Setul 6 – Tablouri şi funcţii. Alocare dinamică de memorie

În enunţuri, **„vector”** corespunde unui tablou unidimensional iar **„matrice”** unui tablou bidimensional. Enunţurile şi părţile de enunţuri notate cu * sunt considerate ca având grad de dificultate mai ridicat. **Se recomandă ca problemele 3, 4, 5, 6, 8, 10, 13, 15 şi 18 (alocare dinamică de memorie) să se rezolve în timpul orelor de laborator.**

1. Fiind dat un vector cu elemente numere **întregi**, să se creeze şi să se utilizeze într-un program C o funcţie care, folosind vectorul ca parametru, să calculeze şi să transmită ca rezultat către *main()* câte perechi de elemente consecutive identice există în vector. Exemplu: pentru valorile 2 3 3 3 5 7 7 9 există 3 perechi ce îndeplinesc cerinţa.

2. Scrieţi şi utilizaţi într-un program C o funcţie care să determine valoarea maximă a elementelor impare ce fac parte dintr-un vector de numere **naturale**, folosit ca parametru. Numărul de elemente din vector şi valorile acestuia se citesc de la tastatură, în afara funcţiei. Exemplu: pentru 3 6 1 7 2 rezultatul este 7.

3. Scrieţi o funcție C care numără câte elemente pozitive, câte elemente negative și câte elemente nule se află într-un vector cu **numere întregi**, primit ca parametru. Cele 3 rezultate vor fi transmise către funcția *main(),* din care vor fi afișate, pe ecran.

4. Scrieţi o funcție C pentru extragerea elementelor comune din 2 vectori (neordonați), primiţi ca parametri, într-un al treilea vector. Se va ţine cont de faptul că cei 2 vectori analizaţi pot avea număr diferit de elemente şi pot, la limită, să nu aibă nici un element comun. Afişaţi din *main()* vectorul rezultat, dacă a putut fi creat.

5. Scrieţi o funcție C care să determine poziţiile pe care se află valoarea minimă dintr-un vector cu elemente de **tip întreg**, primit ca parametru, şi să le înscrie într-un alt vector, ce va fi afişat din *main()* după încheierea execuţiei funcţiei.

6. Scrieţi o funcție C care să elimine toate apariţiile primului element dintr-un vector cu valori de **tip *int*.** Dimensiunea şi valorile iniţiale ale vectorului se citesc de la tastatură, în *main(),* iar valorile vectorului modificat de către funcţie se afişează din *main()*, după încheierea execuţiei funcţiei (după finalizarea apelului). Exemplu: dacă, iniţial, vectorul are şase elemente, -3 4 -5 -3 -3 6, la final el va avea doar trei, adică 4 -5 6.

7. Fiind dat un vector cu elemente numere **naturale**, să se creeze şi să se utilizeze o funcţie care să construiască alţi doi vectori: primul va conţine numai elementele pare, iar al doilea numai elementele impare ale vectorului iniţial. Să se afişeze apoi din *main()* vectorii astfel construiţi.

8. Să se scrie o funcţie C care să permute circular la dreapta componentele unui vector. Să se utilizeze funcţia într-un program C pentru a permuta circular la dreapta componentele fiecărei linii **i** a unei matrice cu exact **i** poziţii. Dimensiunile şi valorile matricei se citesc de la tastatură. Matricea astfel modificată va fi afişata din *main()*. Variantă: permutare circulară la stânga.

9. (Problema 2/pag.164 **[A]**) Fie **a** o matrice cu **m** linii şi **n** coloane, cu elemente de tip real. Să se determine linia **L** şi coloana **C** pe care suma elementelor este maximă.

10. Scrieţi o funcţie C care să verifice identitatea a 2 vectori (ca număr de componente şi ca valori). Folosiţi funcţia într-un program C pentru a verifica dacă 2 matrice sunt identice. Dimensiunile şi valorile elementelor celor 2 matrice se citesc de la tastatură.

11. Creaţi şi utilizaţi într-un program C o funcţie care să citească de la tastatură dimensiunile şi valorile elementelor unei matrice. Procedaţi în mod similar pentru un vector ce conţine valori de acelaşi tip ca ale matricei. Folosiţi apoi funcţia creată în Problema 10 pentru a verifica dacă vectorul coincide cu una sau mai multe dintre liniile matricei. Afişaţi din *main()* un mesaj lămuritor (de exemplu, numerele liniilor respective). Variantă *: dacă vectorul nu coincide cu nici una dintre liniile matricei, scrieţi o funcţie care să înlocuiască o linie a matricei (indicată ca parametru) cu vectorul dat, dacă acest lucru este posibil, şi folosiţi-o în program.

12. Scrieţi o funcţie C care să realizeze interschimbarea valorilor elementelor a 2 vectori de aceeaşi dimensiune *(exemplele de la curs...)*. Utilizaţi apoi funcţia pentru interschimbarea valorilor a 2 linii şi a 2 coloane ale unei matrice. Dimensiunile şi valorile elementelor matricei se citesc de la tastatură. Numărul liniilor şi al coloanelor implicate în interschimbare, de asemenea (exemplu: liniile 0 şi 2; coloanele 0 şi 1).

13. De la tastatură se citeşte o matrice **a** de valori întregi, cu **m** linii şi **n** coloane. Să se scrie o funcţie C care să ordoneze crescător elementele unui tablou unidimensional (vector) şi să se utilizeze funcţia pentru a ordona crescător elementele de pe fiecare linie a matriciei. Să se afişeze apoi din *main()* matricea astfel transformată.

14. Variantă a Problemei 13: să se realizeze ordonarea descrescătoare a elementelor de pe liniile cu număr par; (sau) * să se realizeze ordonarea descrescătoare a elementelor pare de pe liniile cu număr par.

15. De la tastatură se citeşte o matrice **a** de valori întregi, cu **m** linii şi **n** coloane. Să se ordoneze liniile matricei crescător după suma elementelor pe care le conţin. Afisaţi matricea după realizarea ordonării cerute. Pentru scrierea programului se recomandă crearea şi utilizarea unor funcţii C adecvate.

16. De la tastatură se citesc dimensiunile şi valorile unei matrice. Se cere să se elimine din matrice liniile care nu au elementele ordonate strict crescător sau strict descrescător şi să se afișeze apoi matricea rezultată. Pentru rezolvare, să se creeze şi să se utilizeze funcţii C adecvate. Sugestie: puteţi crea o nouă matrice, ca rezultat, sau puteţi genera rezultatul modificând matricea iniţială *. Exemplu numeric:

Iniţial:	Rezultat:
4 4 0 3	6 9 11 13
2 1 3 4	6 5 4 2
6 9 11 13	
6 5 4 2	
4 3 2 44	

17. * În funcţia *main()* a unui program C, folosiţi ***calloc()*** pentru a aloca dinamic spaţiu de memorie unui vector cu **m** valori numere reale (**m** citit de la tastatură). Cu ajutorul unei funcţii C create în acest scop, citiţi valorile vectorului de la tastatură. Apoi afişaţi în *main()* valorile vectorului. La sfârşit, eliberaţi memoria folosind functia ***free().*** Ce concluzii puteţi formula, cu referire la modul în care este folosită zona de memorie *heap* (în care se face alocare dinamică) de către funcţiile ce compun un program C?

18. * Alocaţi dinamic memorie pentru doi vectori şi citiţi de la tastatură valorile componentele lor cu ajutorul unei funcţii C special create. Scrieţi şi utilizaţi în program o a doua funcţie care să realizeze adăugarea valorilor vectorului „mai mic” (care are cel mai mic număr de valori) după cele deja existente în vectorul „mai mare”, cu redimensionarea acestuia – folosind ***realloc()***. Afişaţi din *main()* vectorul astfel modificat şi nu uitaţi ca, înainte de finalul programului, să eliberaţi spaţiul de memorie ce fusese alocat în mod dinamic.

19. Se recomandă şi rezolvarea **problemelor 7, 8, 9 şi 11** de la pag. 165 din **[A]** (adică din „Programarea calculatoarelor. Teorie si aplicatii. Partea I” - Daniela Saru, Ştefan Mocanu - Editura Printech, 2012).