizeBytes(File file) {
        return file.length() + " bytes";
    }
}


public class Main {

    public static void main(String[] args) {

    SecondLab lab2 = new SecondLab();
    lab2.startLab();

    }
}
```

Problema 9
Stack, astfel încât să se poată deduce timpul in minute a ultimei sesiuni a lucrului cu stiva.

```
class useTime{
    private int hour, minute, second;
    void setTime(){
        hour = (int) (((System.currentTimeMillis() / 1000) / 60) / 60) % 24 + 2;
        minute = (int) ((System.currentTimeMillis() / 1000) / 60) % 60;
        second = (int) (System.currentTimeMillis() / 1000) % 60;
    }
    void setTime(int h){
        hour = h;
        minute = (int) ((System.currentTimeMillis() / 1000) / 60) % 60;
        second = (int) (System.currentTimeMillis() / 1000) % 60;
    }
    void setTime(int h, int m){
        hour = h;
        minute = m;
        second = (int) (System.currentTimeMillis() / 1000) % 60;
    }
    void setTime(int h, int m, int s){
        hour = h;
        minute = m;
        second = s;
    }

    void getTime(){
        System.out.println(hour+":"+minute+":"+second);
    }
}

class Stack extends useTime {
    private void createStack(){
        /*
        codul pentru crearea stivei
         */
        setTime();
    }

    private void showStack(){
        /*
        codul pentru afisarea stivei
         */

        System.out.print("Obiectul a fost creat la: ");
        getTime();
    }
public static void main(String[] args){
        Stack stiva = new Stack();
        Stack stiva1 = new Stack();
        Stack stiva2 = new Stack();
        Stack stiva3 = new Stack();

        stiva.createStack();
        stiva.showStack();

        stiva1.setTime(11);
        stiva1.showStack();

        stiva2.setTime(11, 22);
        stiva2.showStack();

        stiva3.setTime(10, 54, 9);
        stiva3.showStack();

    }
}
```