Національний технічний університет України

«Київський політехнічний інститут імені Ігоря Сікорського»

**ЗВІТ З ЛАБОРАТОРНОЇ РОБОТИ**

**№4**

з дисципліни:

«ІНФОРМАТИКА. Основи програмування та алгоритми»

|  |  |
| --- | --- |
|  | Виконав: Демарьов Костянтин  Група: РЕ–11  Викладачі:  доцент Катін П.Ю.  Оцінка: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ |

Київ – 2021

***Мета роботи***:

скласти програми для роботи з двовимірними масивами.

***Код:***

#include <stdio.h>

#include <stdlib.h>

#include <math.h>

#include <time.h>

int \*\* fill\_array(int \*\*A, int a, int b)

{

int counter;

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

{

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

{

printf(" Element [%d][%d] = ",i+1,j+1);

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

printf("\n");

}

}

return A;

}

int \*\* print\_array(int \*\*A, int a, int b )

{

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

{

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

{

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

}

printf("\n");

}

}

int \*\* random\_array(int \*\*A, int a, int b)

{

srand(time(NULL));

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

{

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

{

A[i][j] = rand()%100 + 1;

}

}

}

void transp\_array(int \*\*B, int a, int b)

{

int D[b][a];

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

{

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

{

D[j][i] = B[i][j];

}

}

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

{

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

{

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

}

printf("\n");

}

}

void product\_of\_arrays(int \*\*A, int \*\*B, int nA, int nB, int mB)

{

int C[nA][mB];

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

{

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

{

C[i][j] = 0;

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

C[i][j] += A[i][k] \* B[k][j];

}

}

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

{

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

{

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

}

printf("\n");

}

}

void sort\_array(int \*\*A, int nA, int k)

{

int A1[nA],temp,A\_max,A\_min,counter = 0;

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

{

if(i==k)

{

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

{

A1[j] = A[i][j];

}

}

}

printf("\n");

temp = A1[0];

A\_min = A1[0];

A\_max = A1[0];

int T;

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

{

if(A1[i] > A\_max)

{

A\_max = A1[i];

}

if(A1[i] < A\_min)

{

A\_min = A1[i];

}

}

printf("\nA(max) = %d\n", A\_max);

printf("\nA(min) = %d\n", A\_min);

printf("row %d\n",k);

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

{

printf("%d \t",A1[i]);

}

printf("\n");

do

{

counter = 0;

for(int i = 0;i<nA-1;i++)

{

if(A1[i]>A1[i+1])

{

counter++;

temp = A1[i];

A1[i] = A1[i+1];

A1[i+1] = temp;

}

}

}while(counter !=0);

printf("sorted row %d\n",k);

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

{

printf("%d \t",A1[i]);

}

printf("\n");

}

void max\_min\_above\_array(int \*\*A, int a)

{

printf("\n\nAbove the main diagonal\n");

int k = 0;

int A1[a];

int A\_max = 0;

int A\_min = 100;

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

{

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

{

if(i < j) {

A1[k] = A[i][j];

printf("A[%d][%d] =%d\t", i, j, A1[k]);

k++;

}

}

}

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

{

if(A1[i] > A\_max)

{

A\_max = A1[i];

}

if(A1[i] < A\_min)

{

A\_min = A1[i];

}

}

printf("\n\nA(max) = %d", A\_max);

printf("\tA(min) = %d", A\_min);

}

void max\_min\_below\_array(int \*\*A, int a)

{

printf("\n\nBelow the main diagonal\n");

int k = 0;

int A1[a];

int A\_max = 0;

int A\_min = 100;

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

{

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

{

if(i > j) {

A1[k] = A[i][j];

printf("A[%d][%d] =%d\t", i, j, A1[k]);

k++;

}

}

}

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

{

if(A1[i] > A\_max)

{

A\_max = A1[i];

}

if(A1[i] < A\_min)

{

A\_min = A1[i];

}

}

printf("\n\nA(max) = %d", A\_max);

printf("\tA(min) = %d", A\_min);

}

int \*\* memoryAllocation(int rows, int cols)

{

int i = 0;

int \*\*matrix = (int \*\*)malloc(rows\*sizeof(int \*));

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

{

matrix[i] = (int \*)malloc(cols\*sizeof(int));

}

return matrix;

}

void clearMemory(int \*\*matrix, int rows)

{

int i;

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

{

free(matrix[i]);

}

free(matrix);

}

void sum\_rows\_cols(int \*\*arrayA, int rowsA, int colsA,int \*\*arrayB, int rowsB, int colsB)

{

int i,j,x;

printf("Sum rows A: \n");

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

{

x = 0;

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

{

x+=arrayA[i][j];

}

printf("Row [%d] = %d \n",i+1,x);

}

printf("Sum cols B: \n");

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

{

x = 0;

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

{

x+=arrayB[j][i];

}

printf("Col [%d] = %d \n",i+1,x);

}

}

int main()

{

int nA, nB, mB, b1B, a1B;

int \*\*A;

int \*\*B;

unsigned int method, exit, k;

printf("Enter the size of the array A: ");

scanf("%d", &nA);

nA=nA;

printf("Enter the number of rows of array B: ");

scanf("%d", &nB);

printf("Enter the number of cols of array B: ");

scanf("%d", &mB);

A = memoryAllocation(nA,nA);

B = memoryAllocation(nB,mB);

printf("\nIf you want to enter from the keyboard - enter \"1\".");

printf("\nIf you want it to be asked initialization - enter \"2\".");

printf("\nYour choose: ");

scanf("%d", &method);

switch(method)

{

case 1:

{

printf("Your choose to enter from the keyboard");

fill\_array(A, nA, nA);

fill\_array(B, nB, mB);

break;

}

case 2:

{

printf("Your choose to be asked initialization");

random\_array(A, nA, nA);

random\_array(B, nB, mB);

break;

}

}

printf ("\nArray A\n\n");

print\_array(A, nA, nA);

printf ("\nArray B\n\n");

print\_array(B, nB, mB);

do

{

printf("\n\nFind the maximum element of the array A and the minimum of the elements above / below the main diagonal - enter the \"1\" ");

printf("\n\nTransportation of the array B - enter the \"2\" ");

printf("\n\nFind the product of the arrays A and B - enter the \"3\" ");

printf("\n\nSort the elements from minimum to maximum value in the array A - enter the \"4\"");

printf("\n\nTo find the sum of elements of rows of the array A and cols of the array B - enter the \"5\" ");

printf("\n\nChoose your method: ");

scanf("%d", &method);

switch(method)

{

case 1:

{

printf("\nFind the maximum element of the array A and the minimum of the elements above / below the main diagonal\n");

max\_min\_above\_array(A, nA);

max\_min\_below\_array(A, nA);

break;

}

case 2:

{

printf("\nTransportation of the array B\n\n");

transp\_array(B, nB, mB);

break;

}

case 3:

{

printf("\nFind the product of the arrays A and B\n\n");

if(nA != nB)

{

printf("The product of the arrays is impossible");

}

else

{

product\_of\_arrays(A, B, nA, nB, mB);

}

break;

}

case 4:

{

printf("\nSort the elements from minimum to maximum value in the array A\n");

printf("\nSelect the row in which you want to sort the items. First row - \"k\" = 0, second row - \"k\" = 1 etc");

printf("\nk = ");

scanf("%d", &k);

sort\_array(A, nA, k);

break;

}

case 5:

{

printf("\nTo find the sum of elements of rows of the array A and cols of the array B\n\n");

sum\_rows\_cols(A, nA, nA, B, nB, mB);

break;

}

}

printf("\nEnter \"1\" if you want to exit of the program.");

printf("\n\nEnter more than \"1\" if you want to continue the program: ");

scanf("%d", &exit);

}while(exit>1);

clearMemory(A,nA);

clearMemory(B,nB);

return 0;

}

***Висновки:***

Я навчився працювати з двовимірним масивами, а також виконав різні дії з ними (транспонування матриці, добуток матриць, знаходження максимальних та мінімальних елементів вище та нижче головної діагоналі, сортування елементів матриці та ін.).