**1.Написать программу, которая осуществляет добавление строки**

**или столбца в любое место двумерной**

**матрицы по выбору пользователя.**

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_-**

#include <iostream>

#include <Windows.h>

using namespace std;

void pokaz(int\*\*, int, int);

int\*\* riad(int\*\*, int&, int, int);

int\*\* stolb(int\*\*, int, int&, int);

int main()

{

system("chcp 1251");

int Riad=5 , Colona=5 ;

int\*\* arr = new int\* [Riad];

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

arr[i] = new int[Colona] {1, 1, 1, 1, 1};

cout<<" Программа осуществляет добавление строки или столбца в любое место двумерной матрицы по выбору. "<<endl;

pokaz(arr, Riad, Colona);

cout << endl;

char otvet;

do

{

cout << " Введите куда сделать вставку: если в строку[ 1 ] если в столбец[ 0 ] : ";

cin >> otvet;

}

while (otvet != '1' && otvet != '0');

unsigned int p;

if (otvet == '1')

{

do

{

cout << " Введите после кокой строки вставить : ";

cin >> p;

}

while (p > Riad);

arr = riad(arr, Riad, Colona, p);

}

else

{

do

{

cout << " Введите после какого сталбца вставить : ";

cin >> p;

}

while (p > Colona);

arr = stolb(arr, Riad, Colona, p);

}

pokaz(arr, Riad, Colona);

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

delete[] arr[i];

delete[] arr;

return 0;

}

void pokaz(int\*\* arr, int r, int c)

{

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

{

cout << endl;

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

cout << arr[i][j] << " ";

}

}

int\*\* riad(int\*\* arr, int& R, int C, int pos)

{

R++;

int\*\* new\_arr = new int\* [R];

for (int i = 0; i < R; i++) new\_arr[i] = new int[C];

for (int i = 0, ti = 0; i < R; i++)

{

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

if (i == pos) new\_arr[i][j] = 0;

else new\_arr[i][j] = arr[ti][j];

if (i != pos) ti++;

}

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

delete[] arr[i];

delete[] arr;

arr = NULL;

return new\_arr;

}

int\*\* stolb(int\*\* arr, int R, int& C, int pos)

{

C++;

int\*\* new\_arr = new int\* [R];

for (int i = 0; i < R; i++) new\_arr[i] = new int[C];

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

for (int j = 0, tj = 0; j < C; j++)

{

if (j == pos) new\_arr[i][j] = 0;

else new\_arr[i][j] = arr[i][tj];

if (j != pos) tj++;

}

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

delete[] arr[i];

delete[] arr;

arr = NULL;

return new\_arr;

}

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**2. Дана матрица порядка M×N (M строк, N столбцов). Необходимо заполнить ее значениями и  написать функцию, осуществляющую циклический сдвиг строк и/или столбцов массива указанное количество раз и в указанную сторону.**

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_-**

#include <iostream>

#include <Windows.h>

using namespace std;

int\*\* sozdat\_mass();

void pechat\_mass(int\*\* arr);

void vopros(int, int, char&, int&);

void show(int\*\*, int, int);

void r\_up(int\*\*, int, int);

void r\_dn(int\*\*, int, int);

void c\_lt(int\*\*, int, int);

void c\_rt(int\*\*, int, int);

int STROKA = 5;

int KOLONKA = 5;

int main()

{

system("chcp 1251");

void(\*vypoln\_trebov)(int\*\*, int, int) = show;

char deistvia;

int pozocia;

int\*\* mass = sozdat\_mass();

pechat\_mass(mass);

vopros(STROKA, KOLONKA, deistvia, pozocia);

switch (deistvia)

{

case'r':

vypoln\_trebov = c\_rt;

break;

case'l':

vypoln\_trebov = c\_lt;

break;

case'u':

vypoln\_trebov = r\_up;

break;

case'd':

vypoln\_trebov = r\_dn;

break;

}

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

{

vypoln\_trebov(mass, STROKA, KOLONKA);

}

show(mass, STROKA, KOLONKA);

for (int i = 0; i < STROKA; i++) delete[] mass[i];

delete[] mass;

return 0;

}

/\* Malloc - это высокопроизводительная команда - клиент

функций libc malloc, calloc, free, posix\_memalign

и realloc для управления объектами хранения меньше,

чем 513 байт\*/

int\*\* sozdat\_mass()

{

int i, j;

int\*\* vozvrat = (int\*\*)malloc(sizeof(int\*) \* STROKA);

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

vozvrat[i] = (int\*)malloc(sizeof(int) \* KOLONKA);

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

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

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

return vozvrat;

}

void pechat\_mass(int\*\* arr)

{

int i, j;

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

{

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

cout << arr[i][j] << " ";

cout << endl;

}

cout << endl;

}

void show(int\*\* arr, int r, int c)

{

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

{

cout << endl;

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

cout << arr[i][j] << " ";

}

cout << " ";

}

void r\_up(int\*\* arr, int stroka, int kolonka)

{

int\* tmp = new int[kolonka];

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

tmp[j] = arr[0][j];

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

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

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

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

arr[stroka - 1][j] = tmp[j];

delete[] tmp;

}

void r\_dn(int\*\* arr, int stroka, int klonka)

{

int\* tmp = new int[klonka];

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

tmp[j] = arr[stroka - 1][j];

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

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

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

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

arr[0][j] = tmp[j];

delete[] tmp;

}

void c\_lt(int\*\* arr, int stroka, int kolonka)

{

int\* tmp = new int[stroka];

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

tmp[i] = arr[i][0];

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

for (int j = 0; j < kolonka - 1; j++)

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

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

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

delete[] tmp;

}

void c\_rt(int\*\* arr, int stroka, int kolonka)

{

int\* tmp = new int[stroka];

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

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

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

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

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

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

arr[i][0] = tmp[i];

delete[] tmp;

}

void vopros(int kolonka, int stroka, char& otvet, int& pozicia)

{

do

{

cout << " сдвиг вправо [ r ] влево [ l ] вверх [ u ] вниз [ d ]: ";

cin >> otvet;

} while (otvet != 'r' && otvet != 'l' && otvet != 'u' && otvet != 'd');

if (otvet == 'u' || otvet == 'd')

{

do

{

cout << " сдвинется строке на сколько позиций?: ";

cin >> pozicia;

} while (pozicia < 1 || pozicia > stroka);

}

else

{

do

{

cout << " сдвинется по колонки на сколько позиций?: ";

cin >> pozicia;

} while (pozicia < 1 || pozicia > kolonka);

}

}