Bilet #6 / (150 min) 11.06.2021

Implementați o aplicație în limbajul C care implementează soluții la probleme de gestionare a comenzilor online ale unui restaurant.

Scrieţi secvenţa de cod sursă pentru crearea unei structuri de date de tip Arbore binar de cautare ce conţine date
aferente comenzilor de mancare. Cheia de căutare utilizată este id_comanda. Inserarea unei comenzi se
implementează într-o funcţie care se apelează în secvenţa de creare a structurii Arbore binar de cautare. Structura
Arbore binar de cautare conţine minim 10 comenzi incarcate in aplicatie dintr-un fisier de intrare.
Structura Comanda se va defini astfel încât să contină minim 7 câmpuri, astfel: timp livrare (int), cod client (int),

id comanda (int): celelalte 4 campuri sunt definite la alegere, din care minim unul este de tip char*.

Cerințe de implementare:

- Definire structură Comanda. (0,25p)
- String-urile preluate din fișier trebuie să accepte prezența simbolului blank. (0,25p)
- Absentă memory leaks. (0,25p)
- Implementare logică de creare structură Arbore binar de cautare. (0,75p)
- Populare completă și corectă a structurii Arbore binar de cautare cu date de intrare din fisier. (0,25p)
- Testare implementare cu afisarea la consola a continutului structurii Arbore binar de cautare. (0,25p)
- 2. Scrieţi şi apelaţi funcţia pentru determinarea comenzilor din structura creată la cerinta 1) care au timpul de livrare mai mare decat o valoare specificata ca parametru de intrare al functiei. Comenzile identificate sunt salvate într-un vector şi NU partajează zone de memorie heap cu structura Arbore binar de cautare. Vectorul se returnează în main() prin tipul de retur sau lista de parametri ai funcţiei.

Cerințe de implementare:

- Definire functie cu parametri de I/O definiti complet și corect. (0,25p)
- Realizare deep-copy a comenzilor în vector. (0,25p)
- Implementare logică de determinare și salvare a comenzilor 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. Scrieţi şi apelaţi funcţia pentru determinarea comenzilor cu cea mai mare prioritate de servire din Arborele binar de cautare. Implementarea presupune copierea comenzilor intr-o structura Heap, unde prioritatea este data de timpul de livrare. Arborele si structura Heap NU partajează zone de memorie. Structura Heap se returnează în main() prin tipul de retur sau lista de parametri ai funcţiei.

Cerințe de implementare:

- Definire functie cu parametri de I/O definiti complet și corect. (0,25p)
- Implementare mecanism filtrare *Heap*. (0,50p)
- Implementare inserare element in Heap. (0,50p)
- Implementare extragere element din Heap. (0,50p)
- Populare completă și corectă a structurii *Heap*. (0,25p)
- Testare implementare prin apel de functie si afisare la consola a rezultatului obtinut la apel. (0,25p)
- 4. Scrieţi şi apelaţi funcţia pentru determinarea valorii totale a comenzilor la nivel de client. Se iau in considerare comenzile salvate in structura **Arbore binar de cautare** de la la cerinta 1). Un client poate avea mai multe comenzi la restaurant. Perechile de valori (**cod_client, suma_totala**) sunt salvate intr-un vector. Vectorul si dimensiunea acestuia se returnează în **main()** prin tipul de retur sau lista de parametri ai funcţiei.

Cerinţe de implementare:

- Definire funcție cu parametri de I/O definiți complet și corect. (0,25p)
- Determinare valori (cod client, suma totala) pentru comenzile obtinute pe baza structurii de la cerinta 1). (0.75p)
- Implementare logică de creare vector cu valori (cod_client, suma_totala). (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. Scrieţi şi apelaţi funcţiile care dezalocă structurile *Arbore binar de cautare, Heap* si 2 x Vectori precum şi toate structurile auxiliare utilizate în implementarea cerinţelor (dacă este cazul).

Cerințe de implementare:

- Definire funcții cu parametri de I/O definiți complet și corect. (0,15p)
- Absentă 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 şi corectă a structurilor prin apel de functii si afisare la consola a rezultatelor obtinute la apel. (0,20p)
- Absenţă dezalocări structuri auxiliare utilizate. (-0,20p)

Bilet #6 / (150 min) 11.06.2021

MENTIUNI:

- Proiectele cu erori de compilare nu vor fi evaluate.
- Implementările nu trebuie să conțină variabile definite la nivel global sau statice.
- Implementările plagiate vor fi evaluate cu 0 puncte, indiferent de sursă.
- Toate cerintele trebuie apelate și demonstrate in functia main() pentru a fi evaluate.
- Art. 72 (1) Pentru următoarele fapte, studenţii vor fi exmatriculaţi fără drept de reînmatriculare în Academia de Studii Economice din Bucureşti:
 - (c) încercarea de promovare prin fraudă a examenelor sau a altor evaluări;