Programare orientată pe obiecte – Test de laborator Seria 25 - 15 ian 2020

Enunţ

O un producător de masini dorește să pornească o afacere și în acest sens el dorește să creeze un sistem de gestionare a modelelor de autoturisme pe care le fabrică. Producătorul dorește să realizeze aplicaţia în C++ folosind concepte de OOP.

Producătorul își propune să creeze mașini care funcţionează pe bază de combustibili fosili (care pot fi de două tipuri: benzină și motorină), mașini electrice și mașini hibride (care pot fi alimentate atât electric, cât și benzină)

Despre un model de mașină, producătorul dorește să reţină următoarele informaţii: anul începerii producţiei, numele modelul, viteza maximă și greutatea.

În plus, pentru un model de mașină pe bază de combustibili fosili se mai reţin: tipul combustibilului (benzină sau motorină) și capacitatea rezervorului. Pe de altă parte, pentru un model electric se reţine adiţional: capacitatea bateriei.

Autonomia unei mașini dintr-un model se calculează:

- pentru modele pe combustibili fosili: raportul dintre capacitatea rezervorului și radical de ordinul doi din greutate
- pentru modele electrice: raportul dintre capacitatea bateriei și greutate la pătrat
- pentru modele hibride: suma celor două definite mai sus

Producătorul dorește de asemenea de la program să memoreze o listă a tranzacţiilor efectuate de-a lungul timpului cu diverși clienţi. O tranzacţie este definită prin: numele clientului, data tranzacţiei și o listă de modele achiziţionate.

Un exemplu de tranzacţie: {Popescu, 5 5 2019, [Masina_1, Masina_2, Masina_3]}, unde Masina_i sunt obiecte/pointeri/referinţe din clase reprezentând modele, **NU** nume de modele.

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 model mașină citind de la tastatură tipul mașinii și apoi câmpurile specifice ei
   2. Adaugă o tranzacţie
   3. Afișează cel mai vândut model
   4. Afișează modelul cu autonomia cea mai mare
   5. Aduce o optimizare unui anumit model crescandu-i viteza maximă cu un anumit procent.

Precizări

1. Timpul de lucru este de 90 de minute.
2. La sfârșitul timpului de lucru, studenţii vor trimite pe adresa **examen.oop.fmi@gmail.com** fișierul sursă cu extensia cpp. Acesta trebuie să conţină pe primul rând un comentariu cu numele și prenumele studentului, grupa și compilatorul folosit.
3. 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.
4. Se acceptă și soluţii parţiale, care nu respectă toate cerinţele din enunţ, dar sunt funcţionale. Acestea vor fi depunctate corespunzător.
5. În implementarea programului se vor utiliza cât mai multe dintre noţiunile de programare orientată pe obiecte, care au fost studiate pe parcursul semestrului și care se potrivesc cerinţelor din enunţ.

6. Condiţiile minimale de promovare a testului sunt ca programul să fie scris cu clase şi să citească modelele de maşini unul câte unul, să le memoreze datele şi apoi să le afişeze.
7. Orice tentativă de fraudă se va pedepsi conform regulamentelor Universităţii.

Barem

1. Se acordă 1 punct din oficiu.
2. Dacă sursa nu compilează, nu se mai acordă niciun alt punct (nota 1).
3. Se acordă 4 puncte pentru respectarea condiţiilor minimale, prevăzute în enunţ, (programul să fie scris cu clase şi să ruleze corect funcţia main() în varianta demo) astfel, dintre care:
   2,5 puncte pentru definirea claselor:
   a. Clasa Model
      - constructor de iniţializare fără parametri - 3 x 0,1 p
      - constructor de iniţializare parametrizat - 3 x 0,1 p
      - destructor virtual - 3 x 0,1 p
      - definirea corectă a câmpurilor clasei - 3 x 0,1 p

   b. implementarea metodei calculAutonomie() în clasele ModelFosil, ModelElectric, ModelHibrid - 3 x 0,1 p
   c. Clasa Tranzactie
      - constructor de iniţializare parametrizat - 0,2 p
      - constructor de copiere - 0,2 p
      - destructor - 0,2 p
      - operator de atribuire - 0,2 p
      - implementarea listei de Model* 0,2 p

   iar 1.5 puncte se acordă pentru respectarea condiţiilor minimale, prevăzute în enunţ, astfel:
   d. 0.5 puncte dacă modelele de orice tip (combustibili fosili, electrice şi hibride) sunt citite corect, cu toate informaţiile aferente
   e. 0.5 puncte dacă modelele de orice tip (combustibili fosili, electrice şi hibride) sunt memorate corect într-o listă sau într-o altă structură
   f. 0.5 puncte dacă modelele de orice tip (combustibili fosili, electrice şi hibride) sunt afişate corect, cu toate informaţiile aferente.

   Pentru oricare dintre cerinţele a., b., şi c., dacă cerinţa este îndeplinită parţial, se acordă 0,25 puncte.
4. Se acordă 1 punct pentru folosirea corectă în contextul dat a supraîncărcării operatorilor. Dacă cerinţa este îndeplinită parţial (sau doar pentru << si >>), se acordă 0,5 puncte.
5. Se acordă 1 punct pentru definirea unei ierarhii de clase (**Model** -> **ModelFosil**, **ModelElectric, ModelHibrid**). Dacă cerinţa este îndeplinită parţial, se acordă 0,5 puncte.
6. Se acordă 1 punct pentru definirea corectă a unei ierarhii de clase de tip romb (moştenire multiplă şi virtuală). Nu se acordă punctaje parţiale.
7. Se acordă câte 0,4 puncte pentru rezolvarea corectă a cerinţelor II.1., II.2., II.3., II.4., II.5., II.6. din enunţ. Nu se acordă punctaje parţiale.
8. Se scade 1 punct pentru utilizarea incorectă a constructorilor şi destructorilor. Dacă cerinţa este încălcată parţial, se scad 0,5 puncte.
9. Se scade 1 punct pentru utilizarea incorectă a încapsulării (câmpurile private şi metode publice). Dacă cerinţa este încălcată parţial, se scad 0,5 puncte.
10. Se acordă 1 punct pentru folosirea corectă în contextul dat a cel putin 2 elemente deosebite de OOP (clase şablon, tratarea excepţiilor, design patterns, etc.). Dacă cerinţa este îndeplinită parţial, se acordă 0,5 puncte.
11. Se acordă 0,5 puncte pentru folosirea corectă în contextul dat a metodelor virtuale. Nu se acordă punctaje parţiale.

12. Se acordă 0,5 puncte pentru folosirea corectă în contextul dat a unei clase abstracte.

Observaţii de evaluare

1. Nota maximă este 12.
2. Erorile minore, care sunt cauzate de neatenţie şi care nu se repetă în mai multe locuri (de ex., a uitat să scrie un caracter şi nu îi compilează, a greşit un semn sau un coeficient într-o formulă, nu afişează primul sau ultimul element dintr-o listă etc) pot fi corectate şi trecute cu vederea (se poate acorda punctajul maxim prevăzut pentru o cerinţă îndeplinită corect din cauza unei erori minore).