

**MINISTERUL EDUCAȚIEI ȘI
CERCETĂRII AL REPUBLICII
MOLDOVA**

Centrul de Excelență în Informatică și Tehnologii Informaționale

Raport

Practica de Instruire

**Specialitatea: Programare și analiza produselor
program Tema: Biletul 12**

Elev: Cozma Vasile

Grupa: P-2221

Conducătorul Practicii : Frunza Olga

Chișinău 2024

Cuprins

Cuprins	2
Sarcina	3
Listingul programului	5
App.java	5
Controller.java	8
Muncitori.java	10
Manageri.java	12
Users.java	13
Pachetele	15
Rezultatele execuției codului	17
Concluzie	22
Bibliografie	23
Site-uri Web:	23

Sarcina

Sarcina 2: Scrieți un program Java care modelează următoarea activitate:

Creați un produs program pentru evidența activității a unei companii de consultanță. Compania este împărțită în subdiviziuni. Fiecare subdiviziune are o denumire, adresă, telefon, un singur angajat.

În companie angajații pot face mici achiziții de materiale consumabile pentru birou pentru necesitățile companiei (hârtie, caiete, pixuri, creioane și alte produse de birotică). Bonul de plată pentru achizițiile făcute este prezentat în contabilitate. Achiziții pot fi făcute conform categoriilor de cheltuieli prevăzute în companie. Categoriile de cheltuieli sunt caracterizate prin denumire, descriere, suma maximă pe lună.

Pentru fiecare achiziție angajatul face un raport în care indică categoria de cheltuieli, data, suma, subdiviziunea. Produsul program trebuie să asigure evidența acestor achiziții atât conform subdiviziunilor, cât și conform angajaților, care au făcut achizițiile. Atât pentru subdiviziuni cât și pentru angajați sunt stabilite sume maxime lunare, care pot fi cheltuite în aceste scopuri.

Cerințe față de sarcină: Implementați concepte ale POO

- Creați clase având ca metode (funcții) citirea și afișarea fiecărui câmp (set-teri și get-teri), precum și o metodă care calculează o valoare nouă a unui sau a mai multor câmpuri (ca exemplu, de convertit prețul din lei în EURO sau de calculat durata deplasării). Adăugați cel puțin trei tipuri de constructori (unul trebuie să fie fără parametri, altul- cu toți parametrii indicați, iar al treilea- cu doar câțiva parametri indicați, în dependență de temă, în ultimul caz unele câmpuri ar putea avea valori egale sau careva valori standard).
- Clase care moștenesc câmpurile și metodele clasei părinte vor avea proprietăți proprii, creând și constructorii respectivi (care să corespundă constructorilor clasei de bază). Folosiți relațiile de asociere și agregare între obiecte (acolo unde este cazul).
- Creați o clasă abstractă și o interfață, care are cel puțin o metodă abstractă, apoi le utilizează în procesul de moștenire și o realizează pe deplin. Fiecare clasă trebuie să conțină constructori supraîncărcați, metoda *toString()* supradefinită
- Prelucrați diferite tipuri de excepție (atunci când avem împărțire la zero, radical din număr negativ, când încercăm să accesăm un element inexistent al unui tablou, atunci când în loc de număr utilizatorul introduce litere, atunci când încercăm să apelăm o metodă a unui obiect care are referință nulă etc.) O metodă care generează excepție proprie.
- Înscrisți obiectele într-o listă, minim 20 de înregistrări. Se va utiliza lista și la afișarea pe ecran.
- Programul să conțină un meniu minimalizat.
- Adăugați clase și / sau membri sau metode la clasele dacă se consideră necesar.

Listingul programului

App.java

```
import java.io.File;

import java.io.FileNotFoundException;

import java.io.FileReader;

import java.util.Scanner;


import javafx.application.Application;

import javafx.scene.*;

import javafx.scene.image.Image;

import javafx.stage.Stage;

import javafx.fxml.*;


// https://www.youtube.com/watch?v=TVLq9iSpuMs


public class App extends Application{

    public static Parent register,login;

    @Override

    public void start(Stage primaryStage) throws Exception {

        Scanner in = new Scanner(new FileReader(new
File("src\\Database\\accountsInfo.txt")));

        while(in.hasNext()){

            Controller.acc.add(new
Users(in.next(),in.next(),in.next(),in.next(),in.next(),in.next(),in.ne
xt(),Integer.parseInt(in.next()),Integer.parseInt(in.next())));

        }

        Scanner bin = new Scanner(new FileReader(new
File("src\\Database\\buget.txt")));

        Users.setBuget(bin.nextInt());

    }

}
```

```
        bin.close();

        in.close();

        try {

            primaryStage.getIcons().add(new
Image("Database/icon.png"));

        } catch (Exception e) {}

    }

    Scanner dqin = new Scanner(new FileReader(new
File("src\\Database\\deleteQueue.txt")));

    while(dqin.hasNext()){

        Controller.deleteQueue.add(dqin.next());

    }

    dqin.close();

    Scanner min = new Scanner(new FileReader(new
File("src\\Database\\managersInfo.txt")));

    while(min.hasNext()){

        Controller.manAcc.add(new Manageri(min.next(),min.next(),
min.next(), min.next(), min.next(),min.next()));

    }

    min.close();

    primaryStage.setTitle("Consulting Company");
```

```
primaryStage.setResizable(false);

    Scene scene = new
Scene(FXMLLoader.load(getClass().getResource("Interface\\login.fxml")))
;

scene.getStylesheets().add(this.getClass().getResource("Style\\login.css").toExternalForm());

    primaryStage.setScene(scene);

    primaryStage.show();

}

    public static void main(String[] args) throws
Exception,FileNotFoundException {

        launch(args);

    }

}
```

Controller.java

```
public class Controller implements Initializable{

    protected static String[] sub = {"Consultanță în cetățenii străine", "Consultanță privind strategii", "Consultanță în afaceri", "Consultanță financiară", "Consultanță IT", "Consultanță în management", "Consultanță în vânzări", "Consultanță în marketing", "Consultanță de brand", "Servicii de consultanță în imobiliare"};

    @FXML

    private TextField loginRegister, passwordRegister, loginLogin, passwordLogin, userName, firstName, lastName, reportName, reportPrice, buget, settingsTextField1, settingsTextField2, settingsTextField3, deactivationConfirmation;

    @FXML

    private Label registerMessage, registerMessage1, loginMessage, myLabel, infoLabel, activationMessage, notnullMessage, USER, fileInputMessage, reportSuccess, reportInfo, reportInfo1, reportInfo2, appControlPanelInfo, appControlPanelInfo1, budgetLabel, appInfo, appBuget, settingsInfo, settingsInfo1, changeError, label, label1, label2, labelSubdivision, accountDeactivationMessage, confirmationMessageForDez, notnullDeactivation, deleteMessage, priceError, notnullError;

    @FXML

    private ListView<String> myListView = new ListView<>();

    @FXML

    private ListView<String> reportsList = new ListView<>();

    @FXML

    private ListView<String> reports = new ListView<>();

    @FXML

    private ListView<String> deleteCon;

    @FXML

    private ChoiceBox<String> selectSubdivision = new ChoiceBox<>();

    @FXML

    private Button
```

```
newReport,changeConfirm,userAccountDesactivation,button1,button2,delete  
Acc,deleteAcc1,button3;
```

```
@FXML
```

```
private ImageView imgview,reportPhoto,companyLogo;
```

```
@FXML
```

```
private TextArea reportDesc;
```

```
protected static ArrayList<Users> acc = new ArrayList<>();
```

```
protected static ArrayList<String> deleteQueue = new ArrayList<>();
```

```
protected static ArrayList<Manageri> manAcc = new ArrayList<>();
```

```
private static int userIndex;
```

```
int k = 0;
```

```
..... *
```

```
}
```


Muncitori.java

```
public abstract class Muncitori{

    public String username,nume,prenume,email,parola,creationDate;

    public int raport;

    public static int buget;


    public static int getBuget() {

        return buget;

    }

    public static void setBuget(int buget) {

        Muncitori.buget = buget;

    }

    public int getRaport() {

        return raport;

    }

    public String getParola() {

        return parola;

    }

    public String getEmail() {

        return email;

    }

    public String getNume() {

        return nume;

    }

    public String getPrenume() {

        return prenume;

    }

}
```

```
public String getUsername() {  
  
    return username;  
  
}  
  
public String getCreationDate() {  
  
    return creationDate;  
  
}  
  
public void setEmail(String email) {  
  
    this.email = email;  
  
}  
  
public void setNume(String nume) {  
  
    this.nume = nume;  
  
}  
  
public void setPrenume(String prenume) {  
  
    this.prenume = prenume;  
  
}  
  
public void setUsername(String username) {  
  
    this.username = username;  
  
}  
  
public void setCreationDate(String creationDate) {  
  
    this.creationDate = creationDate;  
  
}  
  
public void setParola(String parola) {  
  
    this.parola = parola;  
  
}  
  
public void setRaport(int raport) {  
  
    this.raport = raport;  
  
}  
  
}
```

Manageri.java

```
import java.text.SimpleDateFormat;

import java.util.Date;

public class Manageri extends Muncitori{

    Manageri(String email,String username,String nume,String
prenume,String parola){

        this.username = username;

        this.nume = nume;

        this.prenume = prenume;

        this.email = email;

        this.parola = parola;

        creationDate = new
SimpleDateFormat("dd-MM-yyyy|HH:mm").format(new Date());

    }

    Manageri(String email,String username,String nume,String prenume,
String parola, String creationDate){

        this.username = username;

        this.nume = nume;

        this.prenume = prenume;

        this.email = email;

        this.parola = parola;

        this.creationDate = creationDate;

    }

}
```

Users.java

```
import java.text.SimpleDateFormat;

import java.util.Date;

public class Users extends Muncitori{

    private boolean activation;

    private int subdivision;

    Users(String email,String username,String nume,String
prenume,String parola,int subdivision){

        this.username = username;

        this.nume = nume;

        this.prenume = prenome;

        this.email = email;

        this.parola = parola;

        this.activation = false;

        this.subdivision = subdivision;

        creationDate = new
SimpleDateFormat("dd-MM-yyyy|HH:mm").format(new Date());
    }

    Users(String email,String username,String nume,String prenome,
String parola, String creationDate, String activationStatus, int
raport, int subdivision){

        this.username = username;

        this.nume = nume;

        this.prenume = prenome;

        this.email = email;

        this.parola = parola;
```

```
        this.creationDate = creationDate;

        this.subdivision = subdivision;

        if(activationStatus.equals("true"))

            this.activation = true;

        else

            this.activation = false;

        this.raport = raport;
    }


    public int getSubdivision() {

        return subdivision;

    }

    public void setSubdivision(int subdivision) {

        this.subdivision = subdivision;

    }

    public boolean getActivation(){

        return activation;

    }

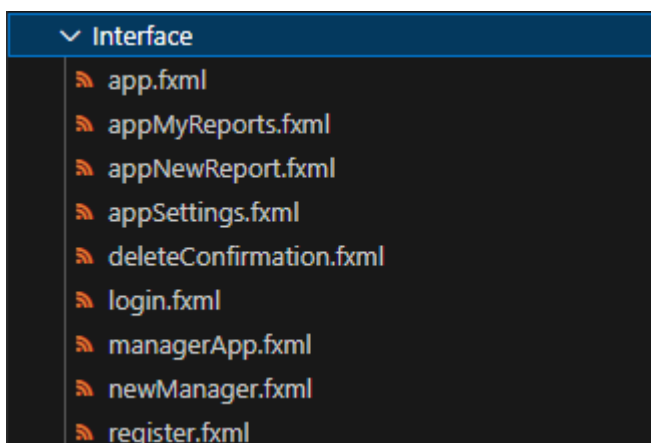
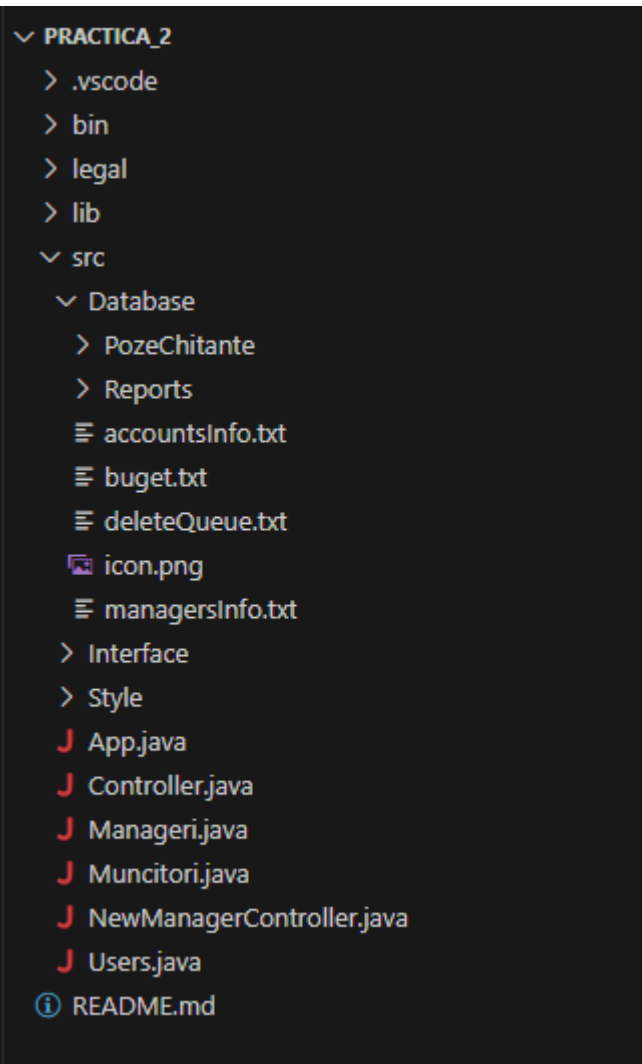
    public void setActivation(boolean activation) {

        this.activation = activation;

    }

}
```

Pachetele










▼ Style

- # app.css
- # appMyReports.css
- # appNewReport.css
- # appSettings.css
- # deleteConfirmation.css
- # dezactivationMessage.css
- # login.css
- # managerApp.css
- # newManager.css
- # register.css


▼ Reports

- ≡ raport---1---0.txt
- ≡ raport---1---1.txt
- ≡ raport---Alexandra_Alex---0.txt
- ≡ raport---Alexandra_Alex---1.txt
- ≡ raport---Alina_Ally---0.txt
- ≡ raport---Andrei.AND---0.txt
- ≡ raport---Gabriela_Gabi---0.txt
- ≡ raport---Vasyle_---0.txt

▼ PozeChitante

-  raport---1---0.png
-  raport---1---1.png
-  raport---Alexandra_Alex---0.png
-  raport---Alexandra_Alex---1.png
-  raport---Alina_Ally---0.png
-  raport---Andrei.AND---0.png
-  raport---Gabriela_Gabi---0.png
-  raport---Vasyle_---0.png

Rezultatele execuției codului




Logare

Creaza cont nou

Login

Login Page



Creare cont

User name

Nume

Prenume

Subdiviziune ▼

Register

Înapoi

Alina_Ally

1

Stergere Cont

Respingere

Raport nou

Rapoartele mele

Setări

Delogare

User name:

1

1

Email:

1

Nume:

1

Prenume:

1

Data creării contului:

11-06-2024|19:12

Numarul de rapoarte:

2



Bugetul companiei : 200 €

Înapoi

Nume Raport

Consum €

Descrizione

Introduceți poza chitanței

Confirm

Înapoi

Raportul numărul 1

Raportul numărul 2

Raportul numărul 2

Consumabile

85 €

Descriere: Achiziții pentru nevoi de birou.

Detalii achiziție:

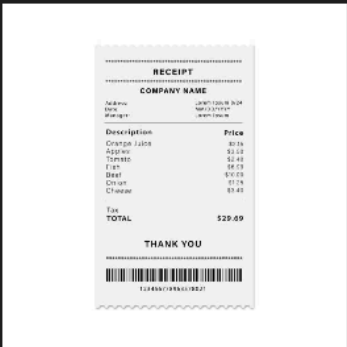
Categorie cheltuieli: Consumabile birou

Data: 12 iunie 2024

Detalii achiziție:

Categorie cheltuieli: Consumabile birou

Data: 12 iunie 2024



[Înapoi](#)

User name:	1
Email:	1
Nume:	1
Prenume:	1
Data crearii contului:	11-06-2024 19:12
Numarul de rapoarte:	2

[Schimbă username-ul](#)

[Schimbă parola](#)

[Schimbă email-ul](#)

[Dezactivează contul](#)

[Șterge contul](#)

Concluzie

În concluzie, sarcina dată în practica de instruire a fost extrem de valoroasă pentru mine în ceea ce privește înțelegerea programării orientate pe obiecte și a aplicațiilor vizuale GUI. Cursul oferit de Iucosoft a reprezentat o componentă esențială în dezvoltarea cunoștințelor mele, permițându-mi să învăț și să aplic metodele necesare pentru crearea unei interfețe grafice intuitive și eficiente. Această experiență practică m-a ajutat să îmbunătățesc semnificativ abilitățile în domeniul programării orientate pe obiecte și să capăt încredere în utilizarea conceptelor teoretice în practică. Astfel, consider că participarea la această practică de instruire a fost fundamentală pentru consolidarea cunoștințelor mele și pentru pregătirea mea pentru viitoarele proiecte în domeniul IT.

În plus, sunt încântat că am primit certificatul de participare de la Iucosoft pentru această practică, ceea ce confirmă și validează eforturile mele în învățarea și aplicarea conceptelor într-un mediu practic și profesional. Acest certificat reprezintă o recunoaștere a angajamentului meu și mă motivează să continui să explorez și să dezvolt abilități în programarea orientată pe obiecte și în crearea de interfețe grafice.



Bibliografie

https://github.com/Vasyle-Czm/practica_2.git - Link către tot proiectul scris în Visual Studio Code IDE

Site-uri Web:

<https://docs.oracle.com/javafx/2/api/>

https://youtube.com/playlist?list=PLZPZq0r_RZOM-8vJA3NQFZB7JroDcMwev&si=4LcTnVR2pxYMHU41