Programe OO în C++

Obiective

Specificarea, proiectarea şi implementarea problemelor în C++.

Aspecte teoretice

Aspecte generale ale limbajului C++.
Aplicaţii în C++. Programare orientată pe obiecte..

Termen de predare

Problemele trebuie rezolvate parțial în cadrul laboratorului, parțial acasă.

Cerinţe

Să se rezolve una dintre problemele de mai jos:
Lab 4:

- LIVE:
  - implementat clasa entitate cu un singur atribut & teste (2p)
  - implementare clasa Repo (**doar** constructorii, destructorul si metodele *addElem*, *getAll*, *getSize*) & teste (3p)
  - Pentru a fi valide aceste puncte trebuie sa existe un commit in GItHub-classroom pt aceasta parte de cod in cadrul celor 2 ore de laborator din Lab4 (aferente fiecarei subgrupe).
- TEMA (predare în Lab 5)
  - dezvoltarea completă a entității și metodelor din Repo & teste complete (5p)
  - Pentru a fi valide aceste puncte trebuie sa existe un commit in GItHub-classroom pt aceasta parte de cod inainte de finalul celei de a 2-a ore a laboratorului 5 (aferente fiecarei subgrupe).

Precizari:

- clasa entitate va folosi char* pentru atributul de tip sir de caractere (ex. char* nume)
- clasa repo va folosi un array de entitati (static sau dinamic)
- in clase se vor implementa toate tipurile de constructori, destructorul, setteri, getterii, operator= (si, optional, alte metode si operatori)

Lab 5:

- - LIVE:
  - implementare clasa Service (operatii CRUD) (2p) si teste aferente (1p)
- - TEMA (predare în Lab 6):
  - implementare clasa UI (2p)
  - functionalitati propriu-zise (in plus fata de CRUD) (3p)
  - restul de teste pt Service (1p)
  - undo (1p)

Exemple de probleme:

1. **Cheltuieli de familie**

O familie dorește să își organizeze cheltuielile lunare și are nevoie de o aplicație care să permită stocarea tuturor cheltuielilor dintr-o anumită lună. Pentru fiecare cheltuială se cunosc ziua din lună în care a fost efectuată (între 1 și 28/29/30/31), suma de bani (întreg pozitiv) și tipul (menaj, mâncare, transport, haine, internet, altele). Familia are nevoie de o aplicație cu următoarele funcționalități:

- adăugarea unei cheltuieli în lista (add, insert)
  - ex. adaugă 10 internet - adaugă pentru ziua curentă o cheltuială de 10 RON pentru internet
  - ex. inserează 25 100 mâncare - inserează pentru ziua 25 a lunii curente o cheltuială de 100 RON pentru mâncare
- modificarea cheltuielilor din listă
  - ex. elimină 15 - elimină toate cheltuielile din ziua 15
  - ex. elimină 2 la 9 - elimină toate cheltuielile pentru zilele 2,3, ..., 9
  - ex. elimină mâncare - elimină totate cheltuielile pentru mâncare din luna curentă.
- stabilirea cheltuielilor cu anumite proprietăți
  - ex. listează - afișarea tuturor cheltuielilor din luna curentă
  - ex. listează mancare - afișarea tuturor cheltuielilor pentru mâncare
  - ex. listează mâncare > 5 - afișarea tuturor cheltuielilor pentru mâncare mai mari de 5 RON
  - ex. listează internet = 44 - afișarea tuturor cheltuielilor pentru internet în valoare de 44 RON
- obținerea unor proprietăți a diferitelor subliste
  - ex. suma mâncare - suma tuturor cheltuielilor din categoria mâncare
  - ex. max zi - afișarea zilei cu cele mai mari cheltuieli
  - ex. sortare zi - afișarea sumelor cheltuite zilnic în ordine descrescătoare
  - ex. sortează mâncare - afișarea sumelor cheltuite zilnic pentru mâncare, ordonate crescător
- filtrarea listei de cheltuieli
  - ex. filtrare mâncare - păstrează doar cheltuielile pentru mâncare
  - ex. filtrare menaj < 100 - păstrează doar cheltuielile pentru menaj mai mici decât 100 RON
  - ex. filtrare haine = 59 - păstrează doar cheltuielile pentru haine egale cu 59 RON
- undo - reface ultima operație. Aplicația trebuie să permită revenirea (prin undo) l starea inițială.

2. **Gestiune conturi bancare**

John vrea să-și gestioneze conturile bancare și are nevoie de o aplicație care să-i permită stocarea și vizualizarea tranzacțiilor bancare efectuate în contul său în timpul unei luni calendaristice. Fiecare tranzacție este identificată prin: ziua din lună când a

fost efectuată (întreg între 1 și 28/29/30/31), suma de bani (întreg pozitiv), tip (in - au intrat bani în cont - sau out - au ieșit bani din cont) și descriere. Creați o aplicație care să-l ajute pe John oferindu-i următoarele funcționalități:

- adăugarea unei tranzacții în lista de tranzacții
  - ex. adaugă 100 out pizza - este adăugată o tranzație de tip out în valoarea de 100 RON pentru ziua curentă cu descrierea ”pizza”
  - ex. inserează 25 100 in salar - inserează o tranzacție de tip in pentru ziua 25 în valaorea de 100 RON cu descrierea salar
- modificarea tranzacțiilor din listă
  - ex. eliminare 15 - se elimină toate tranzacțiile din ziua 15
  - ex. eliminare 5 la 10 - se elimină toate tranzacțiile din ziua 10 până în ziua 15
  - ex. eliminare in - se elimină toate tranzacțiile de tip in din luna curentă
  - ex. înlocuire 12 in salar cu 2000 - se înlocuiește suma vechea tranzacției de tip in din ziua 12 cu descrierea ”salar” cu suma 2000
- identificarea tranzacțiilor cu anumite proprietăți
  - ex. listează - afișează toate tranzacțiile din listă
  - ex. listează in - afișează toate tranzacțiile de tip in din listă
  - ex. listează > 100 - afișează toate tranzacțiile din listă cu suma mai mare decât 100
  - ex. listează = 60 - afișează toate tranzacțiile din listă cu suma egală cu 60
  - ex. listează sol 10 - afișează sold-ul tranzacțiilor din ziua 10 (diferența între suma tranzacțiilor de tip in și suma tranzacțiilor de tip out)
- obținerea unor caracteristici ale tranzacțiilor
  - ex. sum in - afișează suma totală a tranzacțiilor de tip in
  - ex. max out 15 - afișează cea mai valoroasă tranzacție de tip out din ziua 15
- filtrări
  - ex. filtru in - se păstrează doar tranzacțiile de tip in
  - ex. filtru in 100 - se păstrează doar tranzacțiile de tip in cu valoare mai mică decât 100
- undo - reface ultima operație. Aplicația trebuie să permită revenirea (prin undo) l starea inițială.

3. **Gestiune apartamente**

Jane este admistratoarea unei clădiri de apartamente și dorește să gestioneze cheltuielile fiecărui apartament din cursul unei luni. Fiecare cheltuială este identificată prin: numărul apartamentului (întreg pozitiv), suma (întreg pozitiv), tipul (dintr-o listă de categorii predefinite precum apă, căldură, electricitate, gaz, altele). Ajutați-o pe Jane realizaând o aplicație cu următoarele funcționalități:

- adăugarea unei cheltuieli în lista de cheltuieli
  - ex. adaugă 25 gaz 100 - adaugă pentru apartamentul 25 o cheltuială în valaore de 100 RON pentru gaz
- modificarea cheltuielilor din listă
  - ex. eliminare 15 - se elimină toate cheltuielile apartamentului 15
  - ex. eliminare 5 la 10 - se elimină toate cheltuielile apartamentelor 5, 6, ..., 10

  - ex. eliminare gaz - se elimină toate cheltuielile cu gazul de la toate apartamentele
  - ex. înlocuire 12 gaz cu 2000 - se înlocuiește suma veche a cheltuielii cu gazul de la apartamentul 12 cu suma 2000
- identificarea cheltuielilor cu anumite proprietăți
  - ex. listează - afișează toate cheltuielile din listă
  - ex. listează 15 - afișează toate cheltuielile apartamentului 15
  - ex. listează > 100 - afișează cheltuielile în valoare mai mare de 100 RON pentru toate apartamentele
  - ex. listează = 60 - afișează cheltuielile în valoare de 60 RON pentru toate apartamentele
- obținerea unor caracteristici ale cheltuielilor
  - ex. sum gaz - afișează suma totală a cheltuielilor cu gazul pentru toate apartamentele
  - ex. max 15 - afișează cea mai valoroasă cheltuială a apartamentului 15
  - ex. sortează gaz - ordonează descrescător cheltuielile cu gazul a tuturor apartamentelor
- filtrări
  - ex. filtru gaz - se păstrează doar cheltuielile pentru gaz
  - ex. filtru 100 - se păstrează doar cheltuielile cu valoare mai mică decât 100
- undo - reface ultima operație. Aplicația trebuie să permită revenirea (prin undo) l starea inițială.