1. **TABLOURI UNIDIMENSIONALE**
	1. **SCOPUL LUCRĂRII**

Scopul lucrării este acela de a familiariza studentul cu citirea, utilizarea şi afişarea vectorilor (tablourilor unidimensionale). Lucrarea este şi o continuare a lucrării precedente, dedicate utilizării ciclului for.

* 1. **BREVIAR TEORETIC**

Tablourile sunt colecţii de date de acelaşi tip reunite sub un singur nume, care ne permit să programăm mai uşor operaţii asupra grupurilor de valori de acelaşi tip.

Declararea tabloului este similară cu declaraţia unei variabile simple, cu o

singură excepţie: trebuie declarată şi dimensiunea tabloului. Numărul de componente se declară între [ ]. Declaraţia

*float date[50];*

creează un tablou cu 50 de elemente tip float.

Primul element are indicele 0, al doilea are indicele 1, …, iar ultimul are indicele 49.

Un tablou unidimensional (numit de foarte multe ori vector) este deci o colecţie structurată de elemente care pot fi accesate individual, specificând poziţia componentei printr-un indice (variabilă de tip întreg).

Sintaxa unei declaraţii de tablou unidimensional (vector):

*TipDată NumeTablou*[*ExpresieConstInt*];

Elementele unui tablou pot avea aproape orice tip de dată.

Expresia dintre parantezele drepte este o constantă întreagă, ce trebuie să fie strict mai mare decât 0 şi determină numărul de componente ale tabloului. Dacă valoarea este n, domeniul indicilor va fi între 0 şi n-1, deci vectorul are n elemente.

Pentru a avea acces la componentele individuale ale unui vector, scriem numele vectorului urmat de o expresie indice între [ ]. Expresia specifică numărul componentei accesate şi poate fi o constantă întreagă, o variabilă întreagă, sau o expresie care este evaluată la o valoare întreagă. Oricare ar fi, însă, forma indicelui, acesta trebuie să fie o valoare întreagă.

Fiecare componentă a unui tablou poate fi tratată exact ca o variabilă simplă.

Elementele unui tablou pot fi iniţializate în instrucţiunea de declarare prin adăugarea unei liste de valori separate prin virgulă, plasate între acolade.

Exemplu

*#include <iostream.h>*

*#include <conio.h>*

*void main( ){*

 *float x[5] = {11.2, 3.172, 12.1, -27.3, 12};*

 *int i ;*

 *for(i = 0 ; i <5 ; i++)*

 *cout << "x[" << i << "]= " << x[i] << endl ;*

*}*

Dacă se înscriu mai multe valori, compilatorul semnalează o eroare. Dacă sunt mai puţine valori în listă, restul sunt iniţializate cu valoarea 0.

Tablourile sunt transmise funcţiilor prin referinţă.

În lista parametrilor formali, declararea unui tablou nu include şi dimensiunea sa

între [ ]. Dacă se include dimensiunea, compilatorul o ignoră, deoarece compilatorului îi este necesară doar informaţia referitoare la natura parametrului, adică faptul că este vorba despre un tablou, şi la tipul componentelor sale. Acesta este motivul pentru care trebuie adăugat un al doilea parametru al funcţiei prin care se precizează numărul de componente.

În prototipul unei funcţii care are parametri de tip tablou nu este necesară prezenţa numelor parametrilor formali.

* 1. **EXEMPLE**
1. ***Citirea elementelor unui vector de la tastatura***

*#include <conio.h>*

*#include <iostream.h>*

*void main(){*

 *clrscr();*

 *int n, i, v[100], s=0;*

 *// citirea unui vector de la tastatura*

 *cout<<"numarul de elemente? "; cin>>n;*

 *for(i=1; i<=n; i++){*

 *cout<<"v["<<i<<"]=? "; cin>>v[i];*

 *}*

 *cout<<endl;*

 *// afisarea vectorului initial*

 *cout<<"vectorul initial:"<<endl;*

 *for(i=1; i<=n; i++) cout<<v[i]<<" ";*

 *cout<<endl<<endl;*

 *// prelucrarea elementelor*

 *for(i=1; i<=n; i++) v[i]=v[i]\*v[i];*

*// afisarea vectorului final*

 *cout<<"vectorul final:"<<endl;*

 *for(i=1; i<=n; i++) cout<<v[i]<<" ";*

 *cout<<endl<<endl;*

 *getch();*

*}*

1. ***Citirea unui vector dintr-un fisier (cazul în care se ştie numărul elementelor – este înscris pe prima linie din fişier)***

*#include <conio.h>*

*#include <iostream.h>*

*#include <fstream.h>*

*void main(){*

 *clrscr();*

 *int n, i, x[200];*

 *float s, med;*

 *ifstream f("fisier1.in");*

 *f>>n;*

 *cout<<"voi citi "<<n<<" numere"<<endl;*

 *s=0;*

 *for(i=1; i<=n; i++){*

 *f>>x[i];*

 *cout<<x[i]<<" ";*

 *s=s+x[i];*

 *}*

 *cout<<endl;*

 *med=s/n;*

 *cout<<med;*

 *getch();*

*}*

***Ex. Pentru fişierul “fisier1.in” avand continutul***

8

15 23 2 0 4 35 11 61

***Se va obţine rezultatul***



1. ***Citirea unui vector dintr-un fisier (cazul în care NU se ştie numărul elementelor)***

*#include <conio.h>*

*#include <iostream.h>*

*#include <fstream.h>*

*void main(){*

 *clrscr();*

 *int n, i, x[200];*

 *float s, med;*

 *ifstream f("fisier2.in");*

 *s=0;*

 *i=1;*

 *while(f>>x[i]){*

 *cout<<x[i]<<" ";*

 *s=s+x[i];*

 *i++;*

 *}*

 *n=i-1;*

 *cout<<endl;*

 *cout<<"n="<<n<<endl;*

 *med=s/n;*

 *cout<<med<<endl;*

 *getch();*

*}*

***Ex. Pentru fişierul “fisie21.in” avand continutul***

15 23 2 0 4 35 11 61

***Se va obţine rezultatul***

******

1. ***Construcţia unui vector w dintr-un vector v***

*#include <conio.h>*

*#include <iostream.h>*

*void main(){*

 *clrscr();*

 *int n, i, v[100], w[100], s=0;*

 *// citirea vectorului v de la tastatura*

 *cout<<"numarul de elemente? "; cin>>n;*

 *for(i=1; i<=n; i++){*

 *cout<<"v["<<i<<"]=? "; cin>>v[i];*

 *}*

 *cout<<endl;*

 *// afisarea vectorului v*

 *cout<<"vectorul v: "<<endl;*

 *for(i=1; i<=n; i++) cout<<v[i]<<" ";*

 *cout<<endl<<endl;*

 *// constructia vectorului w*

 *for(i=1; i<=n; i++) w[i]=v[i]\*v[i];*

 *// afisarea vectorului w*

 *cout<<"vectorul w: "<<endl;*

 *for(i=1; i<=n; i++) cout<<w[i]<<" ";*

 *cout<<endl;*

 *getch();*

*}*

1. ***Calcul pe totalitatea elementelor unui vector (suma elementelor impare)***

*#include <conio.h>*

*#include <iostream.h>*

*void main(){*

 *clrscr();*

 *int n, i, v[100], s=0;*

 *// citirea unui vector de la tastatura*

 *cout<<"numarul de elemente? "; cin>>n;*

 *for(i=1; i<=n; i++){*

 *cout<<"v["<<i<<"]=? "; cin>>v[i];*

 *}*

 *cout<<endl;*

 *// afisarea vectorului initial*

 *cout<<"vectorul initial:"<<endl;*

 *for(i=1; i<=n; i++) cout<<v[i]<<" ";*

 *cout<<endl<<endl;*

 *// prelucrarea elementelor*

 *for(i=1; i<=n; i++) if(v[i]%2!=0) s=s+v[i];*

 *cout<<"suma elementelor impare din vector este "<<s;*

 *getch();*

*}*

1. ***Cautarea unei valori intr-un vector***

*#include <conio.h>*

*#include <iostream.h>*

*void main(){*

 *clrscr();*

 *int n, i, v[100], k, pos=0;*

 *// citirea vectorului de la tastatura*

 *cout<<"numarul de elemente? "; cin>>n;*

 *for(i=1; i<=n; i++){*

 *cout<<"v["<<i<<"]=? "; cin>>v[i];*

 *}*

 *cout<<endl;*

 *// afisarea vectorului initial*

 *cout<<"vectorul initial:"<<endl;*

 *for(i=1; i<=n; i++) cout<<v[i]<<" ";*

 *cout<<endl<<endl;*

 *cout<<"ce valoare caut in vecvtor? "; cin>>k;*

 *i=1;*

 *while(i<=n && pos==0){*

 *if(v[i]==k) pos=i;*

 *i++;*

 *}*

 *if(pos!=0) cout<<"am gasit valoarea "<<k<<" pe pozitia "<<k;*

 *else cout<<"nu am gasit valoarea "<<k<<" in vector";*

 *getch();*

*}*

1. ***Aflarea valorilor maxime şi minime şi a poziţiilor pe care acestea le ocupă într-un vector***

*#include <conio.h>*

*#include <iostream.h>*

*void main() {*

 *clrscr();*

 *int v[50], i, n, max, pozmax, min, pozmin;*

 *cout<<"n=?";*

 *cin>>n;*

 *for(i=1;i<=n;i++){*

 *cout<<"v["<<i<<"]=?";*

 *cin>>v[i];*

 *}*

 *max=v[1];pozmax=1;*

 *min=v[1];pozmin=1;*

 *for(i=2;i<=n;i++){*

 *if(v[i]>max){*

 *max=v[i];pozmax=i;*

 *}*

 *if(v[i]<min){*

 *min=v[i];pozmin=i;*

 *}*

 *}*

 *cout<<"maximul este "<<max<<" si ocupa pozitia "<<pozmax<<endl;*

 *cout<<"minimul este "<<min<<" si ocupa pozitia "<<pozmin;*

 *getch();*

*}*

1. ***Ordonarea crescătoare a elementelor unui vector, prin selecţie directă***

Comparăm primul element cu toate elementele care urmează după el. Dacă găsim un element mai mic decât primul, atunci le interschimbăm pe cele două. Apoi continuăm cu al doilea element al şirului, pe care, de asemenea îl comparăm cu toate elementele care urmează după el şi în caz de inversiune interschimbăm cele două elemente. Apoi procedăm la fel cu al treilea element al şirului iar procesul continuă astfel pâna la penultimul element al şirului care va fi comparat cu ultimul element din şir.

*#include <conio.h>*

*#include <iostream.h>*

*void main() {*

 *clrscr();*

 *int v[50], i, j, n, aux;*

 *cout<<"n=?";*

 *cin>>n;*

 *for(i=1;i<=n;i++){*

 *cout<<"v["<<i<<"]=?";*

 *cin>>v[i];*

 *}*

 *for(i=1;i<=n-1;i++){*

 *for(j=1;j<=i+1;j++) if(v[i]>v[j]){*

 *aux=v[i];*

 *v[i]=v[j];*

 *v[j]=aux;*

 *}*

 *}*

 *for(i=1;i<=n;i++) cout<<"v[i"<<"]="<<v[i]<<endl;*

 *getch();*

*}*

1. ***Ordonarea crescătoare a elementelor unui vector, prin metoda bulelor***

Acest algoritm se mai numeşte şi "sortarea prin selecţie şi interschimbare", "sortarea rin propagare". Succesul algoritmului este asigurat de trecerea succesivă prin tablou, până când acesta este sortat, cu specificaţia că, la fiecare trecere, elementele succesive i şi i+1 pentru are v[i] > v[i+1], vor fi interschimbate. Metoda poate fi imbunătăţită dacă, după fiecare trecere, se va reţine ultima poziţie din tablou în care a avut loc o interschimbare, iar trecerea următoare se va efectua doar până la acea poziţie. În cazul în care la o trecere nu a avut loc nici o interschimbare, algoritmul se va incheia. Pentru o şi mai bună optimizare, se poate înlocui trecerea prin tablou într-un sens cu trecerea în dublu sens. În acest caz, dacă la două treceri succesive, între două elemente i şi i+1 nu a avut loc o interschimbare, atunci nici la trecereile următoare nu se vor mai înregistra interschimbări.

*#include <conio.h>*

*#include <iostream.h>*

*void main() {*

 *clrscr();*

 *int v[50], i, j, n, aux;*

 *cout<<"n=?";*

 *cin>>n;*

 *for(i=1;i<=n;i++){*

 *cout<<"v["<<i<<"]=?";*

 *cin>>v[i];*

 *}*

 *for(i=1;i<=n-1;i++){*

 *for(j=1;j<=n-i;i++) if(v[j]>v[j+1]){*

 *aux=v[j];*

 *v[j]=v[j+1];*

 *v[j+1]=aux;*

 *}*

 *}*

 *for(i=1;i<=n;i++) cout<<"v[i"<<"]="<<v[i]<<endl;*

 *getch();*

*}*

1. ***Deplasarea la dreapta a unui vector, cu insertia pe prima poziţie a valorii k***

*#include <conio.h>*

*#include <iostream.h>*

*#include <string.h>*

*void main(){*

 *clrscr();*

 *int n, i, v[30], k;*

 *cout<<"n=? "; cin>>n;*

 *for(i=1; i<=n; i++){*

 *cout<<"v["<<i<<"]=? ";*

 *cin>>v[i];*

 *}*

 *cout<<"k=? "; cin>>k;*

 *for(i=1; i<=n; i++) cout<<v[i]<<" ";*

 *cout<<endl;*

 *for(i=n+1; i>=1; i--) v[i+1]=v[i];*

 *v[1]=k;*

 *n++;*

 *for(i=1; i<=n; i++) cout<<v[i]<<" ";*

 *cout<<endl;*

 *getch();*

*}*

 Observaţie: în mod asemănător, se poate insera o valoare ***k*** pe orice poziţie ***p***

1. ***Rotirea spre dreapta a unui vector, cu recirculare (elementul de pe poziţia n trece pe poziţia 1)***

*#include <conio.h>*

*#include <iostream.h>*

*#include <string.h>*

*void main(){*

 *clrscr();*

 *int n, i, v[30], aux;*

 *cout<<"n=? "; cin>>n;*

 *for(i=1; i<=n; i++){*

 *cout<<"v["<<i<<"]=? ";*

 *cin>>v[i];*

 *}*

 *for(i=1; i<=n; i++) cout<<v[i]<<" ";*

 *cout<<endl;*

 *aux=v[n];*

 *for(i=n-1; i>=1; i--) v[i+1]=v[i];*

 *v[1]=aux;*

 *for(i=1; i<=n; i++) cout<<v[i]<<" ";*

 *cout<<endl;*

 *getch();*

*}*

1. ***Rotirea spre stânga a unui vector, cu recirculare (elementul de pe poziţia 1 trece pe poziţia n)***

*#include <conio.h>*

*#include <iostream.h>*

*#include <string.h>*

*void main(){*

 *clrscr();*

 *int n, i, v[30], aux;*

 *cout<<"n=? "; cin>>n;*

 *for(i=1; i<=n; i++){*

 *cout<<"v["<<i<<"]=? ";*

 *cin>>v[i];*

 *}*

 *for(i=1; i<=n; i++) cout<<v[i]<<" ";*

 *cout<<endl;*

 *aux=v[1];*

 *for(i=1; i<=n-1; i++) v[i]=v[i+1];*

 *v[n]=aux;*

 *for(i=1; i<=n; i++) cout<<v[i]<<" ";*

 *cout<<endl;*

 *getch();*

*}*

1. ***Numere Fibonacci***

*#include <conio.h>*

*#include <iostream.h>*

*void main(){*

 *clrscr();*

 *unsigned long n, a, b, f[50], j;*

 *int i, fv[50];*

*//afisarea numerelor fibonacci*

 *f[1]=1;*

 *f[2]=2;*

 *for(i=3; i<=46; i++) f[i]=f[i-2]+f[i-1];*

 *for(i=1; i<=46; i++) cout<<i<<" "<<f[i]<<endl;*

*//descompunerea unui numar n in suma de numere fibonacci*

 *getch();*

 *cout<<"n=?";*

 *cin>>n;*

 *for(i=1; i<=46; i++) fv[i]=0;*

 *for(i=46; i>=1;i--) if(f[i]<=n){*

 *n=n-f[i];*

 *cout<<f[i]<<" ";*

 *}*

 *cout<<endl;*

 *getch();*

*//vizualizarea numerelor de la a la b prin componentele fibonacci*

 *cout<<"a=";cin>>a;*

 *cout<<"b=";cin>>b;*

 *for(j=a; j<=b; j++){*

 *n=j;*

 *for(i=1; i<=46; i++) fv[i]=0;*

 *for(i=46; i>=1;i--) if(f[i]<=n){*

 *n=n-f[i];*

 *fv[i]=1;*

 *}*

 *for(i=46; i>=1;i--) cout<<fv[i];*

 *cout<<endl;*

 *}*

 *getch();*

*}*

* 1. **TEME DE LABORATOR**
1. Se citeşte de la tastatură numărul ***n*** de componente, apoi elementele (numere naturale), ale vectorului unidimensional ***v***. Să se afişeze maximul, minimul şi poziţiile pe care acestea le ocupă în vector (numărul ***n*** şi elementele ***v[i]*** vor fi definite ***unsigned long***)
2. Pentru vectorul de la punctul 1, să se afişeze suma tuturor elemetelor pare de pe poziţii impare
3. Să se ordoneze crescător şi să se afişeze vectorul ***v*** de la punctul 1, utilizând metoda interschimbării
4. Să se ordoneze crescător şi să se afişeze vectorul ***v*** de la punctul 1, utilizând metoda bulelor
5. Utilizând vectorul ***v*** de la punctul 1, să se creeze vectorul ***w***, care conţine toate elementele din ***v***, iar între fiecare două elemente successive media aritmetică a acestor elemente
6. Utilizând vectorul ***v*** de la punctul 1, să se creeze vectorul ***w***, care conţine numerele prime din ***v***
7. Se citeşte de la tastatură un număr ***n*** (***unsigned long***); să se creeze şi afişeze vectorul ***w***, care conţine cifrele numărului ***n***
8. Se citeşte de la tastatură un număr ***n*** (***unsigned long***); să se afişeze cel mai mare număr natural care se poate obţine utilizând cifrele numărului ***n***

**BONUS:**

1. Să se rotească circular spre stânga vectorul ***v*** de la punctual 1 cu exact ***k*** poziţii (***k*** număr natural introdus de la tastatură)
2. Implementaţi o metodă eficientă pentru problema de la punctual 9, în cazul în care ***n*** şi ***k*** pot fi numere foarte mari
3. În vectorul v, de la punctul 1, să se intercaleze pe poziţia ***k*** valoarea ***m*** (***k*** şi ***m*** citite de la tastatură)
4. Să se şteargă elementul de pe poziţia ***k*** din vectorul ***v*** (***k*** citit de la tastatură)

**SUPERBONUS**

1. Elementele vectorului ***v*** de la punctul 1 reprezintă elementele unei mulţimi ***A***; să se construiasă vectorul ***w*** care conţine elementele celei mai largi (cu cel mai mare număr de elemnte posibil) submulţimi ***B***, care satisfice condiţia că suma elementlor este divizibilă prin 3.