Laborator 8

Inmatriculare studenti

Aplicatia are ca obiectiv familiarizarea cu clasele si interfetele din JCF.

Se va simula inmatricularea studentilor in universitate. Pentru a putea testa performantele cu un numar variabil de studenti se va evita introducerea datelor, iar studentii vor putea fi generati automat.

Fiecare student inmatriculat va fi inregistrat in

- Registrul matricol al Universitatii (fiecare student va primi un numar matricol de la 1 la n, iar regasirea studentului in *Registrul matricol* va fi posibila cu ajutorul numarului matricol;
- colectivul de studenti ai facultatii; veti considera structura de facultati a USV (FDSA, FEFS, FIA, FIESC, FIMM, FIG, FLSC, FS, FSEAP, FSE);
- registrul orientat pe gen care sa permita obtinerea listei tuturor studentelor si a tuturor studentilor.

Se va considera ca un student poate fi inscris o singura data in registrul matricol la aceeasi facultate; se va verifica sa nu fie inscris in Registrul matricol un student cu acelasi nume, aceeasi medie si aceeasi facultate (conditia ca 2 studenti sa fie egali) cu unul generat anterior.

Pentru a genera studenti veti scrie in clasa **Student** un constructor fara argumente care va genera date similar cu constructorul fara argumente al clasei **Persoana**.

Aplicatia trebuie sa produca urmatoarea iesire (pentru 1p veti asigura neaparat testarea cu un student identic cu altul generat anterior). In exemplul de mai jos s-au generat 9 studenti diferiti, al 6-lea fiind identic cu cel anterior.

```
//--------------------------- 1p ---------------------------------
stud. Cosmin (M) , 8.99, FIA *** student deja inscris in Registrul matricol

//--------------------------- 1p ---------------------------------
Raport
- total in registru 9 studenti

//--------------------------- 2p ---------------------------------
- inmatriculati pe facultati
      -FDSA : 0 studenti
      -FEFS : 3 studenti
      -FIA  : 1 studenti
      -FIESC      : 0 studenti
      -FIMM : 1 studenti
      -FIG  : 1 studenti
      -FLSC : 1 studenti
      -FS   : 0 studenti
      -FSEAP      : 2 studenti
      -FSE  : 0 studenti
//---------------------------  2p --------------------------
Baieti:4
Fete  :5
//---------------------------  2p --------------------------
Primele 10 nr. matricole:
1. stud. Georgiana (F) , 7.72, FEFS
2. stud. Iulia (F) , 5.22, FIG
3. stud. Maria (F) , 6.76, FSEAP
4. stud. Mirabela (F) , 8.58, FSEAP
5. stud. Cosmin (M) , 8.99, FIA
```

```
6. stud. Iulia (F) , 5.86, FEFS
7. stud. Vladimir (M) , 5.19, FEFS
8. stud. Ion (M) , 5.81, FIMM
9. stud. Ion (M) , 5.65, FLSC
10. null

//--------------------------  2p --------------------------
Alegeti facultatea(FDSA=1 FEFS=2 FIA=3 FIESC=4 FIMM=5 FIG=6 FLSC=7 FS=8
FSEAP=9 FSE=10  gata>=11):4
Nu sunt studenti inmtriculati
Alegeti facultatea(FDSA=1 FEFS=2 FIA=3 FIESC=4 FIMM=5 FIG=6 FLSC=7 FS=8
FSEAP=9 FSE=10  gata>=11):2
1. stud. Georgiana (F) , 7.72, FEFS
2. stud. Iulia (F) , 5.86, FEFS
3. stud. Vladimir (M) , 5.19, FEFS
Alegeti facultatea(FDSA=1 FEFS=2 FIA=3 FIESC=4 FIMM=5 FIG=6 FLSC=7 FS=8
FSEAP=9 FSE=10  gata>=11):11
*** Succes!
```

Aveti la dispozitie

enum Gen

```
public enum Gen {
    F,M;
}
```

enum Facultate

```
public enum Facultate {
    FDSA, FEFS, FIA, FIESC, FIMM, FIG, FLSC, FS, FSEAP, FSE;
    private static List<Facultate> l=Arrays.asList(values());
    final public static int nrFacultati = l.size();
    public static Facultate getFacultate(int i){
        return l.get(i);
    }
}
```

class Persoana

```
public class Persoana {
    private static String numeBaieti[] = {"Andrei","Liviu","Vasile","George",
"Lucian","Marius","Ion","Cosmin","Vladimir","Cosmin","Alexandru","Ciprian"};

    private static String numeFete[] = {"Maria","Mariana","Ana","Lucia","Denisa",
"Cristina","Ioana","Mirabela","Adina","Ligia","Iulia","Georgiana","Angela","Claudia"
};
    private static Random r = new Random();
    // ----------------------------------------------------

    private String nume;
    private Gen gen;
    public Persoana(){
        gen = r.nextInt(1000) < 511? Gen.F: Gen.M;
        nume = gen==Gen.F ? numeFete[r.nextInt(numeFete.length)]:
                            numeBaieti[r.nextInt(numeBaieti.length)];
    }
    public Persoana(String nume, Gen gen) {
        this.nume = nume;
        this.gen = gen;
    }
    public String getNume() {
        return nume;
    }
    public Gen getGen() {
        return gen;
    }
```

```
    @Override
    public String toString() {
        return  nume + " (" + gen +") ";
    }
}
```

Clasa Student - incompleta

```
public class Student  extends Persoana {
    private float medieAdmitere;
    private Facultate facultate;
    @Override
    public String toString() {
        return "stud. " +super.toString()+", " + medieAdmitere + ", " + facultate;
    }
    public Student(String nume, Gen gen, float medieAdmitere, Facultate facultate) {
        super(nume,gen);
        this.medieAdmitere = medieAdmitere;
        this.facultate = facultate;
    }
    public Student(){
        // trebuie completat ca sa genereze aleator date pt. a construi
        // un student (medie admitere >=5 si facultate)
    }
    public float getMedieAdmitere() {
        return medieAdmitere;
    }
    public Facultate getFacultate() {
        return facultate;
    }
    // clasa mai trebuie completata si cu altele pentru
    // ca aplicatia sa poata functiona asa cum s-a cerut
}
```

Trebuie sa realizati

Completarea clasei **Student** si scrierea unei noi clase**, Admitere,** care sa efectueze inmatricularea unui numar maxim de studenti si afisarea rapoartelor prezentate in exemplu.

Clasa **Admitere** va fi realizata conform design pattern-ului **Singleton** (depunctare cu 1p altfel) si va avea cel putin urmatoarele metode:

- *inmatriculeaza(n)* – inscrie maxim ***n*** studenti distincti generati aleator in toate cele 3 registre (matricol, colectivul de stud. ai facultati, cel orientat pe gen);
- *afiseazaStudFacultate()* - afiseaza studentii facultatii indicate de utilizator asa ca in exemplul prezentat anterior
- *main()*

TEMA ACASA

Utilizati structuri diferite pentru colectiile de studenti si determinati timpul de executie functie de numarul de studenti astfel:

Functia	10000	20000	30000	40000	50000	60000	70000	80000	90000
Inserare									
consultare									
eliminare									
cautare									

Pentru fiecare structura completati cate un astfel de tabel.

Punctaj: 2p pentru fiecare structura evaluata.