Programare orientată pe obiecte – Test de laborator Seria 13/14 - 26 mai 2020

Enunţ

Un producător de măşti de protecţie şi dezinfectanţi doreşte să pornească o afacere (cât mai urgent posibil în contextul actual) şi în acest sens el doreşte să creeze un sistem de gestionare a măştilor şi dezinfectanţilor pe care le fabrică. Producătorul doreşte să realizeze aplicaţia în C++ folosind concepte de OOP.

Producătorul îşi propune să fabrice măşti chirurgicale dar şi măşti din policarbonat. Iar pentru secţia de dezinfectanţi, se doreşte fabricarea de dezinfectanţi împotriva bacteriilor, împotriva fungilor dar şi împotriva virusurilor. Bineînţeles, producătorul va fabrica şi dezinfectant care ucide toate aceste microorganisme.

Despre un model de mască chirurgicală, producătorul doreşte să reţină următoarele informaţii: tipul de protecţie (ffp1, ffp2, ffp3), culoarea şi numărul de pliuri. În plus pentru un model de mască din policarbonat se doreşte să se reţină: tipul de protecţie (care este ffp 0) şi tipul de prindere (elastic, scai, etc.). Pentru fiecare tip de dezinfectant în parte se doreşte să se reţină: numărul de specii de organisme ce vor fi ucise, ingredientele şi tipurile de suprafaţă pe care poate fi aplicat (spre exemplu dezinfectantul antifungic nu se aplică pe marmură deoarece poate albi marmura).
Pentru a calcula eficienţa unui dezinfectant vom recurge la următoarea formulă:

$$Eficienta_{TipOrganism} = \begin{cases} \frac{\#specii\ de\ bacterii\ ucise}{TotalBacterii}, TipOrganism = bacterie \\ \frac{\#specii\ de\ fungi\ ucisi}{TotalFungi}, TipOrganism = fung \\ \frac{\#specii\ de\ virusuri\ ucise}{TotalVirusuri}, TipOrganism = virus \end{cases}$$

Unde:

$$\begin{cases} TotalBacterii = 10^9 \\ TotalFungi = 1{,}5 \cdot 10^6 \\ TotalVirusuri = 10^8 \end{cases}$$

Ca orice producător care funcţionează legal, acesta va dori să reţină o arhivă contabilă – toate achiziţiile onorate. Pentru o achiziţie se vor reţine data, numele clientului, un vector de pointeri către dezinfectanţii achiziţionaţi şi un vector de pointeri către măştile achiziţionate. În plus pentru fiecare achiziţie trebuie sa se ştie totalul comenzii. Pentru acesta, se ştiu următoarele preţuri, raportate la bucată:

$$\begin{cases} masca\ policarbonat \rightarrow 20\ lei \\ masca\ textila\ ffp1 \rightarrow 5\ lei \\ masca\ textila\ ffp2 \rightarrow 10\ lei \\ masca\ textila\ ffp3 \rightarrow 15\ lei \\ dezinfectant\ cu\ eficienta < 90\% \rightarrow 10\ lei \\ dezinfectant\ cu\ eficienta < 95\% \rightarrow 20\ lei \\ dezinfectant\ cu\ eficienta < 97.5\% \rightarrow 30\ lei \\ dezinfectant\ cu\ eficienta < 99\% \rightarrow 40\ lei \\ dezinfectant\ cu\ eficienta < 99.99\% \rightarrow 50\ lei \end{cases}$$

În plus, în momentul în care un client comanda o mască chirurgicală, acesta poate solicita un anumit model al măştii (de exemplu un steag sau un logo al unei echipe sportive). Pentru acest tip de preferinţă la preţul unei măşti se adaugă 50% faţă de preţul normal.

Cerinţe:

I. Să se implementeze clasele necesare astfel încât aplicaţia să funcţioneze.
II. Programul este prevăzut cu un meniu interactiv în care administratorul are posibilitatea de a alege dintre următoarele opţiuni:
    1. Adaugă un nou dezinfectant în stoc
    2. Adaugă o nouă mască în stoc
    3. Adaugă o nouă achiziţie
    4. Afişează dezinfectantul cel mai eficient
    5. Calculează venitul dintr-o anumită lună
    6. Calculează venitul obţinut din măştile chirurgicale cu model.
    7. Modifică reţeta unui dezinfectant, modificând unul sau mai multe ingrediente, lucru ce conduce şi la o modificarea a numărului de specii de organisme ce vor fi ucise şi a suprafeţelor pe care poate fi aplicat.
    8. Afişează cel mai fidel client (clienţii NU au nume unic)
    9. Afişează ziua cu cele mai slabe venituri, de la deschidere până în prezent.
    10. Calculeaza TVA-ul (19% din venituri) care trebuie returnat la ANAF pentru un anumit an.

Demo:

```
MascaChirugicala mc1, mc2("ffp2", "verde brotăcel", 55), mc3(m1), mc4, mc5;
mc4 = mc2;
std::cin >> mc5;
std::cout << mc1 << mc2;
MascaPolicarbonat* mp1=new MascaPolicarbonat(), * mp2=new MascaPolicarbonat();
```

```
MascaPolicarbonat* mp3 = new MascaPolicarbonat("elastic");
std::cin >> mp1 >> mp2;
std::cout <<mp3;


Dezinfectant* d1 = new DezinfectantBacterii(100000000, std::vector<string>({"sulfati non-ionici", "sulfati cationici", "parfumuri", "Linalool", "Metilpropanol butilpentil"}), std::vector({"lemn, sticla, metal, ceramica, marmura"}));

Dezinfectant* d2 = new DezinfectantVirusuri(50000000, std::vector<string>({"Alkil Dimetilm Benzil Crlorura de amoniu", "parfumuri", "Butilpentil metilpropinal"}), std::vector({"lemn, sticla, ceramica, marmura"}));

Dezinfectant* d3 = new DezinfectantFungi(1400000, std::vector<string>({"Alkil Etil Benzil Crlorura de amoniu", "parfumuri", "Butilpentil metilpropinal"}), std::vector({"sticla, plastic"}));
std::cout << d1->eficienta() << " " << d2->eficienta() << " " << d3->eficienta() << "\n";



Achizitie* a1 = new Achizitie(26, 5, 2020, "PlushBio SRL");
*a1 += mp1; //se adauga masca de policarbonat mp1 in lista de masti achizitionate
*a1 += (&mc1); //se adauga masca chirugicala mc1 in lista
*a1 += d3; // se adauga dezinfectantu de fungi d3 in lista de dezinfectanti achizitionati

Achizitie* a2 = new Achizitie(25, 5, 2020, "Gucci");
*a2 += d1;
*a2 += d2;
*a2 += d2;

Achizitie a3, a4(*a1);
a3 = *a2;

if(*a1 < *a2) {
        std::cout << a1->nume() << " are valoarea facturii mai mica.\n";
}else if (*a1 == *a2) {
        std::cout << a1->nume() << " si " << a2->nume() << " au aceasi valoare a facturii.\n";
else {
        std::cout << a2->nume() << " are valoarea facturii mai mica.\n";
}

//Continuati voi cu restul de cerințe
```

Precizări

1. Timpul de lucru este de 150 de minute. Orice lucrare trimisă după 11:30 nu va fi luată în considerare.
2. Pentru întrebări despre subiect, tutorii de laborator vor fi online pe Google Meets, la adresa: meet.google.com/vbc-kqnh-ojc
3. La sfârşitul timpului de lucru, studenţii vor trimite pe adresa examen.oop.fmi@gmail.com fişierul/fişierele sursă cu extensia cpp/h/hpp (fără executabil). Acesta trebuie să conţină pe primul rând un comentariu cu numele şi prenumele studentului, grupa şi compilatorul folosit.
4. Puteţi submita mai multe surse, dar să menţionaţi în ultimul mail care să fie cea corectată. Alt fel, ultima sursă trimisă va fi notată.
5. Sursa predată trebuie să compileze. Sursele care au erori de compilare nu vor fi luate în considerare. Înainte de predarea surselor, studenţii vor pune în comentariu eventualele părţi din program care au erori de compilare sau nu funcţionează corespunzător.
6. Se acceptă şi soluţii parţiale, care nu respectă toate cerinţele din enunţ, dar sunt funcţionale. Acestea vor fi depunctate corespunzător.
7. În implementarea programului se vor utiliza cât mai multe dintre noţiunile de programare orientată pe obiecte, care au fost studiate pe parcursul semestrului şi care se potrivesc cerinţelor din enunţ.
8. Condiţiile minimale de promovare a testului sunt ca codul din demo să ruleze.
9. Orice tentativă de fraudă se va pedepsi conform regulamentelor Universităţii.