UNIVERSITATEA DIN BUCUREȘTI
FACULTATEA DE MATEMATICĂ ȘI INFORMATICĂ
DEPARTAMENTUL DE INFORMATICĂ
DOMENIUL DE STUDII INFORMATICĂ
SERIILE 23 ȘI 24

VERIFICARE LA DISCIPLINA "PROGRAMARE AVANSATĂ PE OBIECTE"
- SESIUNEA MAI/IUNIE 2019 -

I. Pentru fiecare dintre cele 5 întrebări de mai jos, indicați varianta de răspuns pe care o considerați corectă:

1. Fie următorul program Java:

```
class Automobil{
    private String marca;
    public Automobil(String marca) { this.marca = marca; }
    public int hashCode() { return 0; }
    public boolean equals(Object obj) { return true; }
}

public class Test {
    public static void main(String[] args) {
        HashSet<Automobil> la = new HashSet<>();
        la.add(new Automobil("Audi"));
        la.add(new Automobil("BMW"));
        la.add(new Automobil("Audi"));
        la.add(new Automobil("Opel"));
        System.out.println(la.size());
    }
}
```

După executarea programului, va fi afișată valoarea:
a) 4 b) 3 c) 2 d) 1

2. Precizați care dintre următoarele afirmații este adevărată pentru clasele imutabile:
a) sunt implicit și clase de tip singleton
b) au toate datele membre de tip final
c) au un singur constructor, fără parametri
d) referințele spre instanțele lor nu pot fi modificate

3. Fie următorul program Java:

```
class A {
    public static void metoda(String s) {System.out.print("A"+s); }
}

class B extends A {
    public static void metoda(String s) {System.out.print("B"+s); }
    public void metoda(String s,String t) { System.out.print("B"+s+t); }
}

public class Test{
    public static void main(String[] args) {
        A ob = new B();
        ob.metoda("P");
        ob.metoda("Q", "R");
    }
}
```

Precizați care dintre următoarele afirmații este adevărată pentru programul dat:
a) există erori de compilare în clasa `Test` b) există erori de compilare în clasa `B`
c) va afișa `BPBQR` după rulare d) va afișa `APBQR` după rulare

4. Fie următorul program Java:

```
class Persoana implements Serializable {
    String nume;
    int varsta;

    public Persoana(String nume, int varsta) {
        this.nume = nume;
        this.varsta = varsta;
        System.out.println("Constructor");
    }
}

public class Test {
    public static void main(String[] args) throws Exception {
        ObjectOutputStream oos = new ObjectOutputStream(
                                 new FileOutputStream("persoana.ser"));
        Persoana p = new Persoana("Popescu Ion", 40), q = p;
        oos.writeObject(q);
        oos.close();
        ObjectInputStream ois = new ObjectInputStream(
                                new FileInputStream("persoana.ser"));
        Persoana r = (Persoana)ois.readObject();
        ois.close();
    }
}
```

De câte ori va fi afișat mesajul *Constructor*, după executarea programului dat?
a) niciodată b) o dată c) de două ori d) de trei ori

5. Fie următorul program Java:

```
class Test {
    static String sir = "A";

    void A() {
        try {
            sir = sir + "B";
            B();
        } catch (Exception e) { sir = sir + "C"; }
    }

    void B() throws Exception {
        try {
            sir = sir + "D";
            C();
        }
        catch (Exception e) { throw new Exception(); }
        finally { sir = sir + "E"; }
    }
```

```
    void C() throws Exception { throw new Exception(); }

    public static void main(String[] args) {
        Test ob = new Test();
        ob.A();
        System.out.println(sir);
    }
}
```

După executarea programului, se va afișa:
a) ABCDE b) ABDE c) ABDEC d) ABCD

II. Scrieți o clasă Java care să determine toate cuvintele de lungime $n \geq 1$ dintr-un fișier text, folosind un fir de executare. Scrieți un program care citește de la tastatură un număr natural n și, utilizând clasa definită anterior, afișează toate cuvintele distincte de lungime n din fișierele text *poezie_1.txt* și *poezie_2.txt*. Cuvintele din fișierele text de intrare sunt despărțite între ele prin spații și semnele de punctuație uzuale.

III. Considerăm definită clasa *DiplomăLicență* cu datele membre *serie*, *absolvent*, *an*, *facultate*, *specializare* și *medie*. Clasa încapsulează constructori, metode de tip *set/get* pentru toate datele membre, precum și metodele *toString()*, *equals()* și *hashCode()*. Creați o listă care să conțină cel puțin 3 obiecte de tip *DiplomăLicență* și, folosind stream-uri bazate pe lista creată și lambda expresii, rezolvați următoarele cerințe:

- afișați diplomele eliberate între anii 2000 și 2010, în ordinea descrescătoare a mediilor;
- afișați specializările distincte pentru care au fost eliberate diplome în Facultatea de Informatică în anul 2018;
- creați o listă formată din numele absolvenților care au obținut media 10;
- afișați numărul diplomelor eliberate pentru specializarea Informatică.

IV. Se consideră baza de date *Diplome*, având următorul URL: *jdbc:derby://localhost:1527/Diplome*. Baza de date conține tabela *DiplomeLicență*, având câmpurile *serie*, *absolvent*, *an*, *facultate*, *specializare* și *medie*. Scrieți un program care să citească de la tastatură două valori întregi *an_min* și *an_max*, precum și denumirea *den_spec* a unei specializări, după care creează o listă cu informații despre diplomele eliberate între anii *an_min* și *an_max* pentru specializarea *den_spec*. Elementele listei vor fi obiecte de tipul clasei *DiplomăLicență* definită anterior (vom considera faptul că aceasta implementează interfața *Serializable*). Dacă lista obținută nu este vidă, aceasta va fi serializată în fișierul *diplome.dat*, altfel se va afișa pe ecran un mesaj corespunzător.

NOTĂ:

- Clasa *DiplomăLicență* se consideră definită complet, deci **NU** trebuie să o implementați!!!
- Datele de intrare se consideră corecte.
- Nu se vor trata excepțiile.
- Punctaj: 2.5p. (5 x 0.5p.) + 2p. + 2.5p. + 2p. + 1p. (din oficiu)