**EXAMEN LA DISCIPLINA "PROGRAMARE AVANSATĂ PE OBIECTE"**

**ANUL II – SESIUNEA IUNIE 2016**

1. Considerăm următoarea clasă abstractă Produs:

**abstract** **class** Produs **implements** Serializable {

**protected** **String** denumire, firma;

**protected** **double** pretBrut;

**protected int** stoc**;**

**public** Produs(**String** denumire, **String** firma, **double** pretBrut,

**int** stoc){

**this**.denumire = denumire;

**this**.firma = firma;

**this**.pretBrut = pretBrut;

**this**.stoc = stoc;

}

**public** **abstract** **double** calculPretVanzare();

@Override

**public** **String** toString(){

**return** denumire + "," + firma + "," + pretBrut + "," + stoc;

}

//clasa conține și metodele de tip get/set necesare!

}

1. Definiți subclasele ProdusAlimentar și ProdusElectronic ale clasei Produs cu următoarele proprietăți:
2. pentru un produs alimentar se poate memora termenul de garanție, iar pentru un produs electronic se poate memora clasa energetică;
3. fiecare clasă conține un constructor cu parametri și metoda toString();
4. fiecare clasă implementează metoda calculPretVanzare() din superclasa Produs ținând cont de faptul că TVA-ul este de 5% pentru produsele alimentare și 20% pentru cele electronice;
5. obiectele din clasa ProdusAlimentar sunt serializabile, iar cele din clasa ProdusElectronic sunt externalizabile, mai puțin data membră firma.
6. Scrieți un program Java care să citească într-un tablou unidimensional informații despre mai multe produse din fișierul text *produse.txt*. Fișierul conține pe prima linie numărul de produse, iar fiecare dintre următoarele linii conține informațiile specifice fiecărui produs, despărțite prin virgule. O linie începe cu 1 dacă ea conține informații despre un produs alimentar sau cu 2 dacă ea conține informații despre un produs electronic.
7. Folosind interfața Comparator, sortați produsele din tablou în ordinea descrescătoare a valorilor totale ale stocurilor lor.
8. Scrieți o clasă Java care să calculeze, folosind un fir de executare, suma numerelor întregi de pe fiecare linie a unui fișier text și să scrie sumele obținute pe prima linie a unui alt fișier text, despărțite tot prin câte un spațiu. Folosind clasa respectivă, scrieți un program care să calculeze suma numerelor întregi din fișierul text *numere.txt*, iar rezultatul obținut să fie scris în fișierul text *suma.txt*.
9. Fișierul text *agenda.txt* conține numerele de telefon ale mai multor persoane. Fiecare linie din agendă este de forma *nume prenume,telefon*. Scrieți un servlet care să primească, prin intermediul unui parametru transmis folosind metoda GET, un prenume și să genereze o pagină HTML care să conțină numele și numerele de telefon ale tuturor persoanelor din agendă care au prenumele respectiv, precum și numărul lor. Conținutul agendei telefonice se va încărca din fișierul text într-o structură de date adecvată o singură dată, în momentul inițializării servlet-ului.

**NOTĂ:**

* Datele de intrare se consideră corecte.
* Punctaj: 3p. (1p. + 1p. + 1p.) + 3p. + 3p. + 1p. (din oficiu)
* Interfața Externalizable:

**public** **interface** Externalizable **extends** [Serializable](https://docs.oracle.com/javase/7/docs/api/java/io/Serializable.html){

**public** **void** readExternal(ObjectInput ob)**throws** IOException,

ClassNotFoundException;

**public void** writeExternal(ObjectOutput ob) **throws** IOException; }

* Metode ale clasei abstracte HttpServlet:
* **protected void** doGet(HttpServletRequest request,

HttpServletResponse response) **throws** ServletException,

IOException;

* **protected** **void** doPost(HttpServletRequest request,

HttpServletResponse response) **throws** ServletException,

IOException;

* **public** **void** init() **throws** ServletException;
* **public** **void** destroy();