

Implementați o aplicație în limbajul C care implementează soluții la probleme de gestionare a a conturilor bancare deschise la o bancă comercială.

1. Scrieți secvența de cod sursă pentru crearea unei structuri de date de tip **Tabelă de Dispersie** ce conține date aferente conturilor bancare. Cheia de căutare utilizată este **nume_client**, iar mecanismul de tratare a coliziunilor este **Chaining**. Inserarea unui cont bancar se implementează într-o funcție care se apelează în secvența de creare a structurii **Tabelă de Dispersie**. Tabela de dispersie conține minim 10 conturi bancare incarcate in aplicatie dintr-un fisier de intrare. Structura **ContBancar** se va defini astfel încât să conțină minim 7 câmpuri, astfel: **nume_client (char*)**, **sold (float)**, **valuta_cont (char*)**; celelalte 4 campuri sunt definite la alegere.

Cerințe de implementare:

- Definire structură **ContBancar**. (0,25p)
- String-urile preluate din fișier trebuie să accepte prezența simbolului **blank**. (0,25p)
- Absență memory leaks. (0,25p)
- Implementare logică de creare structură **Tabelă de Dispersie** cu **Chaining**. (0,75p)
- Populare completă și corectă a structurii **Tabelă de Dispersie** cu date de intrare din fisier. (0,25p)
- Testare implementare cu afisarea la consola a continutului structurii **Tabelă de Dispersie**. (0,25p)

2. Scrieți și apelați funcția pentru determinarea conturilor bancare din structura creată la cerinta 1) care au valuta specificata ca parametru de intrare al functiei. Conturile bancare identificate sunt salvate într-un vector și **NU** partajează zone de memorie heap cu structura **Tabelă de Dispersie**. Vectorul se returnează în **main()** prin tipul de retur sau lista de parametri ai funcției.

Cerințe de implementare:

- Definire funcție cu parametri de I/O definiți complet și corect. (0,25p)
- Realizare deep-copy a conturilor bancare în vector. (0,25p)
- Implementare logică de determinare și salvare a conturilor în vector. (1,00p)
- Populare completă și corectă a vectorului. (0,25p)
- Testare implementare prin apel de functie si afisare la consola a rezultatului obtinut la apel. (0,25p)

3. Scrieți și apelați funcția pentru determinarea numarului si a dimensiunilor (exprimate ca numar de conturi bancare) pentru toate cluster-ele de coliziuni din **Tabela de Dispersie**. Cluster-ele identificate sunt salvate într-un vector in care fiecare element contine perechea de valori (**dimensiune_cluster**, **index_tabela**). Vectorul si dimensiunea acestuia se returnează în **main()** prin tipul de retur sau lista de parametri ai funcției.

Cerințe de implementare:

- Definire funcție cu parametri de I/O definiți complet și corect. (0,25p)
- Determinare numar cluster-e. (0,25p)
- Determinare dimensiuni si indecsi cluster-e. (0,25p)
- Implementare logică de determinare și salvare a cluster-elor în vector. (1,00p)
- Populare completă și corectă a vectorului. (0,25p)
- Testare implementare prin apel de functie si afisare la consola a rezultatului obtinut la apel. (0,25p)

4. Scrieți și apelați funcția pentru determinarea soldurilor bancare totale la nivel de client. Se iau in considerare conturile bancare salvate in vectorul de la la cerinta 2). Un client poate avea deschise mai multe conturi bancare avand aceeasi valuta. Perechile de valori (**nume_client**, **sold_total**) sunt salvate intr-un vector. Vectorul si dimensiunea acestuia se returnează în **main()** prin tipul de retur sau lista de parametri ai funcției.

Cerințe de implementare:

- Definire funcție cu parametri de I/O definiți complet și corect. (0,25p)
- Determinare valori (**nume_client**, **sold_total**) pentru conturile bancare obtinute la cerinta 2). (0,75p)
- Implementare logică de creare vector cu valori (**nume_client**, **sold_total**). (1,25p)
- Populare completă și corectă a vectorului. (0,25p)
- Testare implementare prin apel de functie si afisare la consola a rezultatului obtinut la apel. (0,25p)

5. Scrieți și apelați funcțiile care dezalocă structurile **Tabelă de Dispersie** si **3xVectori** precum și toate structurile auxiliare utilizate în implementarea cerințelor (dacă este cazul).

Cerințe de implementare:

- Definire funcții cu parametri de I/O definiți complet și corect. (0,15p)
- Absență memory leaks. (0,15p)
- Actualizare variabile de gestionare a structurilor în funcția **main()**. (0,20p)
- Implementare logică de dezalocare a structurilor de date. (0,30p)
- Testare implementare, dezalocare completă și corectă a structurilor prin apel de functii si afisare la consola a rezultatelor obtinute la apel. (0,20p)
- Absență dezalocări structuri auxiliare utilizate. (-0,20p)

MENTIUNI:

- Proiectele cu erori de compilare nu vor fi evaluate.
- Implementările nu trebuie să conțină variabile definite la nivel global sau variabile statice.
- Implementările nu trebuie să conțină structuri predefinite (ex STL, 3rd party libraries etc).
- Implementările plagiate vor fi evaluate cu 0 puncte, indiferent de sursă.
- Toate cerințele trebuie apelate și demonstrate în funcția main() pentru a fi evaluate.
- Art. 72 (1) Pentru următoarele fapte, studenții vor fi exmatriculați fără drept de reînmatriculare în Academia de Studii Economice din București:
 - (c) încercarea de promovare prin fraudă a examenelor sau a altor evaluări;