

Setul 6 – Tablouri și funcții. Alocare dinamică de memorie

În enunțuri, „**vector**” corespunde unui tablou unidimensional iar „**matrice**” unui tablou bidimensional. Enunțurile și părțile de enunțuri notate cu * sunt considerate ca având grad de dificultate mai ridicat. **Se recomandă ca problemele 3, 4, 5, 6, 8, 10, 13, 15 și 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 și să se utilizeze într-un program C o funcție care, folosind vectorul ca parametru, să calculeze și 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 cerința.
2. Scrieți și utilizați într-un program C o funcție 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 și valorile acestuia se citesc de la tastatură, în afara funcției. Exemplu: pentru 3 6 7 1 2 rezultatul este 7.
3. Scrieți 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. Scrieți o funcție C pentru extragerea elementelor comune din 2 vectori (neordonați), primiți ca parametri, într-un al treilea vector. Se va ține cont de faptul că cei 2 vectori analizați pot avea număr diferit de elemente și pot, la limită, să nu aibă nici un element comun. Afișați din *main()* vectorul rezultat, dacă a putut fi creat.
5. Scrieți o funcție C care să determine pozițiile pe care se află valoarea minimă dintr-un vector cu elemente de **tip întreg**, primit ca parametru, și să le înscrie într-un alt vector, ce va fi afișat din *main()* după încheierea execuției funcției.
6. Scrieți o funcție C care să elimine toate aparițiile primului element dintr-un vector cu valori de **tip int**. Dimensiunea și valorile inițiale ale vectorului se citesc de la tastatură, în *main()*, iar valorile vectorului modificat de către funcție se afișează din *main()*, după încheierea execuției funcției (după finalizarea apelului). Exemplu: dacă, inițial, vectorul are șase 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 și să se utilizeze o funcție C care să construiască alți doi vectori: primul va conține numai elementele pare, iar al doilea numai elementele impare ale vectorului inițial. Să se afișeze apoi din *main()* vectorii astfel construiți.
8. Să se scrie o funcție C care să permute circular la dreapta componentele unui vector. Să se utilizeze funcția într-un program C pentru a permuta circular la dreapta componentele fiecărei linii *i* a unei matrice cu exact *i* poziții. Dimensiunile și valorile matricei se citesc de la tastatură. Matricea astfel modificată va fi afișată din *main()*. Variantă: permutare circulară la stânga.
9. (Problema 2/pag.164 [A]) Fie **a** o matrice cu **m** linii și **n** coloane, cu elemente de tip real. Să se determine linia **L** și coloana **C** pe care suma elementelor este maximă.
10. Scrieți o funcție C care să verifice că 2 vectori sunt sau nu identici (ca număr de componente și ca valori). Folosiți funcția într-un program C pentru a verifica dacă 2 matrice sunt identice. Dimensiunile și valorile elementelor celor 2 matrice se citesc de la tastatură.
11. Creați și utilizați într-un program C o funcție care să citească de la tastatură dimensiunile și valorile elementelor unei matrice. Procedați în mod similar pentru un vector ce conține valori de același tip ca ale matricei. Folosiți apoi funcția creată în Problema 10 pentru a verifica dacă vectorul coincide cu una sau mai multe dintre liniile matricei. Afișați din *main()* un mesaj lămuritor (de exemplu, numerele liniilor respective). Variantă *: dacă vectorul nu coincide cu nici una dintre liniile matricei, scrieți o funcție C care să înlocuiască o linie a matricei (indicată ca parametru) cu vectorul dat, dacă acest lucru este posibil, și folosiți-o în program.
12. Scrieți o funcție C care să realizeze interschimbarea valorilor elementelor a 2 vectori de aceeași dimensiune (*au fost prezentate exemple la curs...*). Utilizați apoi funcția pentru interschimbarea valorilor a 2 linii și a 2 coloane ale unei matrice. Dimensiunile și valorile elementelor matricei se citesc de la

tastatură. Numărul liniilor și al coloanelor implicate în interschimbare, de asemenea (exemplu: liniile 0 și 2; coloanele 0 și 1).

13. De la tastatură se citește o matrice **a** de valori întregi, cu **m** linii și **n** coloane. Să se scrie o funcție C care să ordoneze crescător elementele unui tablou unidimensional (vector) și să se utilizeze funcția pentru a ordona crescător elementele de pe fiecare linie a matriciei. Să se afișeze 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 citește o matrice **a** de valori întregi, cu **m** linii și **n** coloane. Să se ordoneze liniile matriciei crescător după suma elementelor pe care le conțin. Afișați matricea după realizarea ordonării cerute. Pentru scrierea programului se recomandă crearea și utilizarea unor funcții C adecvate.
16. De la tastatură se citesc dimensiunile și valorile unei matrice. Se cere să se elimine din matrice liniile care nu au elementele ordonate strict crescător sau strict descrescător și să se afișeze apoi matricea rezultată. Pentru rezolvare, să se creeze și să se utilizeze funcții C adecvate. Sugestie: puteți crea o nouă matrice, ca rezultat, sau puteți genera rezultatul modificând matricea inițială *. Exemplu numeric:

Inițial:	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 funcția *main()* a unui program C, folosiți **calloc()** pentru a alocă dinamic spațiu de memorie unui vector cu **m** valori numere reale (**m** citit de la tastatură). Cu ajutorul unei funcții C create în acest scop, citiți valorile vectorului de la tastatură. Apoi afișați în *main()* valorile vectorului. La sfârșit, eliberați memoria folosind funcția **free()**. Ce concluzii puteți formula, cu referire la modul în care este folosită zona de memorie *heap* (în care se face alocare dinamică) de către funcțiile ce compun un program C?
18. * Alocați dinamic memorie pentru doi vectori și citiți de la tastatură valorile componentele lor cu ajutorul unei funcții C special create. Scrieți și utilizați în program o a doua funcție 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()**. Afișați din *main()* vectorul astfel modificat și nu uitați ca, înainte de finalul programului, să eliberați spațiul de memorie ce fusese alocat în mod dinamic.
19. Se recomandă și rezolvarea **problemelor 7, 8, 9 și 11** de la pag. 165 din **[A]** (adică din „Programarea calculatoarelor. Teorie și aplicații. Partea I” - Daniela Saru, Ștefan Mocanu - Editura Printech, 2012).