MINISTERUL EDUCAȚIEI AL REPUBLICII MOLDOVA

UNIVERSITATEA TEHNICĂ A MOLDOVEI

**RAPORT**

LUCRAREA DE LABORATOR NR. 4

LA PROGRAMAREA CALCULATOARELOR

Elaborat: st. gr. TI – 211, Popa Ecaterina

Verificat: Stratulat Ștefan

UTM, Chișinău, 2021

**Probleme pentru tablouri unidimensionale:**

1. Scrieți un program C pentru a citi și a imprima elementele unui tablou. - folosind recursia.

#include <stdio.h>

int main ()

{

int n;

printf ("Introduceti numarul de elemente:");

scanf ("%d", &n);

int citire[n];

printf ("Introduceti elementele:");

for (int i=0; i<n; i++)

{

scanf ("%d", &citire[i]);

}

PrintCitire (n, citire);

}

int PrintCitire(int n , int citire[])

{

if (n==0)

return 0;

else

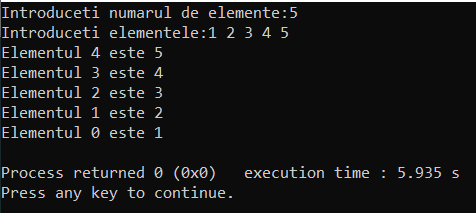
{

printf ("Elementul %d este %d \n", n-1, citire[n-1]);

}

return PrintCitire(n-1, citire);

}



1. Scrieți un program C pentru a imprima toate elementele negative dintr-un tablou.

#include <stdio.h>

int main ()

{

int n, citire[10], count=0;

printf ("Introduceti numarul de elemente:");

scanf ("%d", &n);

printf ("Introduceti elementele:");

for (int i=0; i<n; i++)

{

scanf ("%d", &citire[i]);

}

printf ("Elemente negative = ");

for (int i=0; i<n; i++)

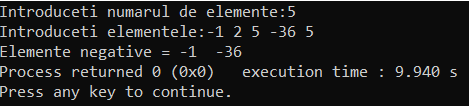
{

if (citire[i]<0)

printf ("%d\t", citire[i]);

}

}



1. Scrieți un program C pentru a găsi suma tuturor elementelor tabloului. - folosind recursia.

#include <stdio.h>

int main ()

{

int n;

printf ("Introduceti numarul de elemente:");

scanf ("%d", &n);

int citire[n];

printf ("Introduceti elementele:");

for (int i=0; i<n; i++)

{

scanf ("%d", &citire[i]);

}

printf ("Suma elementelor = %d", suma(n, citire));

}

int suma(int n , int citire[])

{

if (n==0)

return 0;

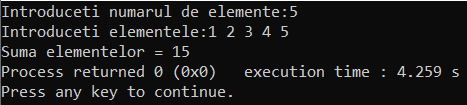
else

{

return suma(n-1, citire)+citire[n-1];

}

}



1. Scrieți un program C pentru a găsi elementul maxim și minim într-un tablou. - folosind recursia.

#include <stdio.h>

int main ()

{

int n;

printf ("Introduceti numarul de elemente:");

scanf ("%d", &n);

int arr[n];

printf ("Introduceti elementele:");

for (int i=0; i<n; i++)

{

scanf ("%d", &arr[i]);

}

printf ("Maxim = %d\n", MaxValue(n, arr, arr[n-1]));

printf ("Minim = %d", MinValue(n, arr, arr[n-1]));

}

int MaxValue(int n , int arr[], int max)

{

if (n==0)

return max;

else if(arr[n-1]<max)

{

return MaxValue(n-1, arr, max);

}

else

{

return MaxValue(n-1, arr, arr[n-1]);

}

}

int MinValue(int n , int arr[], int min)

{

if (n==0)

return min;

else if(arr[n-1]>min)

{

return MinValue(n-1, arr, min);

}

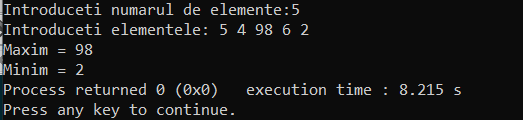
else

{

return MinValue(n-1, arr, arr[n-1]);

}

}



1. Scrieți un program C pentru a găsi al doilea cel mai mare element dintr-un tablou.

#include <stdio.h>

int main ()

{

int n, matrice[10], Max2, temp;

printf ("Introduceti numarul de elemente:");

scanf ("%d", &n);

printf ("Introduceti elementele:");

for (int i=0; i<n; i++)

{

scanf ("%d", &matrice[i]);

}

for (int i=0; i<n; i++)

{

for (int j=i+1; j<n; j++)

{

temp = matrice[i];

matrice[i]=matrice[j];

matrice[j]=temp;

}

printf ("%d\t", matrice[i]);

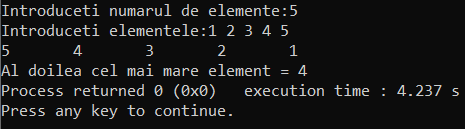
}

printf ("\n");

Max2 = matrice [1];

printf ("Al doilea cel mai mare element = %d", Max2);

}



1. Scrieți un program C pentru a număra numărul total de elemente pare și impare dintr-un tablou.

#include <stdio.h>

int main ()

{

int n, matrice[10], countpare=0, countimpare=0;

printf ("Introduceti numarul de elemente:");

scanf ("%d", &n);

printf ("Introduceti elementele:");

for (int i=0; i<n; i++)

{

scanf ("%d", &matrice[i]);

}

for (int i=0; i<n; i++)

{

if (matrice[i]%2==0)

{

countpare++;

}

if (matrice[i]%2==1)

{

countimpare++;

}

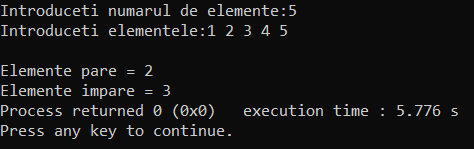
}

printf ("\n");

printf ("Elemente pare = %d\n", countpare);

printf ("Elemente impare = %d", countimpare);

}



1. Scrieți un program C pentru a număra numărul total de elemente negative dintr-un tablou.

#include <stdio.h>

int main ()

{

int n, matrice[10], negative=0;

printf ("Introduceti numarul de elemente:");

scanf ("%d", &n);

printf ("Introduceti elementele:");

for (int i=0; i<n; i++)

{

scanf ("%d", &matrice[i]);

}

for (int i=0; i<n; i++)

{

if (matrice[i]<0)

{

negative++;

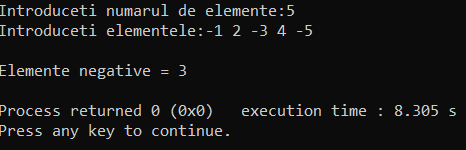
}

}

printf ("\n");

printf ("Elemente negative = %d\n", negative);

}



1. Scrieți un program C pentru a copia toate elementele dintr-un tablou în alt tablou.

#include <stdio.h>

int main ()

{

int n, i, matrice1[10], matrice2[10];

printf ("Introduceti numarul de elemente:");

scanf ("%d", &n);

printf ("Introduceti elementele:");

for (int i=0; i<n; i++)

{

scanf ("%d", &matrice1[i]);

}

printf ("Tablou copiat: \n");

for (int i=0; i<n; i++)

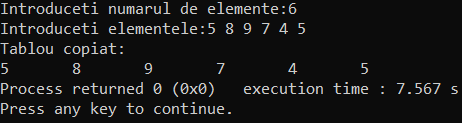
{

matrice2[i]=matrice1[i];

printf ("%d\t", matrice2[i]);

}

}



1. Scrieți un program C pentru a insera un element într-un tablou pe o poziție dată de la tastatură.

#include <stdio.h>

int main ()

{

int n, i;

int poz, num;

int matrice[10]={};

printf ("Introduceti pozitia numarului dorit:");

scanf ("%d", &poz);

printf ("Introduceti numarul dorit:");

scanf ("%d", &num);

matrice[poz]=num;

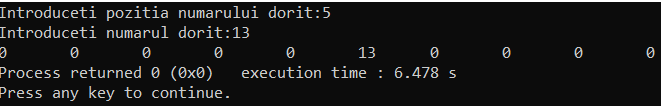
for (int i=0; i<10; i++)

{

printf ("%d\t", matrice[i]);

}

}



1. Scrieți un program C pentru a șterge un element dintr-un tablou în poziția specificată de la tastatură.

#include <stdio.h>

int main ()

{

int n, i;

int poz, num;

int matrice[]={1, 2, 3, 4, 5, 6, 7, 8, 9};

printf ("Introduceti pozitia numarului dorit:");

scanf ("%d", &poz);

matrice[poz-1]=0;

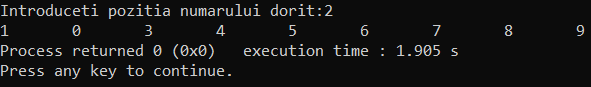
for (int i=0; i<9; i++)

{

printf ("%d\t", matrice[i]);

}

}



1. Scrieți un program C pentru a număra frecvența relativă fiecărui element dintr-un tablou.

#include <stdio.h>

void main ()

{

int n, matrice[100], frecventa[100],k;

printf ("Introduceti numarul de elemente: ");

scanf ("%d", &n);

printf ("Introduceti elementele: ");

for (int i=0; i<n; i++)

{

scanf ("%d", &matrice[i]);

frecventa[i]=-1;

}

printf ("\n");

for (int i=0; i<n; i++)

{

int count = 1;

for (int j=i+1; j<n; j++)

{

if (matrice[i]==matrice[j])

{

count++;

for(k=j; k<n; k++)

{

matrice[k]=matrice[k+1];

}

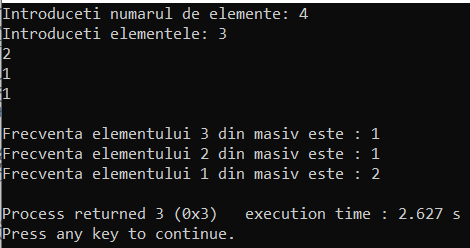
n--;

j--;

}

}

printf("Frecventa elementului %d din masiv este : %d\n",matrice[i],count);}



1. Scrieți un program C pentru a imprima toate elementele unice din tablou.

#include <stdio.h>

void main()

{

int arr[100], n, count=0;

int i, j, k;

printf("Introduceti numarul de elemente: ");

scanf("%d",&n);

printf("Introduceti %d elemente ale vectorului:\n",n);

for(i=0; i<n; i++)

{

printf("Elementul %d: ",i+1);

scanf("%d",&arr[i]);

}

printf("\nElementele unice ale vectorului sunt: ");

for(i=0; i<n; i++)

{

count=0;

for(j=0, k=n; j<k+1; j++)

{

if (i!=j)

{

if(arr[i]==arr[j])

{

count++;

}

}

}

if(count==0)

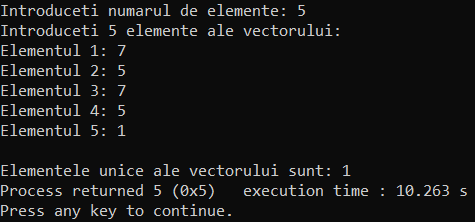
{

printf("%d ",arr[i]);

}

}

}



1. Scrieți un program C pentru a număra numărul total de elemente duplicate dintr-un tablou.

#include <stdio.h>

void main()

{

int arr[100], n, count=0;

int i, j, k;

printf("Introduceti numarul de elemente: ");

scanf("%d",&n);

printf("Introduceti %d elemente ale vectorului:\n",n);

for(i=0; i<n; i++)

{

printf("Elementul %d: ",i+1);

scanf("%d",&arr[i]);

}

for (i=0; i<n; i++)

{

for (j=i+1; j<n; j++)

{

if (arr[i]==arr[j])

{

count++;

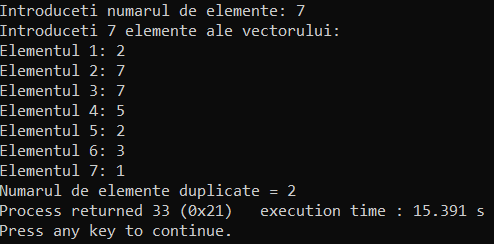
}

}

}

printf ("Numarul de elemente duplicate = %d", count);

}



1. Scrieți un program C pentru a îmbina două tablouri în al treilea tablou.

#include <stdio.h>

void main ()

{

int arr1[100], arr2[100], arr\_fin[100];

int size1, size2, sizefin, i, j, size\_fin;

printf ("Introduceti numarul de elemente pentru primul tablou: ");

scanf ("%d", &size1);

printf ("Introduceti elementele primului tablou \n");

for (i=0; i<size1; i++)

{

scanf ("%d", &arr1[i]);

}

printf ("Introduceti numarul de elemente pentru cel de-al doilea tablou: ");

scanf ("%d", &size2);

printf ("Introduceti elementele celui de-al doilea tablou \n");

for (i=0; i<size2; i++)

{

scanf ("%d", &arr2[i]);

}

size\_fin=size1+size2;

for (i=0; i<size1; i++)

{

arr\_fin[i]=arr1[i];

}

for (i=0, j=size1; j<size\_fin && i<size2; i++, j++)

{

arr\_fin[j]=arr2[i];

}

printf ("Tabloul final \n");

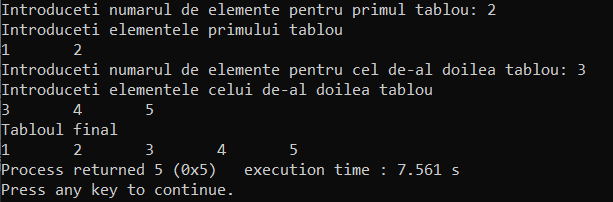
for (i=0; i<size\_fin; i++)

{

printf ("%d\t", arr\_fin[i]);

}

}



1. Scrieți un program C pentru a inversa ordinea elementelor unui tablou.

#include <stdio.h>

void main ()

{

int arr[100];

int size, i, j, temp;

printf ("Introduceti numarul de elemente pentru primul tablou: ");

scanf ("%d", &size);

printf ("Introduceti elementele primului tablou \n");

for (i=0; i<size; i++)

{

scanf ("%d", &arr[i]);

}

for (i=0, j=size-1; i<j; i++, j--)

{

temp=arr[i];

arr[i]=arr[j];

arr[j]=temp;

}

printf ("Tabloul final \n");

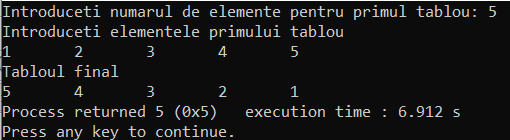
for (i=0; i<size; i++)

{

printf ("%d\t", arr[i]);

}

}



1. Scrieți un program C pentru a pune elemente pare și impare ale unui tablou în două tablouri separate.

#include <stdio.h>

void main ()

{

int array[100], pare[100], impare[100];

int n, num, i, m=0, k=0;

printf ("Introduceti numarul de elemente:");

scanf ("%d", &n);

printf ("Introduceti elementele tabloului initial: ");

for (int i=0; i<n; i++)

{

scanf ("%d", &array[i]);

}

for (i=0; i<n; i++)

{

if (array[i]%2==0)

{

pare[m]=array[i];

m++;

}

else

{

impare[k]=array[i];

k++;

}

}

printf ("Tabloul par: \n");

for (i=0; i<m; i++)

{

printf ("%d\t", pare[i]);

}

printf ("\nTablou impar: \n");

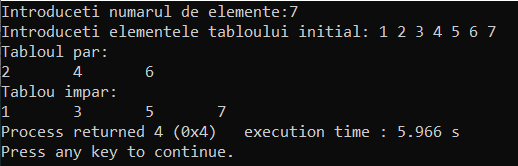
for (i=0; i<k; i++)

{

printf ("%d\t", impare[i]);

}

}



1. Scrieți un program C pentru a căuta un element din tablou și afiza pozițiile găsite.

#include <stdio.h>

void main ()

{

int array[100];

int n, num, i;

printf ("Introduceti numarul de elemente:");

scanf ("%d", &n);

printf ("Introduceti elementele tabloului: ");

for (int i=0; i<n; i++)

{

scanf ("%d", &array[i]);

}

printf ("Introduceti numarul dorit pentru cautare:");

scanf ("%d", &num);

for (i=0; i<n; i++)

{

if (array[i]==num)

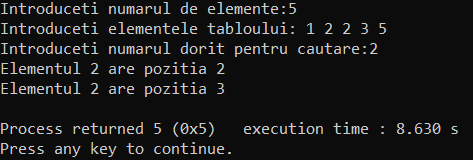
{

printf ("Elementul %d are pozitia %d\n", num, i+1);

}

}

}



1. Scrieți un program C pentru a sorta elementele tabloului în ordine crescătoare sau descrescătoare.

#include <stdio.h>

void main ()

{

int arr[100];

int size, i,j, temp, caz;

printf ("Introduceti numarul de elemente pentru primul tablou: ");

scanf ("%d", &size);

printf ("Introduceti elementele tabloului: \n");

for (i=0; i<size; i++)

{

scanf ("%d", &arr[i]);

}

printf("Pentru ca elementele sa apara:\nIn ordine crescatoare - introduceti 1\nIn ordine descrescatoare - introduceti 2\n");

scanf ("%d", &caz);

switch (caz)

{

case 1:

for(int i=0; i<size; i++)

{

for(int j=i+1; j<size; j++)

{

if(arr[i]<arr[j])

{

temp=arr[i];

arr[i]=arr[j];

arr[j]=temp;

}

}

}

case 2:

for(int i=0; i<size; i++)

{

for(int j=i+1; j<size; j++)

{

if(arr[i]>arr[j])

{

temp=arr[i];

arr[i]=arr[j];

arr[j]=temp;

}

}

}

}

printf ("Tabloul final \n");

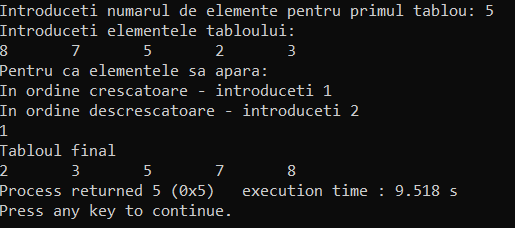
for (i=0; i<size; i++)

{

printf ("%d\t", arr[i]);

}

}



1. Scrieți un program C pentru a roti la stânga un tablou, operația ROL.

#include <stdio.h>

void main ()

{

int arr[100];

int n, i, temp;

printf ("Introduceti numarul de elemente:");

scanf ("%d", &n);

printf ("\n");

printf ("Introduceti elementele tabloului initial: ");

for (int i=0; i<n; i++)

{

scanf ("%d", &arr[i]);

}

temp=arr[0];

for (i=0; i<n; i++)

{

arr[i]=arr[i+1];

}

arr[n-1]=temp;

printf ("\n");

printf ("Tablou rotit la stanga cu o pozitie: ");

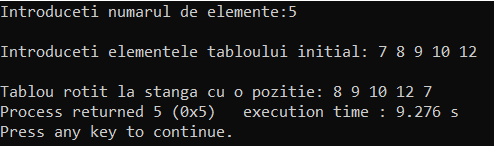
for (int i=0; i<n; i++)

{

printf ("%d\t", arr[i]);

}

}



1. Scrieți un program C pentru a roti dreapta un tablou, operația ROR.

#include <stdio.h>

void main ()

{

int arr[100];

int n, i, temp, temp1;

printf ("Introduceti numarul de elemente:");

scanf ("%d", &n);

printf ("\n");

printf ("Introduceti elementele tabloului initial: ");

for (int i=0; i<n; i++)

{

scanf ("%d", &arr[i]);

}

temp=arr[n-1];

for (i=n-1; i>0; i--)

{

arr[i]=arr[i-1];

}

arr[0]=temp;

printf ("\n");

printf ("Tablou rotit la dreapta cu o pozitie: ");

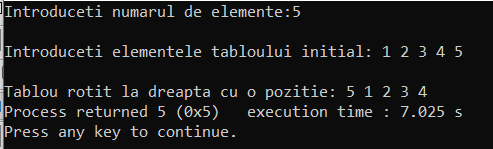
for (int i=0; i<n; i++)

{

printf ("%d ", arr[i]);

}

}



**Probleme pentru tablouri bidimensionale:**

1. Scrieți un program C pentru a aduna două matrice.

#include <stdio.h>

void main ()

{

int i, j;

int row, col;

printf ("Introduceti numarul de randuri pentru ambele matrice: ");

scanf ("%d", &row);

printf ("Introduceti numarul de coloane pentru ambele matrice: ");

scanf ("%d", &col);

int arr1[10][10], arr2[10][10];

printf ("Introduceti elementele primei matrice: \n");

for ( i=0; i<row; i++)

{

for (j=0; j<col; j++)

{

printf ("Elementul %d%d = ",i+1, j+1);

scanf("%d", &arr1[i][j]);

}

}

printf ("Introduceti elementele celei de-a doua matrice: \n");

for (i=0; i<row; i++)

{

for (j=0; j<col; j++)

{

printf ("Elementul %d%d = ",i+1, j+1);

scanf("%d", &arr2[i][j]);

}

}

printf ("\n");

printf ("Matricea suma: \n");

for (int i=0; i<row; i++)

{

for (int j=0; j<col; j++)

{

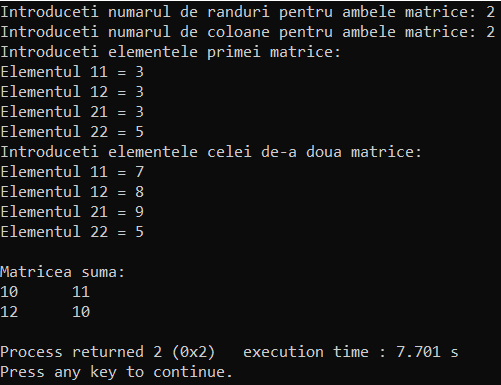
printf("%d\t", arr1[i][j]+arr2[i][j]);

}

printf ("\n");

}

}



1. Scrieți un program C pentru a scădea două matrice.

#include <stdio.h>

void main ()

{

int i, j;

int row, col;

printf ("Introduceti numarul de randuri pentru ambele matrice: ");

scanf ("%d", &row);

printf ("Introduceti numarul de coloane pentru ambele matrice: ");

scanf ("%d", &col);

int arr1[10][10], arr2[10][10];

printf ("Introduceti elementele primei matrice: \n");

for ( i=0; i<row; i++)

{

for (j=0; j<col; j++)

{

printf ("Elementul %d%d = ",i+1, j+1);

scanf("%d", &arr1[i][j]);

}

}

printf ("Introduceti elementele celei de-a doua matrice: \n");

for (i=0; i<row; i++)

{

for (j=0; j<col; j++)

{

printf ("Elementul %d%d = ",i+1, j+1);

scanf("%d", &arr2[i][j]);

}

}

printf ("\n");

printf ("Matricea diferenta: \n");

for (int i=0; i<row; i++)

{

for (int j=0; j<col; j++)

{

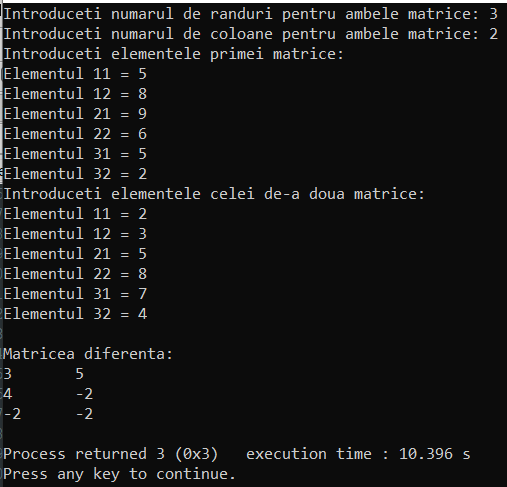
printf("%d\t", arr1[i][j]-arr2[i][j]);

}

printf ("\n");

}

}



1. Scrieți un program C pentru a efectua multiplicarea matricei cu un scalar.

#include <stdio.h>

void main ()

{

int i, j;

int row, col, scalar;

printf ("Introduceti numarul de randuri: ");

scanf ("%d", &row);

printf ("Introduceti numarul de coloane: ");

scanf ("%d", &col);

int arr1[10][10], arr2[10][10];

printf ("Introduceti elementele matricei: \n");

for ( i=0; i<row; i++)

{

for (j=0; j<col; j++)

{

printf ("Elementul %d%d = ",i+1, j+1);

scanf("%d", &arr1[i][j]);

}

}

printf ("Introduceti scalarul: ");

scanf ("%d", &scalar);

printf ("\n");

printf ("Matricea multiplicata cu %d: \n", scalar);

for (int i=0; i<row; i++)

{

for (int j=0; j<col; j++)

{

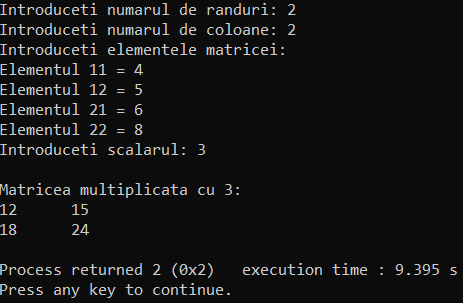
printf("%d\t", arr1[i][j]\*scalar);

}

printf ("\n");

}

}



1. Scrieți un program C pentru a multiplica două matrice.

#include <stdio.h>

void main ()

{

int i, j;

int row, col;

printf ("Introduceti numarul de randuri pentru ambele matrice: ");

scanf ("%d", &row);

printf ("Introduceti numarul de coloane pentru ambele matrice: ");

scanf ("%d", &col);

int arr1[10][10], arr2[10][10], mult[10][10];

printf ("Introduceti elementele primei matrice: \n");

for ( i=0; i<row; i++)

{

for (j=0; j<col; j++)

{

printf ("Elementul %d%d = ",i+1, j+1);

scanf("%d", &arr1[i][j]);

}

}

printf ("Introduceti elementele celei de-a doua matrice: \n");

for (i=0; i<row; i++)

{

for (j=0; j<col; j++)

{

printf ("Elementul %d%d = ",i+1, j+1);

scanf("%d", &arr2[i][j]);

}

}

printf ("\n");

printf ("Matricea multiplicata: \n");

for (int i=0; i<row; i++)

{

for (int j=0; j<col; j++)

{

mult[i][j]=0;

for (int k=0; k<col; k++)

{

mult[i][j]=mult[i][j]+arr1[i][k]\*arr2[k][j];

}

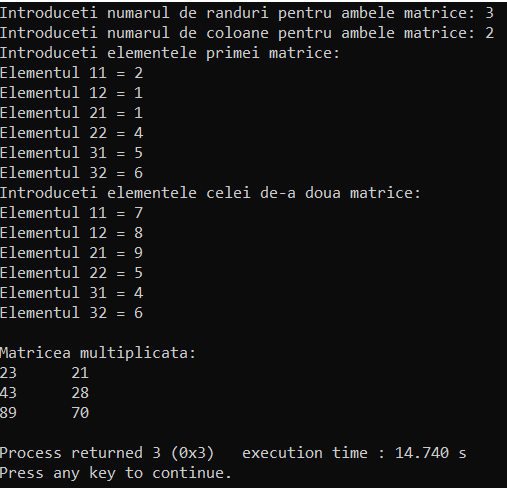
printf ("%d\t", mult[i][j]);

}

printf ("\n");

}

}



1. Scrieți un program C pentru a verifica dacă două matrice sunt egale sau nu.

#include <stdio.h>

#include <stdbool.h>

int main ()

{

int i, j;

int row1, col1, row2, col2;

int arr1[10][10], arr2[10][10], mult[10][10];

bool flag;

printf ("Introduceti numarul de randuri pentru prima matrice: ");

scanf ("%d", &row1);

printf ("Introduceti numarul de coloane pentru a doua matrice: ");

scanf ("%d", &col1);

printf ("\n");

printf ("Introduceti numarul de randuri pentru prima matrice: ");

scanf ("%d", &row2);

printf ("Introduceti numarul de coloane pentru a doua matrice: ");

scanf ("%d", &col2);

printf ("\n");

if (row1!=row2 || col1!=col2)

{

printf ("Matricele nu sunt egale");

}

else

{

printf ("Introduceti elementele primei matrice: \n");

for ( i=0; i<row1; i++)

{

for (j=0; j<col1; j++)

{

printf ("Elementul %d%d = ",i+1, j+1);

scanf("%d", &arr1[i][j]);

}

}

printf ("Introduceti elementele celei de-a doua matrice: \n");

for (i=0; i<row2; i++)

{

for (j=0; j<col2; j++)

{

printf ("Elementul %d%d = ",i+1, j+1);

scanf("%d", &arr2[i][j]);

}

}

printf ("\n");

for ( i=0; i<row1; i++)

{

for (j=0; j<col1; j++)

{

if (arr1[i][j]==arr2[i][j])

flag = true;

}

}

if (flag)

{

printf ("Matricele sunt egale");

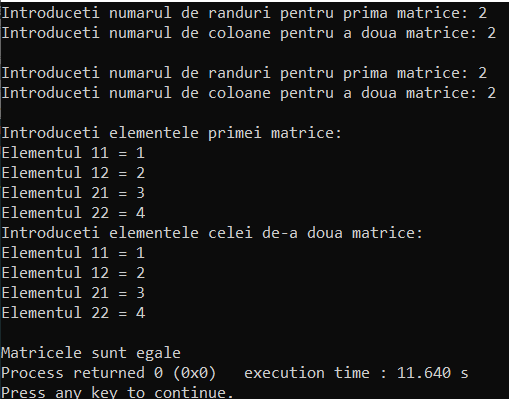
}

else

printf ("Matricele nu sunt egale");

}

}



1. Scrieți un program C pentru a găsi suma elementelor diagonale principale ale unei matrice.

#include <stdio.h>

int main ()

{

int i, j;

int row, col;

int arr1[10][10], sum;

printf ("Introduceti numarul de randuri: ");

scanf ("%d", &row);

printf ("Introduceti numarul de coloane : ");

scanf ("%d", &col);

printf ("\n");

for (i=0; i<row; i++)

{

for (j=0; j<col; j++)

{

printf ("Elementul %d%d = ",i+1, j+1);

scanf("%d", &arr1[i][j]);

}

}

for (i=0; i<row; i++)

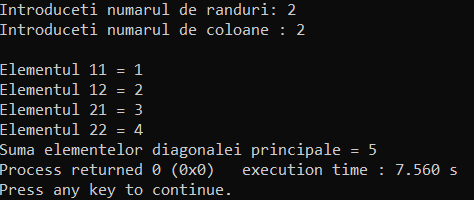
{

sum+=arr1[i][i];

}

printf ("Suma elementelor diagonalei principale = %d",sum);

}



1. Scrieți un program C pentru a găsi suma elementelor diagonalei secundare ale unei matrice.

#include <stdio.h>

int main ()

{

int i, j;

int row, col;

int arr1[10][10], sum;

printf ("Introduceti numarul de randuri: ");

scanf ("%d", &row);

printf ("Introduceti numarul de coloane : ");

scanf ("%d", &col);

printf ("\n");

for (i=0; i<row; i++)

{

for (j=0; j<col; j++)

{

scanf("%d", &arr1[i][j]);

}

}

for (i=0; i<row; i++)

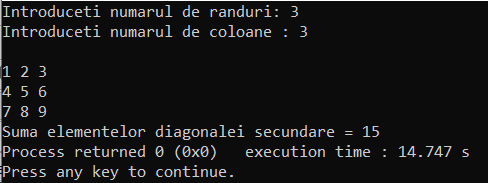
{

sum+=arr1[i][row-1-i];

}

printf ("Suma elementelor diagonalei secundare = %d",sum);

}



1. Scrieți un program C pentru a găsi suma fiecărui rând și coloană a unei matrice.

#include <stdio.h>

int main ()

{

int i, j;

int row, col;

int arr[10][10], sum;

printf ("Introduceti numarul de randuri: ");

scanf ("%d", &row);

printf ("Introduceti numarul de coloane: ");

scanf ("%d", &col);

printf ("\n");

printf ("Introduceti elementele matricei: ");

for (i=0; i<row; i++)

{

for (j=0; j<col; j++)

{

scanf("%d", &arr[i][j]);

}

}

for (i = 0; i < row; i++)

{

for (j = 0; j < col; j++)

{

sum=sum+arr[i][j] ;

}

printf("Suma elementelor randului %d = %d\n", i+1, sum);

sum = 0;

}

for (j = 0; j < col; ++j)

{

for (i = 0; i < row; ++i)

{

sum = 0;

sum = sum + arr[i][j];

}

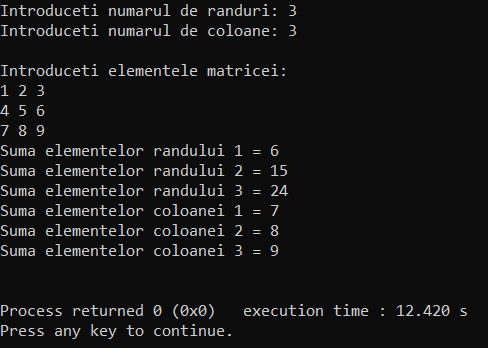
printf("Suma elementelor coloanei %d = %d\n", j+1, sum);

sum = 0;

}

printf ("\n");

}



1. Scrieți un program C pentru a schimba diagonalele unei matrice.

#include <stdio.h>

void main()

{

int i, j, row, col, arr[10][10], temp;

printf("Introduceti numarul de randuri: ");

scanf("%d", &row);

printf("Introduceti numarul de coloane: ");

scanf("%d", &col);

printf("Introduceti elementele matricei patratice: \n");

for(i=0; i<row; i++)

{

for(j=0;j<col; j++)

{

scanf("%d", &arr[i][j]);

}

}

if(i!=j)

printf ("Matricea introdusa nu este una patratica");

else

{

for(i = 0; i < row; i++)

{

temp = arr[i][i];

arr[i][i] = arr[i][row-i-1];

arr[i][row-i-1] = temp;

}

printf("\n Matricea dupa schimbarea diagonalelor: \n");

for(i = 0; i < row; i++)

{

for(j = 0; j < col; j++)

{

printf("%d \t ", arr[i][j]);

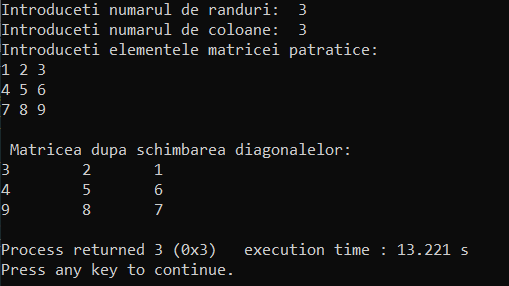
}

printf("\n");

}

}

}



1. Scrieți un program C pentru a găsi matricea triunghiulară superioară.

#include <stdio.h>

void main()

{

int i, j, row, col, arr[10][10];

printf("Introduceti numarul de randuri: ");

scanf("%d", &row);

printf("Introduceti numarul de coloane: ");

scanf("%d", &col);

printf("Introduceti elementele matricei: \n");

for (i = 0; i < row; i++)

{

for (j = 0; j < col; j++)

{

scanf("%d", &arr[i][j]);

}

}

printf("\n");

printf("Matricea triunghiulara superioara: \n");

for (i = 0; i < row; i++)

{

printf("\n");

for (j = 0; j < col; j++)

{

if (i<=j)

{

printf("%d\t", arr[i][j]);

}

else

{

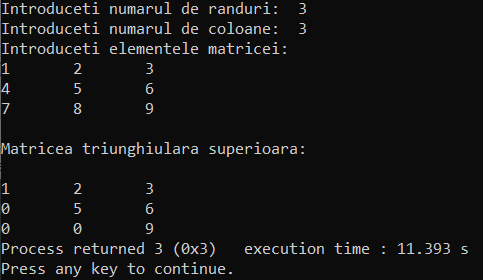
printf("0\t");

}

}

}

}



1. Scrieți un program C pentru a găsi matricea triunghiulară inferioară.

#include <stdio.h>

void main()

{

int i, j, row, col, arr[10][10];

printf("Introduceti numarul de randuri: ");

scanf("%d", &row);

printf("Introduceti numarul de coloane: ");

scanf("%d", &col);

printf("Introduceti elementele matricei: \n");

for (i = 0; i < row; i++)

{

for (j = 0; j < col; j++)

{

scanf("%d", &arr[i][j]);

}

}

printf("Matricea triunghiulara inferioara: \n");

for (i = 0; i < row; i++)

{

printf("\n");

for (j = 0; j < col; j++)

{

if (j<=i)

{

printf("%d\t", arr[i][j]);

}

else

{

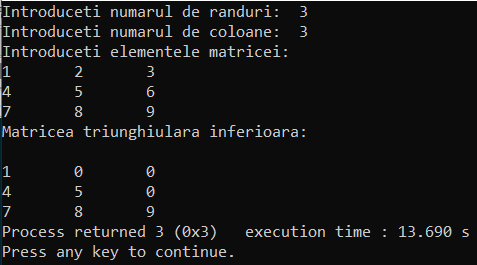
printf("0\t");

}

}

}

}



1. Scrieți un program C pentru a găsi suma matricei triunghiulare superioare.

#include <stdio.h>

void main()

{

int i, j, row, col, arr[10][10], sum=0;

printf("Introduceti numarul de randuri: ");

scanf("%d", &row);

printf("Introduceti numarul de coloane: ");

scanf("%d", &col);

printf("Introduceti elementele matricei: \n");

for (i = 0; i < row; i++)

{

for (j = 0; j < col; j++)

{

scanf("%d", &arr[i][j]);

}

}

printf("\n");

printf("Matricea triunghiulara superioara: \n");

for (i = 0; i < row; i++)

{

printf("\n");

for (j = 0; j < col; j++)

{

if (i<=j)

{

sum+=arr[i][j];

printf("%d\t", arr[i][j]);

}

else

{

printf("0\t");

}

}

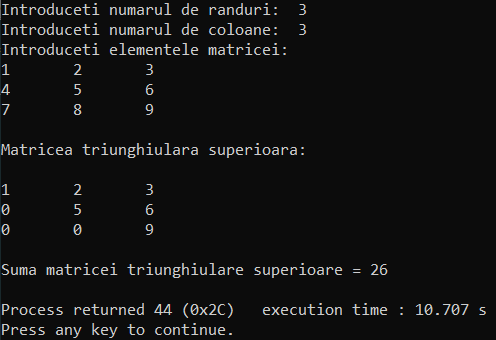
}

printf("\n");

printf("\n");

printf ("Suma matricei triunghiulare superioare = %d\n", sum);

}



1. Scrieți un program C pentru a găsi suma matricei triunghiulare inferioare.

#include <stdio.h>

void main()

{

int i, j, row, col, arr[10][10], sum=0;

printf("Introduceti numarul de randuri: ");

scanf("%d", &row);

printf("Introduceti numarul de coloane: ");

scanf("%d", &col);

printf("Introduceti elementele matricei: \n");

for (i = 0; i < row; i++)

{

for (j = 0; j < col; j++)

{

scanf("%d", &arr[i][j]);

}

}

printf("\n");

printf("Matricea triunghiulara inferioara: \n");

for (i = 0; i < row; i++)

{

printf("\n");

for (j = 0; j < col; j++)

{

if (j<=i)

{

sum+=arr[i][j];

printf("%d\t", arr[i][j]);

}

else

{

printf("0\t");

}

}

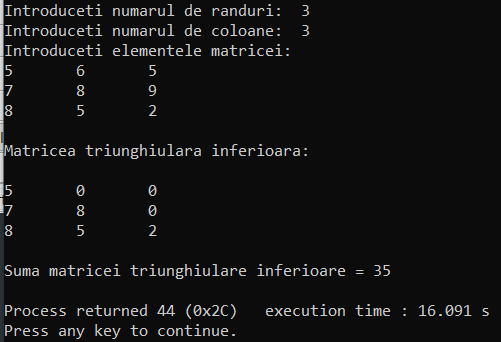
}

printf("\n");

printf("\n");

printf ("Suma matricei triunghiulare inferioare = %d\n", sum);

}



1. Scrieți un program C pentru a roti la stînga o matrice.

#include <stdio.h>

int main()

{

int row, col;

int i, j;

printf("Introduceti numarul de randuri: ");

scanf("%d", &row);

printf("Introduceti numarul de coloane: ");

scanf("%d", &col);

int arr[10][10];

printf("Introduceti elementele matricei: \n");

for (i = 0; i < row; i++)

{

for (j = 0; j < col; j++)

{

scanf("%d", &arr[i][j]);

}

}

printf("\n");

printf ("Matricea rotita la stanga: \n");

rotate\_left(arr, row, col);

}

void rotate\_left(int m[10][10],int row, int col)

{

for(int i=row-1; i>=0; i--)

{

for(int j=0; j<col; j++)

{

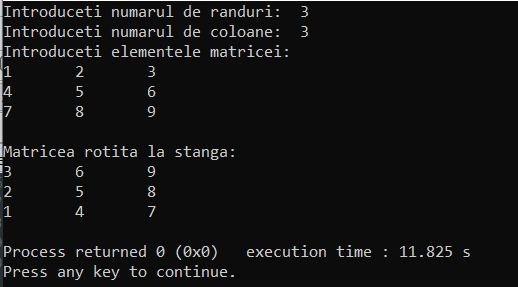
printf("%d\t",m[j][i]);

}

printf("\n");

}

}



1. Scrieți un program C pentru a roti la dreapta o matrice.

#include <stdio.h>

int main()

{

int row, col;

int i, j;

printf("Introduceti numarul de randuri: ");

scanf("%d", &row);

printf("Introduceti numarul de coloane: ");

scanf("%d", &col);

int arr[10][10];

printf("Introduceti elementele matricei: \n");

for (i = 0; i < row; i++)

{

for (j = 0; j < col; j++)

{

scanf("%d", &arr[i][j]);

}

}

printf("\n");

rotate(arr, row, col);

}

void rotate(int m[10][10],int row, int col)

{

for(int i=0; i<row; i++)

{

for(int j=col-1; j>=0; j--)

{

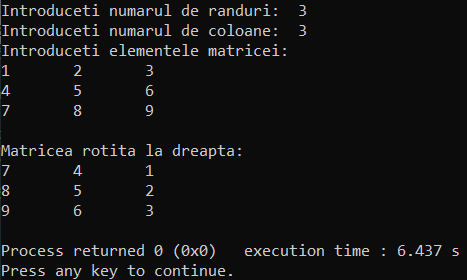
printf("%d\t",m[j][i]);

}

printf("\n");

}

}



1. Scrieți un program C pentru a roti în sus o matrice.

#include <stdio.h>

int main()

{

int row, col;

int i, j;

printf("Introduceti numarul de randuri: ");

scanf("%d", &row);

printf("Introduceti numarul de coloane: ");

scanf("%d", &col);

int arr1[10][10], arr2[10][10];

printf("Introduceti elementele matricei: \n");

for (i = 0; i < row; i++)

{

for (j = 0; j < col; j++)

{

scanf("%d", &arr1[i][j]);

}

}

printf("\n");

printf ("Matricea rotita in sus: \n");

rotate\_up(arr1, arr2, row, col);

for(int i=0; i<row; i++)

{

for(int j=0; j<col; j++)

{

printf("%d\t",arr2[i][j]);

}

printf("\n");

}

}

void rotate\_up(int m[10][10], int n[10][10], int row, int col)

{

for(int i=0; i<row; i++)

{

for(int j=0; j<col; j++)

{

if (i==0)

{

n[row-1][j]=m[i][j];

}

else

{

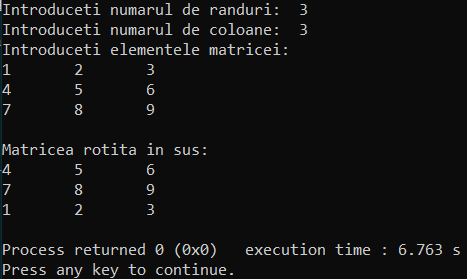
n[i-1][j]=m[i][j];

}

}

}

}

****

1. Scrieți un program C pentru a roti în jos o matrice.

#include <stdio.h>

int main()

{

int row, col;

int i, j;

printf("Introduceti numarul de randuri: ");

scanf("%d", &row);

printf("Introduceti numarul de coloane: ");

scanf("%d", &col);

int arr[10][10];

printf("Introduceti elementele matricei: \n");

for (i = 0; i < row; i++)

{

for (j = 0; j < col; j++)

{

scanf("%d", &arr[i][j]);

}

}

printf("\n");

printf ("Matricea rotita in jos: \n");

rotate\_down(arr, row, col);

for(int i=0; i<row; i++)

{

for(int j=0; j<col; j++)

{

printf("%d\t",arr[i][j]);

}

printf("\n");

}

}

void rotate\_down(int m[10][10], int row, int col)

{

int temp;

for(int i=0; i<row; i++)

{

for(int j=0; j<col; j++)

{

temp = m[i][j];

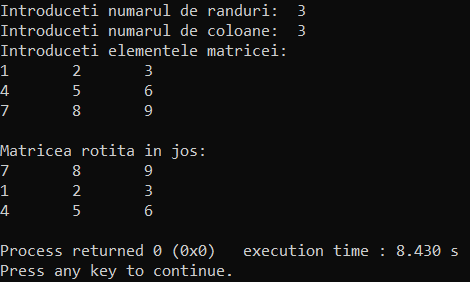
m[i][j]= m[row-1][j];

m[row-1][j]=temp;

}

}

}



**Rezultatul lucrării:**

* Ce reprezintă un tablou unidimensional în limbajul C?

Un tablou unidimensional în limbajul C reprezintă un șir de elemente, ce se includ în același tip de date și sunt aranjate uniform într-un rând.

* Ce reprezintă un tablou bidimensional în limbajul C?

Un tablou bidimensional în limbajul C reprezintă o structură formată din rânduri și coloane, fiecare element fiind determinat de 2 coordonate: numărul rândului și numărul coloanei.

* Cu ce operator se accesează elementele unui tablou în limbajul C?

Elementele unui tablou în limbajul C se accesează prin denumirea tabloului și index-ul elementului utilizând operatorul „[]” :

*tipul\_de\_date\_tablou nume\_tablou[index]*

* Ce tip de date pot fi indecșii unui tablou în limbajul C?

Indecșii unui tablou pot fi doar de tipul de date *int*, deoarece reprezintă numere naturale.

* Ce index are primul element din tabloul unidimensional ?

Primul element din tabloul unidimensional are index-ul 0.

* Ce index are ultimul element din tabloul unidimensional cu lungimea N?

Ultimul element din tabloul unidimensional cu lungimea N are indexul N-1.

* Ce algoritmi de sortare a tablourilor ați utilizat în lucrare?

1. Metoda bulelor (bubble sort) – se precaută cel mai mic element până ce vectorul nu este sortat în ordinea solicitată.
2. Sortarea prin selecție (selection sort) – se parcurge vectorul cu indicele i și vectorul cu indicele j (j=i+1). Dacă elementele nu se află în ordinea dorită, se schimbă ordinea acestora, utilizând o variabilă auxiliară.