Національний технічний університет України

«Київський політехнічний інститут імені Ігоря Сікорського»

Кафедра теоретичних основ радіотехніки

**ЗВІТ З ЛАБОРАТОРНОЇ РОБОТИ №4**

з дисципліни: «Інформатика 1»

|  |  |
| --- | --- |
|  | Виконав : Пінчук Олександр Олександрович  Група: РЕ-12  Викладачі: доцент Катін П.Ю.  Оцінка: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  Підпис: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ |

Київ – 2021

**Мета роботи**: скласти програми для роботи з двовимірними масивами.

**Ключові моменти**:

1. В лабораторній ми використувуємо вказівники, та вказівники на вказівники
2. Використовуємо оператори malloc для виділення пам'яті.
3. Використовуємо free для звільнення пам'яті.

**Код:**

#include <stdio.h>

#include <stdlib.h>

#include <locale.h>

#include <conio.h>

#include <malloc.h>

//M - кол-во строчек,N - кол-во столбиков

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

{

int i = 0;

int \*\*matrix = (int )malloc(rows\*sizeof(int \*));// даем А тип "инт" малок выделяет память под А

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);

}

int Maximum(int \*\*A,int N,int i,int j,int max)//элементы матрицы, А - квадратная матрица

{

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

{

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

{

if (A[i][j] > max)

{

max = A[i][j]; // i - кол-во строчек, j - кол-во столбиков

}

}

}

printf("\nMaximum value of matrix A: %d\n",max);

}

void Transponate (int \*\*B,int N,int M) //B - свободная, Т - квадратная

{

int i,j;

printf("\n Transpose matrix B : \n");

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

{

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

{

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

}

printf("\n");

}

}

void Increase(int \*\*A,int rowsA,int colsA,int \*\*B,int rowsB,int colsB )

{

int i,j,p;

int S=0;

if(rowsA!=rowsB)

{

printf("Error\n");

}

else

{

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

{

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

{

S=0;

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

{

S+= A[i][p] \* B[p][j];//S=S+A[i][p] \* B[p][j]

}

printf("%d \t",S);

}

printf("\n");

}

}

}

int Sort(int \*\*A,int sorting,int save,int N,int i,int j)

{

int k;

for(k=1;k<N;k++)

{

for(j=0;j<N-k;j++)

{

if(A[sorting][j]>A[sorting][j+1])

{

save = A[sorting][j];

A[sorting][j] = A[sorting][j+1];

A[sorting][j+1] = save;

}

}

}

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

{

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

}

}

int Summ(int \*\*A,int \*\*B,int sum,int N,int M,int i,int j)

{

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

{

sum=0;

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

{

sum += A[i][j];

}

printf("\n%d) The sum of the elements of the rows of matrix A = \t%d",i+1,sum);

}

printf("\n");

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

{

sum=0;

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

{

sum += B[i][j];

}

printf("\n%d) The sum of the elements of the columns of the matrix B = \t%d",j+1,sum);

}

}

int main()

{

int i,j,save,N=3,M=4,action;

int max=0,sorting,sum=0,enter;

int \*\*A,\*\*B;

printf("\nEnter the size of the square matrix A (N x N)\n");

printf("N = ");

scanf("%d",&N);

printf("\nEnter the size of the rectangular matrix B (N x M)\n");

printf("M = ");

scanf("%d",&M);

A=memoryAllocation(N,N);

B=memoryAllocation(N,M);

printf("Create matrix by:\n1. Automatic\n2. Keyboard\n");

scanf("%d",&enter);

if(enter == 1)

{

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

{

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

{

A[i][j] = rand()%10;

}

}

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

{

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

{

B[i][j] = rand()%15;

}

}

}

else if(enter == 2)

{

printf("Enter the elements of a square matrix:\n");

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

{

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

{

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

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

}

}

printf("Enter the elements of a rectangular matrix:\n");

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

{

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

{

printf("B[%d][%d]=",i,j);

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

}

}

}

else

{

printf("Sorry no variant \n");

}

printf("\nMatrix A :\n");

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

{

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

{

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

}

printf("\n");

}

printf("\n");

printf("\nMatrix B : \n");

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

{

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

{

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

}

printf("\n");

}

do

{

printf("\nChoose action::\n1. Max A \n2. Transpose B\n3. Multiplication AxB\n4. Sort row A\n5. Sum in rows and columns\n");

scanf("%d",&action);

}while(action != 1 && action != 2 && action != 3 && action != 4 && action != 5);

switch (action)

{

case 1:

Maximum(A,N,i,j,max);

break;

case 2:

Transponate(B,N,M);

break;

case 3:

printf("AxB:\n");

Increase(A,N,N,B,N,M);

break;

case 4:

printf("\nEnter the line number to sort: ");

scanf("%d",&sorting);

Sort(A,sorting,save,N,i,j);

break;

case 5:

Summ(A,B,sum,N,M,i,j);

break;

default:

printf("No such a varian\n");

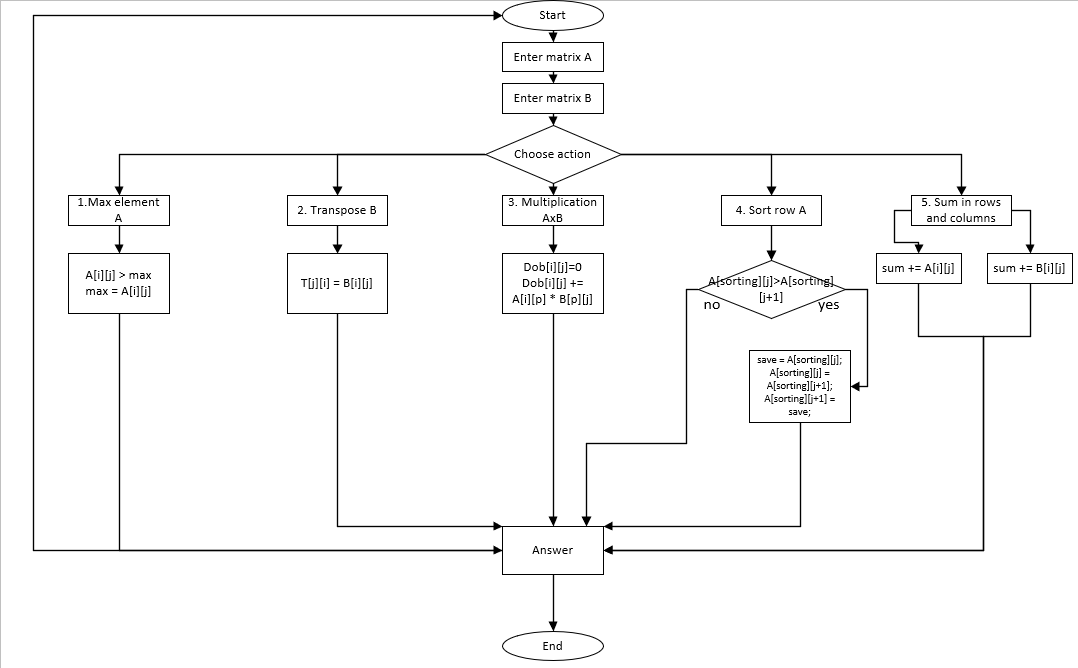
break;

}

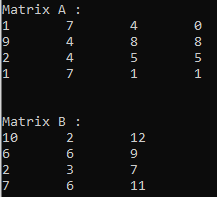
clearMemory(A,N);

clearMemory(B,N);

return 0;

}

***Висновок:***

Матриці А та B (В квадратна, А прямокутна)

Ми написали програми, що виконують певні дії с матрицями (массивами) – множення, транспонування, сортування, знаходження найбільшого числа у певній строчці

Для динамічного масиву ми виділяємо пам'ять, яку маємо очистити її після виконання програми.