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 двовимірними масивами.

Код:

#define N 30

#include <stdio.h>

#include <stdlib.h>

#include <math.h>

void fill\_array(int arr[][N], int a, int b)

{

int counter;

printf("The function of inputing elements of massive\n");

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

{

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

{

arr[i][j] = 0;

}

}

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

{

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

{

printf("Press the element [%d][%d] of massive: ", i, j);

scanf("%d", &counter);

arr[i][j] = counter;

}

}

}

void print\_array(int arr[][N], int a, int b)

{

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

{

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

{

printf("%3d ", arr[i][j]);

}

printf("\n");

}

}

void random\_array(int arr[][N], int a, int b)

{

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

{

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

{

arr[i][j] = 0;

}

}

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

{

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

{

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

}

}

srand(time(NULL));

}

void sort\_array(int arr[][N], int a, int b)

{

int z, x, w, q;

for (w = 0; w < a\*b; w++)

{

for (z = 0; z < b; z++)

{

for (x = 0; x < a; x++)

{

if (x == 0 && z > 0)

{

if (arr[z-1][b-1] > arr[z][x])

{

q = arr[z-1][b-1];

arr[z-1][b-1] = arr[z][x];

arr[z][x] = q;

}

}

else if (arr[z][x-1] > arr[z][x])

{

q = arr[z][x-1];

arr[z][x-1] = arr[z][x];

arr[z][x] = q;

}

}

}

}

for (z = 0; z < a; z++)

{

for (x = 0; x < b; x++)

{

printf("%3d ", arr[z][x]);

}

printf("\n");

}

}

void swap\_array(int arr1[][N], int arr2[][N], int a, int b)

{

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

{

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

{

arr2[i][j] = arr1[i][j];

}

}

}

void transp\_array(int B[][N], int a, int b)

{

int F[b][a];

B[a][b];

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

{

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

{

F[j][i] = B[i][j];

}

}

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

{

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

{

printf("%3d ", F[i][j]);

}

printf("\n");

}

}

void dob\_array(int A[][N], int B[][N], int a, int b, int c)

{

int sum;

int D[a][c];

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

{

for (int h = 0; h < c; h++)

{

sum=0;

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

{

sum+=A[j][g]\*B[g][h];

}

D[j][h]=sum;

}

}

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

{

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

{

printf("%3d ", D[i][j]);

}

printf("\n");

}

}

void sum\_r\_array(int A[][N], int h)

{

int G[N]={0};

printf("The sum of the elements of the rows of the matrix A\n");

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

{

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

{

G[i]+=A[i][j];

}

}

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

{

printf("%3d ", G[i]);

}

}

void sum\_s\_array(int B[][N], int a, int b)

{

int H[N]={0};

printf("The sum of the elements of the cols of the matrix B\n");

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

{

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

{

H[i]+=B[j][i];

}

}

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

{

printf("%3d ", H[i]);

}

}

void max\_min\_array(int A[10][10], int a)

{

int vm=A[0][0], vn=A[0][0], nm=A[0][0], nn=A[0][0];

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

{

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

{

if (A[i][j] > vm) {

vm=A[i][j];

}

if (A[i][j] < vn) {

vn=A[i][j];

}

}

}

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

{

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

{

if (A[i][j] > nm) {

nm=A[i][j];

}

if (A[i][j] < nn) {

nn=A[i][j];

}

}

}

printf("\nUpper the general diagonal the highest %3d and the lowest value %3d", vm, vn);

printf("\nBelow the general diagonal the highest %3d and the lowest value %3d", nm, nn);

}

int main()

{

int na, nb, mb;

unsigned int m, metod, exit;

printf("Set the number of rows and cols for matrix A:");

scanf("%u",&na);

printf("Set the number of rows for matrix B:");

scanf("%u",&mb);

printf("Set the number of cols for matrix B:");

scanf("%u",&nb);

int A[na][na];

int B[nb][mb];

int C[na][na];

int L[nb][mb];

printf("\nBuild the matrix A with size %uX%u and the matrix B with the size %uX%u", na, na, nb, mb);

printf("\nChoose the method:\n\t|1-Set own values for the matrix\n\t|2-Do the automatic selection 1-100 of matrix\n");

scanf("%u",&m);

switch(m) {

case 1:

{

fill\_array(A, na, na);

fill\_array(B, mb, nb);

break;

}

case 2:

{

random\_array(A, na, na);

random\_array(B, mb, nb);

break;

}

}

printf("\nOutput of the matrix A\n");

print\_array(A, na, na);

printf("\nOutput of the matrix B\n");

print\_array(B, mb, nb);

swap\_array(A, C, na, na);

swap\_array(B, L, mb, nb);

do{

printf("\nWhat would we do:\n\t|1-The looking for the maximal and minimal values above and below the general diagonal A\n\t|2-The transportation of matrix B");

printf("\n\t|3-The looking for product of the matrix A\*B\n\t|4-Sort the matrix A from 0\n\t|5-To find the sum of the rows of the matrix A\n\t|6-To find the sum of the cols of the matrix B\n");

scanf("%u",&metod);

switch(metod) {

case 1:

{

max\_min\_array(C, na);

break;

}

case 2:

{

printf("\nOutput of transpontationed matrix B\n");

transp\_array(L, mb, nb);

break;

}

case 3:

{

printf("\nOutput of the product of matrix A\*B\n");

if (mb != na)

printf("Product of matrix is impossible!!!\n");

else

dob\_array(A, B, na, na, nb);

printf("\n");

break;

}

case 4:

{

printf("\nOutput of sorted matrix A\n");

sort\_array(C, na, na);

break;

}

case 5:

{

sum\_r\_array(C, na);

break;

}

case 6:

{

sum\_s\_array(B, mb, nb);

break;

}

}

printf("\nIf you want to exit of the program press the \"1\", but if you want to continue - press the \"2\"\n");

scanf("%u",&exit);

if (exit < 1 || 2 < exit) {

printf("\n1 or 2, no\n");

scanf("%u",&exit);

}

} while (exit > 1);

return 0;

}