Ministerul Educaţiei al Republicii Moldova

Universitatea de Stat din Moldova

Facultatea de Matematică şi Informatică

Departamentul “Informatica »

**Dare de seama la „Java”**

Lucrare de laborator № 1

Tema : „*Realizarea claselor în Java*”

Varianta: *№* *19*

Clasa: *„Prof”*

Efectuat: studentul (-a) grupei IA 2402

numele, prenumele Zemlinschi Dragos

Verificat: lect.univers., magistru in inform.

*Anghelova Irina .*

* Chişinău 2025 -

**Formularea problemei:**

De creat clasa “**Prof**” cu câteva câmpuri.

Câmpurile obligatorii:

- 2 cîmpuri de tip șir de caractere, care păstrează numele profesorului și denumirea obiectului condus de el.

- cîmp fracțional (double), care păstrează răutăciunea profesorului (procentul restanțierilor lui) (de la 0 la 100)

- cîmp fracțional (float), care păstrează intensitatea maximă în decibeli a țipătului profesorului enervat.

- cîmp întreg (short), care păstrează numărul de grupe unde predă profesorul.

- vector dinamic (long), care păstrează informația despre numărul studenților în fiecare grupă.

- cîmp intreg static, în care se va păstra numărul profesorilor creați.

Cîmpuri neobligatorii: stagiul, sexul, media notelor puse de profesor s.a.m.d

De creat trei tipuri  de constructori pentru această clasă (cu alocare dinamică a memoriei):

- constructor standard (implicit - fără parametri)

- câteva constructori cu parametri

- constructor de copiere

De creat metodele pentru acces şi de modificare a campurilor clasei (set() si get()).

* De creat metoda pentru a afişa la ecran toată informaţia despre profesor (toate câmpurile clasei).
* De descris funcția, care returnează numărul total al sărmanilor, care au nimerit la acest prof (suma elementelor vectorului).
* De descries metoda, care completeaza toate campurile obiectului cu valori citite de la tastatura.
* De descries metoda, care completeaza toate campurile obiectului cu valori aleatoare.
* De descris funcția, care compară doi profesori (pe cel curent și cel primit ca parametru) dupa ”popularitate” (după numărul total de stundenți, care învață la el). De afișat numele și obiectul celui mai popular prof dintre cei comparați.
* De descris o funcție statică  ”stresul studentului”, care va primi ca parametri doi profesori, și care calculează cu cît primul profesor influnențează mai bine (sau mai rău) starea de stres a studentului, luînd în considerație răutăciunea și strigăciunea lui în decibeli. Rezultatul comparării de afișat la ecran.

În funcția **main()** de creat nişte profi, folosind toți constructorii descriși. Dupa crearea fiecărui exemplar nou al clasei ”**Prof**”, de afișat la ecran valorile cîmpurilor lui printr-o metodă a clasei de afișare. De creat vector dinamic al profilor – ”univer”. Vectorul să fie iniţializat pe bucăţi, folosind constructori diferiţi. În ciclu de a afişa toată informaţia despre fiecare ciudat din univer. De comparat cîțiva profi după popularitate și dupa nivelul de stres al studenților. De găsit și de afișat numele tuturor profilor care duc același obiect ca și primul prof creat. Ultimul rînd afişat să conţină numărul profilor creați, folosind variabila statică a clasei.

**Pentru nota 10.** Adaugator pentru tot ce este descris mai sus:

- un constructor care primeşte ca parametru denumirea fişierului textual (String), de unde se incarcă valori pentru câmpurile obiectului creat.

- o funcţie care înscrie toate cîmpurile clasei în fişier, numele fisierului se indică ca parametru la intrare.

- În funcţia main pentru toate obiectele create de salvat  datele în fişiere, denumirile cărora se preiau de la numele obiectelor.

După ce totul a fost terminat fără greşele — de spus “**URRRRRRAAAAAAA ! ! !**” şi de prezentat primul laborator profesorului. :)

**Copy-paste codului sursa**

    public class Prof {

        private String nume;

        private String obiect;

        private double rautaciune;

        private float galagie;

        private short nrGrupe;

        private long nrStudenti[];

        private static int profesori = 0;

/////////////////////////……….. ////////////////////////

public Prof(){ //constructor implicit

            this.nume = "Profesor";

            this.obiect = "Nu preda la moment!";

            this.rautaciune = 0d;

            this.galagie = 60f;

            this.nrGrupe = 0;

            this.nrStudenti = new long[0];

            profesori++;

        }

public Prof(String nume, double rautaciune, float galagie){ //constructor cu 3 parametri

            this();

            this.nume = nume;

            this.rautaciune = rautaciune;

            this.galagie = galagie;

        }

public Prof(String nume, String obiect){ //constructor cu 2 parametiri

            this();

            this.nume = nume;

            this.obiect = obiect;

        }

//constructor pe deplin

        public Prof(String nume, String obiect, double rautaciune, float galagie, short nrGrupe, long nrStudenti){

            this.nume = nume;

            this.obiect = obiect;

            this.rautaciune = rautaciune;

            this.galagie = galagie;

            this.nrGrupe = nrGrupe;

            this.nrStudenti = new long[nrGrupe];

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

                this.nrStudenti[i] = nrStudenti;

            }

            profesori++;

        }

public Prof(Prof p){ //constructor de copiere

            this.nume = p.nume;

            this.obiect = p.obiect;

            this.rautaciune = p.rautaciune;

            this.galagie = p.galagie;

            this.nrGrupe = p.nrGrupe;

            this.nrStudenti = new long[p.nrGrupe];

            for(int i=0; i<p.nrGrupe; i++){

                this.nrStudenti[i] = p.nrStudenti[i];

            }

            profesori++;

        }

        public Prof(String fileName) { //constructor cu citire din fisier

            try {

                Scanner sc = new Scanner(new File(fileName));

                this.nume = sc.nextLine();

                this.obiect = sc.nextLine();

                this.rautaciune = (double) sc.nextInt();

                this.galagie = (float) sc.nextFloat();

                this.nrGrupe = (short) sc.nextShort();

                this.nrStudenti = new long[this.nrGrupe];

                sc.nextLine(); // consumă linia rămasă

                String stringArr[] = sc.nextLine().split(" ");

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

                    this.nrStudenti[i] = (long) Integer.parseInt(stringArr[i]);

                }

                profesori++;

                sc.close();

            } catch (java.io.FileNotFoundException e) {

                System.out.println("Fisierul nu a fost gasit: " + fileName);

                // Set default values if file not found

                this.nume = "Profesor";

                this.obiect = "Nu preda la moment!";

                this.rautaciune = 0d;

                this.galagie = 60f;

                this.nrGrupe = 0;

                this.nrStudenti = new long[0];

                profesori++;

            }

        }

////////////////////////////////////// metodele de acces la cîmpurile clasei (set\_\*) ////////////////////////////////////////

public void setNume(String nume){

            this.nume = nume;

        }

        public void setObiect(String obiect){

            this.obiect = obiect;

        }

        public void setRautaciune(double rautaciune){

            this.rautaciune = rautaciune;

        }

        public void setGalagie(float galagie){

            this.galagie = galagie;

        }

        public void setNrGrupe(short nrGrupe){

            this.nrGrupe = nrGrupe;

        }

        public void setNrStudenti(long nrStudenti[]){

            this.nrStudenti = new long[nrGrupe];

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

                this.nrStudenti[i] = nrStudenti[i];

            }

        }

////////////////////////////////////// metodele de acces la cîmpurile clasei (get\_\*) ////////////////////////////////////////

        public String getNume(){

            return this.nume;

        }

        public String getObiect(){

            return this.obiect;

        }

        public double getRautaciune(){

            return this.rautaciune;

        }

        public float getGalagie(){

            return this.galagie;

        }

        public short getNrGrupe(){

            return this.nrGrupe;

        }

        public long[] getNrStudenti(){

            return this.nrStudenti;

        }

        public static int getProfesori(){

            return profesori;

        }

////////////////////////////////////////////////////// metodele ////////////////////////////////////////////////////////

        public void afisare(){ //afisarea la consola a unu obiect

            System.out.println("Nume: " + nume);

            System.out.println("Obiect: " + obiect);

            System.out.println("Rautaciune: " + rautaciune);

            System.out.println("Galagie: " + galagie);

            System.out.println("Numar grupe: " + nrGrupe);

            System.out.print("Numar studenti pe grupe: ");

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

                System.out.print(nrStudenti[i] + " ");

            }

            System.out.println();

        }

        public int sumaSarmanilor(){ //calcularea sumei elementelor vectorului

            int suma = 0;

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

                suma += nrStudenti[i];

            }

            return suma;

        }

        public void citireProf(){ // Citirea de la tastatura a cimpurilor din obiect

            java.util.Scanner scanner = new java.util.Scanner(System.in); // Crearea instantei de scanner

            System.out.print("Nume: ");

            this.nume = scanner.nextLine();

            System.out.print("Obiect: ");

            this.obiect = scanner.nextLine();

            System.out.print("Rautaciune: ");

            this.rautaciune = scanner.nextDouble();

            System.out.print("Galagie: ");

            this.galagie = scanner.nextFloat();

            System.out.print("Numar grupe: ");

            this.nrGrupe = scanner.nextShort();

            this.nrStudenti = new long[nrGrupe];

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

                System.out.print("Numar studenti grupa " + (i+1) + ": ");

                this.nrStudenti[i] = scanner.nextLong();

            }

        }

        public void randomProf(){ // Completarea campurilor cu valori aleatorii

            Random r = new Random(); // Utilizarea clasei Random

            String numePosibile[] = {"Cristian","Eugen","Nicolae","Catalin","Stanescu","Maria","Vlad"}; // Array de nume

            String obiectePosibile[] = {"Java","Fundamentele","C++","Antreprenoriat","Rust","Criptare"}; // Array de obiecte posibile

            this.nume = numePosibile[r.nextInt(0,6)];

            this.obiect = obiectePosibile[r.nextInt(0,5)];

            this.rautaciune = r.nextInt(0,100);

            this.galagie = r.nextInt(40,120);

            this.nrGrupe = (short) r.nextInt(0,12);

            this.nrStudenti = new long[nrGrupe];

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

                this.nrStudenti[i] = r.nextLong(1,35);

            }

        }

        public void popularProf(Prof b){

            int popularitateProfA = 0;

            int popularitateProfB = 0;

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

                popularitateProfA += this.nrStudenti[i];

            }

            for(int i = 0; i < b.nrGrupe;i++){

                popularitateProfB += b.nrStudenti[i];

            }

            if(popularitateProfA>=popularitateProfB){

                System.out.println("Profesorul cel mai popular este: " + this.nume + ",si preda " + this.obiect);

            }else {

                System.out.println("Profesorul cel mai popular este: " + b.nume + ",si preda " + b.obiect);

            }

        }

        public static void stressCalc(Prof a,Prof b){

            //how does it affect student

            float stressA  = (float) ((a.rautaciune \* 0.7) + (a.galagie \* 0.3));

            float stressB  = (float) ((b.rautaciune \* 0.7) + (b.galagie \* 0.3));

            if(stressA==stressB){

                System.out.println("Ambii profesori sunt la fel de stresanti, cu un indice de stress de " + stressA);

            }else if(stressA>stressB){

                System.out.println("Profesorul " + a.nume + " este mai stresant decit " + b.nume + ", cu un indice de stress de " + stressA + " fata de " + stressB);

            }else{

                System.out.println("Profesorul " + a.nume + " este mai putin stresant decit " + b.nume + ", cu un indice de stress de " + stressA + " fata de " + stressB);

            }

            //comparation of the teachers

            boolean rautaciuneTop = a.rautaciune<=b.rautaciune;

            boolean galagieTop = a.galagie <= b.galagie;

            if (rautaciuneTop ^ galagieTop) {

                if(rautaciuneTop){

                    System.out.println("Amindoi profi sunt stresanti, cel mai rautacios fiind " + b.nume + " cu o rautaciune de " + b.rautaciune + ". Si cel mai galagios fiind " + a.nume + " cu o galagie de " + a.galagie);

                }else{

                    System.out.println("Amindoi profi sunt stresanti, cel mai rautacios fiind " + a.nume + " cu o rautaciune de " + a.rautaciune + ". Si cel mai galagios fiind " + b.nume + " cu o galagie de " + b.galagie);

                }

            }else{

                if (rautaciuneTop) {

                    System.out.println("Profesorul cel mai stressant este " + b.nume + " cu o rautaciune de " + b.rautaciune + " si o galagie de " + b.galagie);

                }else{

                    System.out.println("Profesorul cel mai stressant este " + a.nume + " cu o rautaciune de " + a.rautaciune + " si o galagie de " + a.galagie);

                }

            }

        }

        public void inscrieProf(String fileName,Prof prof) {

            try {

                FileWriter fw = new FileWriter(fileName + ".txt");

                fw.write(prof.nume + "\n");

                fw.write(prof.obiect + "\n");

                fw.write(prof.rautaciune + "\n");

                fw.write(prof.galagie + "\n");

                fw.write(prof.nrGrupe + "\n");

                for(int i=0; i<prof.nrGrupe; i++){

                    fw.write(prof.nrStudenti[i] + " ");

                }

                fw.close();

            } catch (IOException e) { // Catch error si scrierea ei in terminal

                System.out.println("Eroare la scrierea in fisier: " + e.getMessage());

            }

        }

    }

////////////////////////////////////////

…………………

//////////////////////////////////////////// funcția principală **main** //////////////////////////////////////////////////

public static void main(String[] args){

Prof prof1 = new Prof(); // Apelarea constructorului implcit

Prof prof2 = new Prof("Popescu", 80.5, 90.0f); // Apelarea constructorului cu 3 parametri

Prof prof3 = new Prof("Ionescu", "Matematica"); // Apelarea constructorului cu 2 parametri

Prof prof4 = new Prof("Georgescu", "Fizica", 60.0, 70.0f, (short)2, 25L); // Apelarea constructorului pe deplin

Prof prof5 = new Prof(prof4); // Apelarea constructorului de copiere

System.out.println("--- Prof1 (implicit) ---"); // Afisarea obiectelor create

prof1.afisare();

System.out.println("--- Prof2 (partial: nume, rautaciune, galagie) ---");

prof2.afisare();

System.out.println("--- Prof3 (partial: nume, obiect) ---");

prof3.afisare();

System.out.println("--- Prof4 (full) ---");

prof4.afisare();

System.out.println("--- Prof5 (copiere prof4) ---");

prof5.afisare();

// Vector dinamic de Profi cu array simplu

int cap = profesori; // Capacitatea initiala

Prof[] univer = new Prof[cap];

int n = 0;

univer[n++] = prof1; // Adaugare Profi

univer[n++] = prof2;

univer[n++] = prof3;

univer[n++] = prof4;

univer[n++] = prof5;

Prof[] deAdaugat = new Prof[] { // Adaugam inca 3 Profi

new Prof("profText.txt"),

new Prof("Dumitru", 40.0, 60.0f),

new Prof("Vasilescu", "Chimie")

};

for (Prof p : deAdaugat) {

if (n == cap) {

cap = cap \* 2; //extindem array-ul

Prof[] nou = new Prof[cap];

for (int i = 0; i < n; i++) nou[i] = univer[i];

univer = nou;

}

univer[n++] = p;

}

System.out.println("\n--- Toți profii din univer ---"); // Afișare toată informația despre fiecare prof din univer

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

System.out.println("\nProf " + (i+1) + ":");

univer[i].afisare();

}

System.out.println("\n--- Comparare popularitate ---"); // Comparare câțiva profi după popularitate și stres

univer[0].popularProf(univer[3]);

univer[1].popularProf(univer[2]);

System.out.println("\n--- Comparare stres studenti ---");

Prof.stressCalc(univer[0], univer[3]);

Prof.stressCalc(univer[1], univer[2]);

String obiectPrimul = univer[0].getObiect(); //Găsire și afișare nume profi care predau același obiect ca primul prof

System.out.println("\n--- Profii care predau acelasi obiect ca primul prof (\"" + obiectPrimul + "\") ---");

boolean found = false;

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

Prof p = univer[i];

if (p.getObiect() != null && p.getObiect().equals(obiectPrimul)) {

System.out.println(p.getNume());

found = true;

}

}

if (!found) {

System.out.println("Niciun prof nu predă acelasi obiect ca primul prof.");

}

System.out.println("\nNumar total de profi creati: " + Prof.getProfesori()); //Afișare număr total de profi creați

inscrieProf("profInscrisManual.txt", univer[0]); // Inscrierea profesorului în fișier

File folder = new File(".");

File[] files = folder.listFiles((dir, name) -> name.endsWith(".txt") && !name.equals("profText.txt"));

if (files != null) { // Curatarea fisierelor de la rularea trecuta

System.out.println("\n--- Ștergerea fișierelor .txt existente (excluzând profText.txt) ---");

for(int i = 0; i<files.length; i++){

System.out.println("Șterg: " + files[i].getName());

files[i].delete();

}

}

for(int i = 0; i< univer.length; i++){ // Scriere in fisiere a tuturor Profilor

if(univer[i] != null){

try {

FileWriter fw = new FileWriter(univer[i].getNume() + ".txt"); // Utilizam clasa FileWriter

fw.write(univer[i].getNume() + "\n");

fw.write(univer[i].getObiect() + "\n");

fw.write(univer[i].getRautaciune() + "\n");

fw.write(univer[i].getGalagie() + "\n");

fw.write(univer[i].getNrGrupe() + "\n");

for(int j = 0; j < univer[i].getNrGrupe(); j++){

fw.write(univer[i].getNrStudenti()[j] + " ");

}

fw.close(); //pentru a finisa scrierea in fisier

} catch (Exception e) {

System.out.println("error");

}

}else{ // In caz ca elemntul din arrayul care in parcurgem este gol/ null

System.out.println("Found a null entry, skipping.");

}

}

}

**Copy-paste rezultatului din consola**

--- Prof1 (implicit) ---

Nume: Profesor

Obiect: Nu preda la moment!

Rautaciune: 0.0

Galagie: 60.0

Numar grupe: 0

Numar studenti pe grupe:

--- Prof2 (partial: nume, rautaciune, galagie) ---

Nume: Popescu

Obiect: Nu preda la moment!

Rautaciune: 80.5

Galagie: 90.0

Numar grupe: 0

Numar studenti pe grupe:

--- Prof3 (partial: nume, obiect) ---

Nume: Ionescu

Obiect: Matematica

Rautaciune: 0.0

Galagie: 60.0

Numar grupe: 0

Numar studenti pe grupe:

--- Prof4 (full) ---

Nume: Georgescu

Obiect: Fizica

Rautaciune: 60.0

Galagie: 70.0

Numar grupe: 2

Numar studenti pe grupe: 25 25

--- Prof5 (copiere prof4) ---

Nume: Georgescu

Obiect: Fizica

Rautaciune: 60.0

Galagie: 70.0

Numar grupe: 2

Numar studenti pe grupe: 25 25

--- To?i profii din univer ---

Prof 1:

Nume: Profesor

Obiect: Nu preda la moment!

Rautaciune: 0.0

Galagie: 60.0

Numar grupe: 0

Numar studenti pe grupe:

Prof 2:

Nume: Popescu

Obiect: Nu preda la moment!

Rautaciune: 80.5

Galagie: 90.0

Numar grupe: 0

Numar studenti pe grupe:

Prof 3:

Nume: Ionescu

Obiect: Matematica

Rautaciune: 0.0

Galagie: 60.0

Numar grupe: 0

Numar studenti pe grupe:

Prof 4:

Nume: Georgescu

Obiect: Fizica

Rautaciune: 60.0

Galagie: 70.0

Numar grupe: 2

Numar studenti pe grupe: 25 25

Prof 5:

Nume: Georgescu

Obiect: Fizica

Rautaciune: 60.0

Galagie: 70.0

Numar grupe: 2

Numar studenti pe grupe: 25 25

Prof 6:

Nume: Alexandru

Obiect: Sisteme de Operare

Rautaciune: 25.0

Galagie: 60.0

Numar grupe: 3

Numar studenti pe grupe: 20 12 30

Prof 7:

Nume: Dumitru

Obiect: Nu preda la moment!

Rautaciune: 40.0

Galagie: 60.0

Numar grupe: 0

Numar studenti pe grupe:

Prof 8:

Nume: Vasilescu

Obiect: Chimie

Rautaciune: 0.0

Galagie: 60.0

Numar grupe: 0

Numar studenti pe grupe:

--- Comparare popularitate ---

Profesorul cel mai popular este: Georgescu,si preda Fizica

Profesorul cel mai popular este: Popescu,si preda Nu preda la moment!

--- Comparare stres studenti ---

Profesorul Profesor este mai putin stresant decit Georgescu, cu un indice de stress de 18.0 fata de 63.0

Profesorul cel mai stressant este Georgescu cu o rautaciune de 60.0 si o galagie de 70.0

Profesorul Popescu este mai stresant decit Ionescu, cu un indice de stress de 83.35 fata de 18.0

Profesorul cel mai stressant este Popescu cu o rautaciune de 80.5 si o galagie de 90.0

--- Profii care predau acelasi obiect ca primul prof ("Nu preda la moment!") ---

Profesor

Popescu

Dumitru

Numar total de profi creati: 8

--- ?tergerea fi?ierelor .txt existente (excluzând profText.txt) ---

?terg: profInscrisManual.txt.txt

Found a null entry, skipping.

Found a null entry, skipping.

**Concluzii.**

În acest lab, am aplicat și consolidat concepte esențiale din programarea orientată pe obiecte în Java. Am utilizat diverse unelte și tehnici pentru a crea o aplicație funcțională, de la modelarea datelor până la interacțiunea cu sistemul de fișiere.

**Ce am învățat**

* **Programare Orientată pe Obiecte (POO)**: Am lucrat cu elementele de bază ale POO, inclusiv definirea unei **clase** (Prof) cu atribute și metode, crearea de **obiecte** (instanțe ale clasei Prof) și utilizarea diferiților **constructori** (implicit, parametrizat, de copiere) pentru a inițializa aceste obiecte în moduri flexibile.
* **Manipularea structurilor de date**: Am gestionat o colecție de obiecte folosind un **array simplu**, pe care l-am tratat ca un vector dinamic. Am implementat o logică de redimensionare a array-ului pentru a adăuga mai mulți profesori decât capacitatea inițială.
* **Citirea și scrierea în fișiere**: Am învățat să interacționez cu sistemul de fișiere pentru a **persista datele** despre obiecte. Am utilizat Scanner pentru a citi dintr-un fișier și FileWriter pentru a scrie în altul, o abilitate esențială pentru salvarea și încărcarea datelor.
* **Metode statice**: Am înțeles și aplicat conceptul de **metodă statică** (stressCalc și getProfesori), care aparține clasei și poate fi apelată fără a crea o instanță a acesteia.

**Un minus** al acestui laborator este ca nu am implementat o exceptie pentru cind incercam sa scriem un fisier deja existent dar am recurs la rescriere.

*.*

* Chişinău 2020 -