**МИНОБРНАУКИ РОССИИ**

**Санкт-Петербургский государственный**

**электротехнический университет**

**«ЛЭТИ» им. В.И. Ульянова (Ленина)**

**Кафедра САПР**

отчет

**по лабораторной работе №5**

**по дисциплине «Программирование»**

Тема: многомерные массивы

|  |  |  |
| --- | --- | --- |
| Студент гр. 1302 |  | Новиков Г.В. |
| Преподаватель |  | Калмычков В. А |

Санкт-Петербург

2021

**Оглавление**

[1. Исходная формулировка 3](#_Toc88692179)

[2. Особенности задания 3](#_Toc88692180)

[3. Контрольный пример 3](#_Toc88692181)

[4. Формальная постановка задачи 3](#_Toc88692182)

[5. Формат хранения данных 4](#_Toc88692183)

[Ограничения, обусловленные работой компьютера 4](#_Toc88692184)

[6. Макеты ввода/вывода 4](#_Toc88692185)

[7. Средства обеспечения ввода/вывода 5](#_Toc88692186)

[8. Алгоритм решения 5](#_Toc88692187)

[9. Текст программы 7](#_Toc88692188)

[10. Результаты работы программы 8](#_Toc88692189)

[11. Вывод 9](#_Toc88692190)

1. Исходная формулировка

На клеточном листе бумаги размером n × m расположены прямоугольники. Задана матрица размера n × m, в которой элемент aij = 1, если клетка листа (i, j) является частью прямоугольника, и aij = 0, если это пустая клетка. Напечатать число прямоугольников и координаты их вершин.

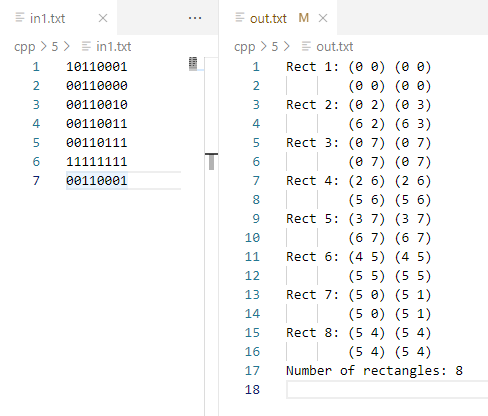
1. Особенности задания

Особенности, связанные с обработкой двумерных массивов

Программа должна реагировать на разные случаи:

* Если прямоугольники не касаются друг друга
* Если прямоугольники касаются друг друга и образуют форму “Г”
* Если прямоугольники касаются друг друга и образуют форму “Т”
* Если прямоугольники касаются друг друга и образуют форму “+”

1. Контрольный пример



1. Формальная постановка задачи

Дано: двумерный массив из 0 и 1

Вывести число прямоугольников и координаты их вершин

Способ решения:

1. rect\_count = 0 – количество прямоугольников
2. Запускаем цикл
3. Находим первую единицу слева сверху – начало прямоугольника
4. Находим вершины прямоугольника
5. Выводим его вершины
6. Удаляем его (заменяем все 1 в прямоугольнике на 0)
7. rect\_count++
8. выводим rect\_count

В итоге выведены все вершины прямоугольников и их количество

1. Формат хранения данных

|  |  |  |
| --- | --- | --- |
| **Переменная** | **Тип** | **Назначение** |
| filename | string | Имя входного файла |
| size | Size | Размер массива |
| n | int | Количество элементов в строке |
| m | int | Количество строк |
| rect\_map | int\*\* | Матрица с прямоугольниками |
| rect\_count | int | Количество прямоугольников |
| rect\_index | Index\_2d | Левая верхняя вершина найденного прямоугольника |
| rect | Rect | Представляет прямоугольник |
| file | fstream | Чтение файла |
| line | string | Хранит считанную строку |
| i | int | Итерация |
| j | int | Итерация |
| arr | \*\*int | Считывание матрицы из файла |
| row | \*int | Строка матрицы |

1. Ограничения, обусловленные работой компьютера

Тип int ограничен промежутком: [–32768 ; 32767]

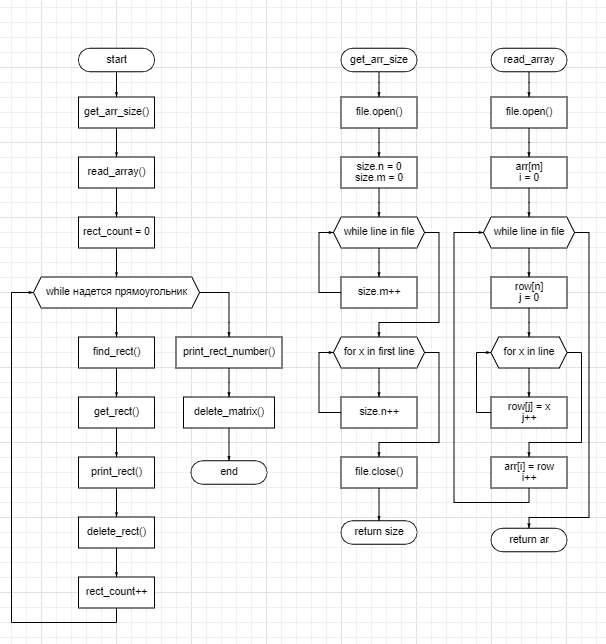
1. Макеты ввода/вывода

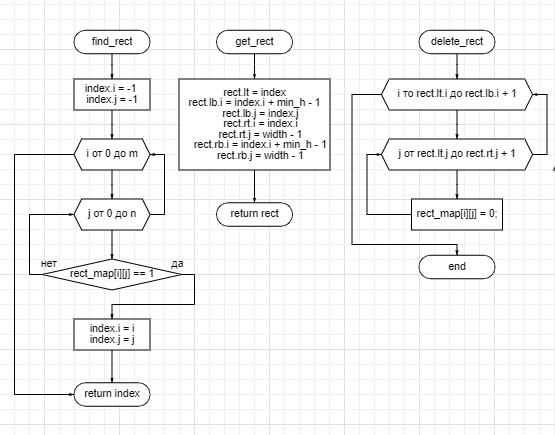
|  |  |  |  |
| --- | --- | --- | --- |
| № макета | Действие | Макет | Назначение |
|  | Вывод | Author: Novikov G.  Group: 1302  Start date: 21.11.2021  End date: 22.11.2021  Version: 5.1.01 | Приветствие |
|  | Входной файл | bbb…bbb  bbb…bbb  …….…...  bbb…bbb | Файл, содержащий входные данные (b - 1 или 0) |
|  | Выходной файл | Rect 1: (i i) (i i)  (i i) (i i)  Rect 2: (i i) (i i)  (i i) (i i)  ………………..  Rect n: (i i) (i i)  (i i) (i i)  Number of rectangles: n | Файл, содержащий выходные данные (i – целое число, n – число прямоугольников) |
|  | Вывод | Rect 1: (i i) (i i)  (i i) (i i)  Rect 2: (i i) (i i)  (i i) (i i)  ………………..  Rect n: (i i) (i i)  (i i) (i i)  Number of rectangles: n | Вывод в консоль |

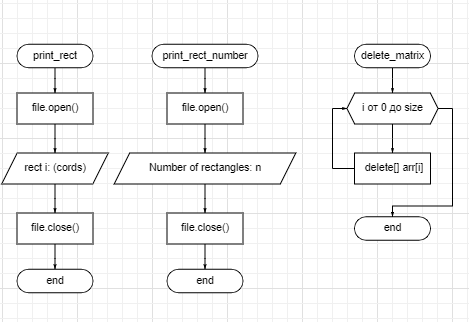
1. Средства обеспечения ввода/вывода

|  |  |
| --- | --- |
| getline() | Потоковый ввод из файла |
| cout << | Потоковый вывод в консоль |
| file << | Потоковый вывод в файл |

1. Алгоритм решения







1. Текст программы

#include <iostream>

#include <fstream>

#include <iomanip>

using namespace std;

struct ArrSize {

int n;

int m;

};

struct Index\_2d {

int i;

int j;

};

struct Rect {

Index\_2d lt;

Index\_2d rt;

Index\_2d lb;

Index\_2d rb;

};

ArrSize get\_arr\_size(string filename) {

fstream file;

file.open(filename, ios::in);

if (!file.is\_open()) {

perror("Error opening file");

}

ArrSize size;

size.n = 0;

size.m = 0;

string line;

while (getline(file, line)) {

size.m++;

}

file.clear();

file.seekg(0);

getline(file, line);

for (char x: line) size.n++;

file.close();

return size;

}

int\*\* read\_array(string filename, int n, int m) {

fstream file;

file.open("in1.txt", ios::in);

if (!file.is\_open()) {

perror("Error opening file");

}

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

int i = 0;

string line;

while (getline(file, line)) {

int\* row = new int[n];

int j = 0;

for(char x: line) {

row[j] = (int)(x - '0');

j++;

}

arr[i] = row;

i++;

}

return arr;

}

Index\_2d find\_rect(int\*\* rect\_map, int n, int m) {

Index\_2d index;

index.i = -1;

index.j = -1;

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

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

if (rect\_map[i][j] == 1) {

index.i = i;

index.j = j;

return index;

}

}

}

return index;

}

Rect get\_rect(int\*\* rect\_map, int n, int m, Index\_2d index) {

Rect rect;

rect.lt = index;

int min\_h = n + 1;

int i = index.i;

int j = index.j;

while (j < n && rect\_map[i][j] == 1) {

int i = index.i;

while (i < m && rect\_map[i][j] == 1) {

i++;

}

if (i - index.i < min\_h) min\_h = i - index.i;

j++;

}

rect.lb.i = index.i + min\_h - 1;

rect.lb.j = index.j;

rect.rt.i = index.i;

rect.rt.j = j - 1;

rect.rb.i = index.i + min\_h - 1;

rect.rb.j = j - 1;

return rect;

}

void delete\_rect(int\*\* rect\_map, Rect rect) {

for (int i = rect.lt.i; i < rect.lb.i + 1; i++) {

for (int j = rect.lt.j; j < rect.rt.j + 1; j++) {

rect\_map[i][j] = 0;

}

}

}

void print\_rect(Rect rect, int rect\_number, bool erase = false) {

fstream file;

if (erase) {

file.open("out.txt", ios::out);

} else {

file.open("out.txt", ios::app);

}

if (!file.is\_open()) {

perror("Cannot open file 'out.txt'");

}

cout << "Rect " << rect\_number << ": (" << rect.lt.i << ' ' << rect.lt.j << ") (" << rect.rt.i << ' ' << rect.rt.j << ')' << endl;

cout << setw(9) << "(" << rect.lb.i << ' ' << rect.lb.j << ") (" << rect.rb.i << ' ' << rect.rb.j << ')' << endl;

file << "Rect " << rect\_number << ": (" << rect.lt.i << ' ' << rect.lt.j << ") (" << rect.rt.i << ' ' << rect.rt.j << ')' << endl;

file << setw(9) << "(" << rect.lb.i << ' ' << rect.lb.j << ") (" << rect.rb.i << ' ' << rect.rb.j << ')' << endl;

file.close();

}

void print\_rect\_number(int rect\_number, bool erase = false) {

fstream file;

if (erase) {

file.open("out.txt", ios::out);

} else {

file.open("out.txt", ios::app);

}

if (!file.is\_open()) {

perror("Cannot open file 'out.txt'");

}

cout << "Number of rectangles: " << rect\_number << endl;

file << "Number of rectangles: " << rect\_number << endl;

file.close();

}

void delete\_matrix(int\*\* arr, int size) {

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

delete[] arr[i];

}

}

// Rectangles can touch each other

// Part of rectangle cannot be part of another rectangle

// Not rectangles:

// '+'-forms(3 rects)

// 'T'-forms(2 rects)

int main(int argc, char const \*argv[]) {

cout << "Author: Novikov G. \n"

"Group: 1302 \n"

"Start date: 21.11.2021 \n"

"End date: 22.11.2021 \n"

"Version: 5.1.01 \n" << endl;

string filename = "in1.txt";

ArrSize size = get\_arr\_size(filename);

int n = size.n;

int m = size.m;

int\*\* rect\_map = read\_array("in1.txt", n, m);

int rect\_count = 0;

while (find\_rect(rect\_map, n, m).i >= 0) {

Index\_2d rect\_index = find\_rect(rect\_map, n, m);

Rect rect = get\_rect(rect\_map, n, m, rect\_index);

delete\_rect(rect\_map, rect);

if (rect\_count == 0) {

print\_rect(rect, rect\_count + 1, true);

} else {

print\_rect(rect, rect\_count + 1);

}

rect\_count++;

}

print\_rect\_number(rect\_count);

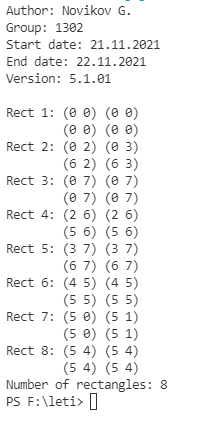
delete\_matrix(rect\_map, m);

return 0;

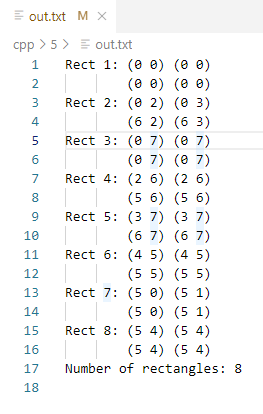
}

1. **Результаты работы программы**

*Console:*



*Out.txt:*



1. Вывод

В процессе выполнения данной работы я успешно организовал работу с двумерным массивом, ввод из файла и вывод в него. Написанная мной программа выдала правильные результаты, совпавшие с контрольным примером.