**Лабораторная работа №4**

*РАЗРЕЖЕННЫЕ МАТРИЦЫ*

*Выполнил:* Чекулаев В. Ю.

ПО(аб)-81

***Задание:***

В лабораторной работе необходимо программно **реализовать**:

1. Представление разреженной матрицы размером NxN или разреженного вектора размером N в соответствии с вариантом (см. табл. 1);
2. Операцию считывания ненулевых элементов матрицы из файла;
3. Операцию получения значения элемента по номеру строки и номеру столбца;
4. Вывод полной матрицы на экран;
5. Операцию, указанную в табл. 2, в соответствии с вариантом.

***Структура данных для представления матрицы:*** мультисписок.

***Индивидуальное задание:*** определение суммы элементов указанного столбца в нечетных строках.

1) ***Код программы:***

#include<iostream>

#include<stdio.h>

#include<string>

using namespace std;

const int N = 16;

struct ELEM

{

int num;

int row;

int col;

ELEM \*right;

ELEM \*down;

};

int matrixForm(ELEM \*List\_Row, ELEM \*List\_Col); // Чтение матрицы из файла

ELEM\* findRow(ELEM \*List\_Row, int i); // Поиск строки по индексу

ELEM\* findCol(ELEM \*List\_Col, int i); //Поиск колонки по индексу

ELEM\* initRow(ELEM \*List\_Row, int i); //Создание списка строк

ELEM\* initCol(ELEM \*List\_Col, int i); //Создание списка колонок

void showMatrix(ELEM \*List\_Row); // Вывод матрицы на экран

int getData(ELEM\* List\_Row, int i, int j); // Поиск значения элемента матрицы по индексу

int sum(ELEM\* List\_Row, int j); // Определение суммы элементов указанного столбца в нечетных строках

int main(){

ELEM \*List\_Row = NULL;

ELEM \*List\_Col = NULL;

bool flag = true;

int ans, i, j, num;

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

List\_Col = initCol(List\_Col, i);

List\_Row = initRow(List\_Row, i);

}

matrixForm(List\_Row, List\_Col);

while(flag){

cout << "\033[2J\033[1;1H";

cout << "1. Вывести матрицу на экран\n";

cout << "2. Получить значение элемента по номеру строки и столбца\n";

cout << "3. Определить суммы элементов указанного столбца в нечетных строках\n\n";

cout << "0. Выйти из программы\n\n >";

cin >> ans;

switch (ans)

{

case 1:

cout << "\033[2J\033[1;1H";

showMatrix(List\_Row);

cout << "\n\n Продолжить? (1-да, 2-нет)\n\n >";

cin >> ans;

if(ans == 2){

flag = false;

}

break;

case 2:

cout << "\033[2J\033[1;1H";

cout << " Введите номер строки и номер столбца\n";

cout << "Строка: ";

cin >> i;

cout << "Столбец: ";

cin >> j;

if(i > N || i < 0 || j > N || j < 0){

cout << "\n Такого элемента не существует!";

goto m1;

}

cout << "\n\n Ответ: " << getData(List\_Row, i, j);

m1: cout << "\n\n Продолжить? (1-да, 2-нет)\n\n >";

cin >> ans;

if(ans == 2){

flag = false;

}

break;

case 3:

cout << "\033[2J\033[1;1H";

cout << "Введите номер столбца: ";

cin >> num;

if(num > N || num < 0){

cout << "\n Такого столбца не существует!";

goto m2;

}

cout << "Ответ: " << sum(List\_Row, num);

m2: cout << "\n\n Продолжить? (1-да, 2-нет)\n\n >";

cin >> ans;

if(ans == 2){

flag = false;

}

break;

case 0:

flag = false;

break;

default:

break;

}

}

cout << "\033[2J\033[1;1H";

return 0;

}

int sum(ELEM\* List\_Row, int j){ // Определение суммы элементов указанного столбца в нечетных строках

int sum = 0;

int i = 1;

while(i <= N){

sum += getData(List\_Row, i, j);

i += 2;

}

return sum;

}

int getData(ELEM\* List\_Row, int i, int j){ // Поиск значения элемента матрицы по индексу

ELEM\* ListCpy = List\_Row;

while(ListCpy){

if(ListCpy->row == i){

while(ListCpy){

if(ListCpy->col == j){

return ListCpy->num;

} else ListCpy = ListCpy->right;

}

return 0;

}

ListCpy = ListCpy->down;

}

return 0;

}

ELEM\* initCol(ELEM \*List\_Col, int i){ //Создание списка колонок

ELEM \*tmp = new ELEM;

tmp->row = 0;

tmp->col = N-i;

tmp->right = List\_Col;

tmp->down = NULL;

List\_Col = tmp;

return List\_Col;

}

ELEM\* initRow(ELEM \*List\_Row, int i){ //Создание списка строк

ELEM \*tmp = new ELEM;

tmp->col = 0;

tmp->row = N-i;

tmp->down = List\_Row;

tmp->right = NULL;

List\_Row = tmp;

return List\_Row;

}

ELEM\* findCol(ELEM \*List\_Col, int i){ //Поиск колонки по индексу

while(List\_Col){

if(List\_Col->col == i){

while(List\_Col->down){

List\_Col = List\_Col->down;

}

return List\_Col;

}

List\_Col = List\_Col->right;

}

return NULL;

}

ELEM\* findRow(ELEM \*List\_Row, int i){ // Поиск строки по индексу

while(List\_Row){

if(List\_Row->row == i){

while(List\_Row->right){

List\_Row = List\_Row->right;

}

return List\_Row;

}

List\_Row = List\_Row->down;

}

return NULL;

}

int matrixForm(ELEM \*List\_Row, ELEM \*List\_Col){ // Чтение матрицы из файла

int i = 1; int j = 1, num;

FILE\* file;

file = fopen("matrix.txt", "r");

if(!file){

cout << "Не удалось открыть файл!";

return 0;

}

for(int i = 1; i <= N; i++){

for(int j = 1; j <= N; j++){

fscanf(file, "%d", &num);

if(num){

ELEM \*tmp = new ELEM;

tmp->num = num;

tmp->row = i;

tmp->col = j;

tmp->right = NULL;

tmp->down = NULL;

findCol(List\_Col, j)->down = tmp;

findRow(List\_Row, i)->right = tmp;

}

}

}

fclose(file);

return 0;

}

void showMatrix(ELEM \*List\_Row){ // Вывод матрицы на экран

ELEM\* temp, \*ListCpy = List\_Row;

int mass[N][N] = {0};

while(ListCpy){

temp = ListCpy->right;

while(temp){

mass[temp->row-1][temp->col-1] =temp->num;

temp = temp->right;

}

ListCpy = ListCpy->down;

}

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

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

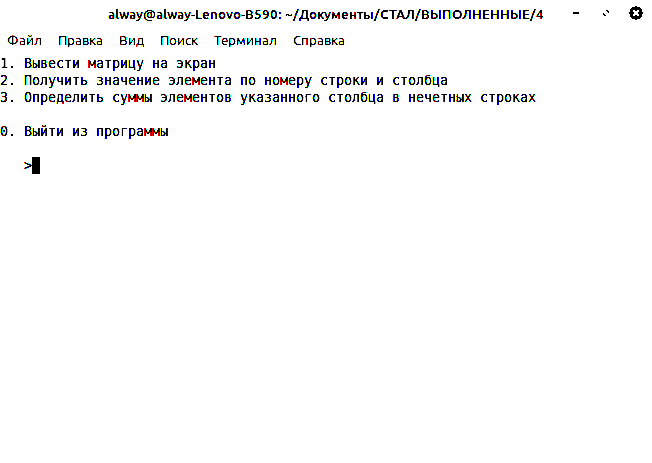
cout << mass[i][j] << " ";

}

cout << "\n";

}

}

2) ***Пример работы программы:***

