Министерство образования Российской Федерации

Пензенский государственный университет

Кафедра «Вычислительная техника»

**Пояснительная записка**

По курсовому проектированию

По курсу «Логика и основы алгоритмизации в инженерных задачах»

На тему «Поиск наибольшего паросочетания»

Выполнил:

Студент группы 20ВВ3

Баулин А.Д.

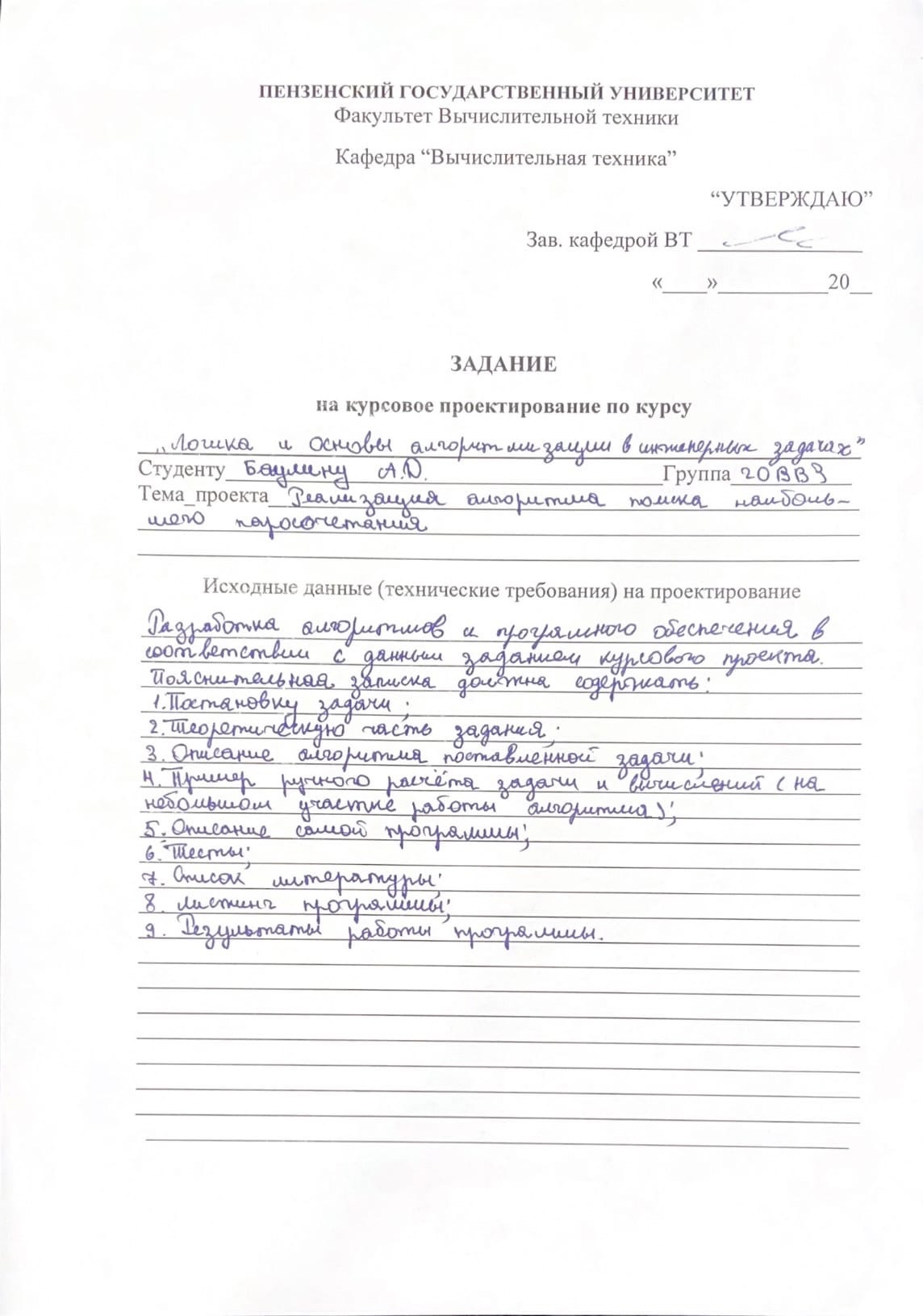
Приняли:

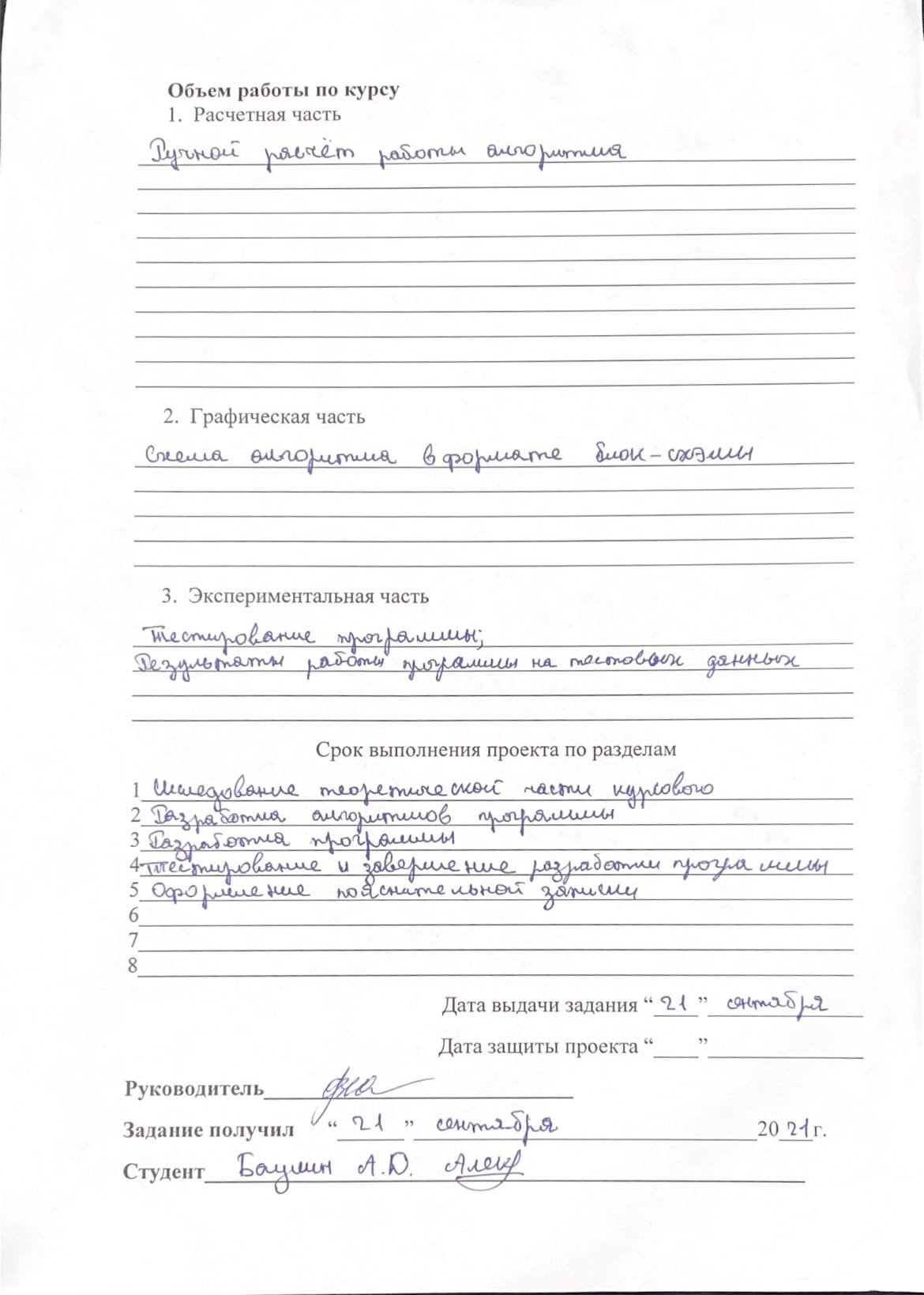
к.т.н., доцент

Митрохин М.А.

Юрова О.В.

Пенза 2022

****



**Содержание**

[Реферат 4](#_Toc115127654)

[Введение 5](#_Toc115127655)

[1.Постановка задачи 6](#_Toc115127656)

[2.Теоретическая часть задания 7](#_Toc115127657)

[3.Описание алгоритма программы 8](#_Toc115127658)

[4.Описание программы 11](#_Toc115127659)

[5.Тестирование 19](#_Toc115127660)

[Заключение 24](#_Toc115127661)

[Список литературы 25](#_Toc115127662)

[Приложение А. 26](#_Toc115127663)

# Реферат

Отчет 35 стр, 19 рисунков.

ГРАФ, ТЕОРИЯ ГРАФОВ, ПОИСК НАИБОЛЬШЕГО ПАРОСОЧЕТАНИЯ.

.

Цель исследования – разработка программы, способная найти наибольшее паросочетание.

В данной работе разработан алгоритм, который находит наибольшее паросочетание в матрице смежности.

# Введение

Термин «граф» впервые был введен в 1936 г Д. Кенигом.

**Граф -**это топологичекая модель, которая состоит из множества вершин и множества соединяющих их рёбер. При этом значение имеет только сам факт, какая вершина с какой соединена.

Теория графов стала активно применяться в программировании одновременно с использованием ЭВМ в силу удобного выражения задач обработки информации на теоретико - графовом языке. Среди первых работ, существенно использующих теоретико-графовые методы в решении задач программирования, можно отметить работы по организации вычисления арифметических выражений и оптимальному использованию оперативной памяти, а также была введена теоретико-графовая модель программы в виде управляющего графа. Эта модель стала к настоящему времени классической для решения задач трансляции и конструирования программ.

В качестве среды разработки мною была выбрана среда MicrosoftVisualStudio2022, язык программирования – Си.

Целью данной курсовой работы является разработка программы на языке С++, который является широко используемым.

# 1.Постановка задачи

Требуется разработать программу, которая находит наибольшее паросочетание.

Исходный граф в программе должен задаваться матрицей смежности, причѐм при генерации данных должны быть предусмотрены граничные условия. Программа должна работать так, чтобы пользователь сначала выбирал ввод из файла, либо генерацию случайными числами, либо заполнение вручную. В случае случайной или ручной генерации вводил размер для генерации матрицы смежности. После обработки этих данных на экран должна выводиться матрица смежности графа.

Устройство ввода – клавиатура и мышь.

Задания выполняются в соответствии с вариантом №27.

# 2.Теоретическая часть задания

Граф G (рисунок 1) задается множеством вершин X1, X2, ..., Xn. и множеством ребер, соединяющих между собой определенные вершины[[2](#Источник2)].

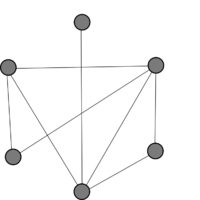


Рисунок 0– Пример графа

При представлении графа матрицей смежности информация о ребрах графа хранится в квадратной матрице, где присутствие пути из одной вершины в другую обозначается единицей, иначе нулем.

**Граф -**это топологичекая модель, которая состоит из множества вершин и множества соединяющих их рёбер. При этом значение имеет только сам факт, какая вершина с какой соединена.

Паросочетание —  это набор попарно несмежных рёбер.

Максимальное паросочетание — это такое паросочетание M в графе G, которое не содержится ни в каком другом паросочетании этого графа.

Мощность паросочетания — количество рёбер в графе.

Наибольшее паросочетание — паросочетание, мощность которого максимальна.

# 3.Описание алгоритма программы

Для программной реализации алгоритма понадобиться 5 массивов: vis (int) – для хранения информации о посещенных вершинах (1 или 0), set (int), в который будут заноситься найденные номера рёбер максимальных паросочетаний, vertex (int) – исходную матрицу, max\_mas (int) – хранит матрицу смежности рёбер и num\_reb (int) – хранит номера рёбер и инцидентные им вершины.

Алгоритм начинается с цикла, который проверяет имеется ли в строке матрицы смежности несмежные рёбра и посещено ли оно. Если таковы имеются, то срабатывает счётчик sled и помещает это ребро в массив set, затем в массив vis добавляется это ребро как посещённое. Далее в цикле повторяется данная процедура столько раз, сколько имеется вершин. После завершения работы цикла счётчик sled сравнивается с переменной most\_set, в которой хранится максимальное количество паросочетаний. Если в sled число больше, то она заносится в most\_set. В вектор most\_set записывается значение из массива set. Затем алгоритм повторяет действие с следующей строчкой.

Ниже представлены псевдокод функций, которые осуществляют поиск максимальных паросочетаний в программе.

1. алг matrix\_inc
2. начало
3. mass = динамическое выделение памяти для хранения матрицы смежности реберного графа
4. цикл от i=0 до i<num, i++
5. цикл от j=0 до j<num, j++
6. mass[i][j] = 0
7. конец цикла
8. конец цикла
9. iter = 0
10. цикл от i=0 до i<count i++
11. цикл от j=i до j<count, j++
12. если mass[i][j]=1 то
13. num\_reb[0][iter] = iter
14. num\_reb [1][iter] = i
15. num\_reb [2][iter] = j
16. iter++
17. конец условия
18. конец цикла
19. конец цикла
20. цикл от i=0 до i<num, i++
21. цикл от j=i+1 до j<num, j++
22. если (mass[1][i]=mass[1][j]
23. mass[1][i]= mass[2][j]
24. mass[2][i]=mass[1][j]
25. mass[2][i]=mass[2][j])
26. mass[i][j] = 1
27. mass [j][i] = 1
28. конец условия
29. конец цикла
30. конец цикла
31. возврат значения mass
32. конец
33. алг most
34. начало
35. цикл от i=0 до i<num i++
36. если mass[start][i]=0 и vis[i]=0 то
37. schet = 0
38. цикл от j=0 до j<sled+1, j++
39. если mass[set[j]][i]=0 то
40. schet++
41. конец условия
42. конец цикла
43. если schet=sled+1 то
44. sled++
45. set[sled] = i
46. vis[i] = 1
47. конец условия
48. конец условия
49. конец цикла
50. если sled=most\_step то
51. vec\_iter++
52. цикл от i=0 до i<step+1, i++
53. запись во внутренний вектор элемента массива - set[i]
54. конец цикла
55. конец условия
56. если sled > most\_step то
57. most\_set = 0
58. vec\_iter = 0
59. цикл от i=0 до i<sled+1, i++
60. запись во внутренний вектор элемента массива – set[i]
61. конец цикла
62. most\_step = sled
63. конец условия
64. vis[set[sled]] = 0
65. sled--
66. возврат значения sled
67. конец

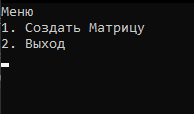
# 4.Описание программы

Для написания данной программы использован язык программирования С++.Язык программирования С++ - универсальный язык программирования, который завоевал особую популярность у программистов, благодаря сочетанию возможностей языков программирования высокого и низкого уровней.

Проект был создан в виде консольного приложения Win32 (Visual C++).

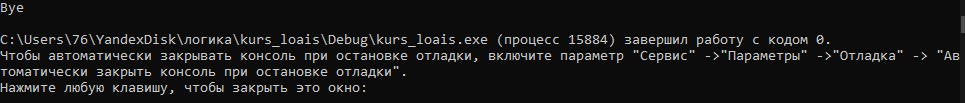
Данная программа является многомодульной, поскольку состоит из нескольких функций: Create, save\_in\_file, read\_from\_file, Free, Input, FillRandomNumbers, Print, menu, search, null, most, matrix\_inc.

Работа программы начинается с главного меню имеющего два пункта: Выход и Создать матрицу.



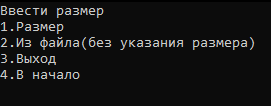
*Рис.1- Главное меню*

Пункт Выход будет встречаться на протяжении всей работы программы, он завершает программу.



*Рис.2- Выход*

Пункт Создать матрицу открывает следующее меню, имеющее пункты Размер, Из файла, Выход, В начало.



*Рис.3- Создание матрицы*

Пункт в начало очищает динамические массивы, присваивает им значения NULL и переходит в главное меню.

Free(A, n, vis, set);

null(A, vis, set);

Оставшиеся 2 пункта отвечают за вид загрузки графа. Предлагается загрузить матрицу из файла (п. Из файла), либо заполнить случайными числами (п. Вручную), либо заполнить вручную (п. Вручную). Если пользователь выбирает загрузку из файла, происходит переход к указанию имени файла, файл указывается полностью, например: file.txt.



*Рис.4- Пример указания имени файла*

После вызывается функция read\_from\_file, размер матрицы не указывается, так как он считывается с файла. После происходит переход к меню Работа с матрицей.

cout << "Имя файла" << endl;

cin >> str;

f.open(str);

f >> n;

A = Create(n);

ifstream f(str);

size\_t trash = NULL;

if (f.is\_open())

{

f >> trash;

for (size\_t i = 0; i < n; ++i)

{

for (size\_t j = 0; j < n; ++j)

{

f >> M[i][j];

}

}

}

else

{

cout << "Ошибка: файл(запись)" << endl;

}

f.close();

В случае выбора пункта Размер происходит переход к выбору размера матрицы



*Рис.5- Пример указания размера матрицы*

Далее открывается меню заполнения имеющее пункты загрузки матрицы случайными числами(п. Авто) и вручную(п. Вручную).



*Рис.6- Меню заполнения матрицы*

При выборе п. Авто вызывается функция FillRandomNumbers, которая заполняет матрицу случайными числами.

srand(time(NULL));

int z = 1;

for (size\_t i = 0; i < n; i++)

{

M[i][i] = 0;

for (size\_t j = z; j < n; j++)

{

if (1 + rand() % 100 < 25)

{

M[i][j] = 1;

}

else

{

M[i][j] = 0;

}

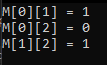
M[j][i] = M[i][j];

}

z++;

}

При выборе загрузки вручную(п. Вручную) вызывается функция Input(заполняется только одна из частей матрицы так как вторая полностью идентична).



*Рис.7- Пример ручного заполнения матрицы*

int z = 1;

for (size\_t i = 0; i < n; ++i)

{

M[i][i] = 0;

for (size\_t j = z; j < n; ++j)

{

cout << "M[" << i << "][" << j << "] = ";

cin >> M[i][j];

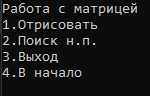
M[j][i] = M[i][j];

}

z++;

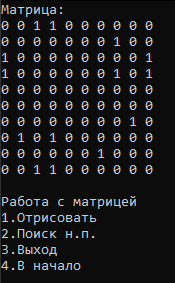
}

После этого на экран выводится меню Работы с матрицей, имеющее пункты Отрисовать, Поиск н.п.(поиск наибольшего парочосетания).



*Рис.8- Меню работы с матрицей*

Пункт Отрисовать запускает функцию Print, он отрисовывает матрицу .



*Рис.9- Отрисовка*

for (size\_t i = 0; i < n; ++i)

{

for (size\_t j = 0; j < n; ++j)

{

cout << M[i][j] << ' ';

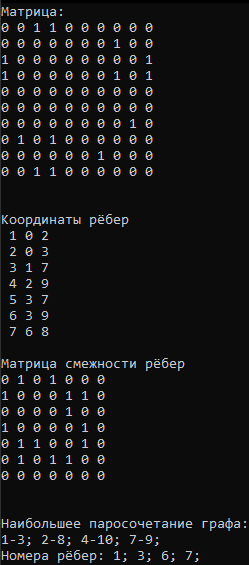
}

cout << endl;

}

cout << endl;

Пункт Поиск н.п. запускает функцию Print, для отрисовки, после запускает функцию search, которая выводит координаты ребер, матрицу смежности ребер и наибольшее паросочетание.



*Рис.10- Поиск наибольшего паросочетания*

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

{

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

{

if (A[i][j] == 1) {

max++;

}

}

}

num\_reb = new int\* [3];

for (int i = 0; i < 3; i++)

{

num\_reb[i] = new int[max];

}

max\_mas = matrix\_inc(A, num\_reb, n, &max);

cout << "\nМатрица смежности рёбер" << endl;

Print(max\_mas, max);

vis = new int[max];

set = new int[max];

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

{

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

{

vis[j] = 0;

}

vis[i] = 1;

set[sled] = i;

sled = most(i, max\_mas, vis, set, &max, &max\_shag, i, sled, most\_set, &x);

sled = 0;

}

ofstream fout;

fout.open("data.txt");

cout << "\nНаибольшее паросочетание графа:" << endl;

i = most\_set.begin();

for (vector<int>::iterator j = (\*i).begin(); j != (\*i).end(); ++j)

{

cout << num\_reb[1][(\*j)] + 1 << "-" << num\_reb[2][(\*j)] + 1 << "; ";

fout << num\_reb[1][(\*j)] + 1 << " " << num\_reb[2][(\*j)] + 1 << endl;

max\_size++;

}

fout.close();

cout << "\nНомера рёбер: ";

for (vector<int>::iterator j = (\*i).begin(); j != (\*i).end(); ++j)

{

cout << \*j + 1 << "; ";

}

cout << endl;

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

delete[] max\_mas[i];

delete[] max\_mas;

max\_mas = NULL;

for (int i = 0; i < 3; i++)

delete[] num\_reb[i];

delete[] num\_reb;

num\_reb = NULL;

После выполнения функции search, открывается последнее меню имеющее пункты Сохранить, Выход, В начало.



*Рис.11- Меню сохранения*

Пункт сохранить запускает функцию save\_in\_file и сохраняет в файл (имя которого указывается в ручную) матрицу смежности вершин и наибольшее паросочетание ребер.



*Рис.12- Пример указания имени файла для сохранения*

ofstream f(str);

ifstream fout("data.txt");

int\*\* write = new int\* [2];

for (int i = 0; i < 2; i++)

{

write[i] = new int[max\_size];

}

if (f.is\_open())

{

f << "Матрица" << endl;

for (size\_t i = 0; i < n; ++i)

{

for (size\_t j = 0; j < n; ++j)

{

f << M[i][j] << ' ';

}

f << endl;

}

f << "\nНаибольшее паросочетание графа:" << endl;

for (int i = 0; i < max\_size; i++)

{

fout >> write[0][max\_size];

f << write[0][max\_size] << "-";

fout >> write[1][max\_size];

f << write[1][max\_size] << endl;

}

}

else

{

cout << "Ошибка: файл(сохранение)" << endl;

}

fout.close();

f.close();

При запуске функции search запускается цикл просчёта единиц в матрице (количества рёбер).

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

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

if (vertex[i][j] == 1) {

max++;

}

}

Далее происходит вызов функции matrix\_inc, которая формирует матрицу, размер которой определяет количество рёбер и заполняется нулями. Затем формируется массив 3 на max. В него записываются номера рёбер и вершины, между которыми он находится.

mass\_edge = (int\*\*)malloc(\*num \* sizeof(int\*));

for (int i = 0; i < \*num; i++) {

mass\_edge[i] = (int\*)malloc(\*num \* sizeof(int));

}

for (int i = 0; i < \*num; i++) {

for (int j = 0; j < \*num; j++) {

mass\_edge[i][j] = 0;

}

}

cout << "\nКоординаты рёбер\n";

int iter = 0;

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

for (int j = i; j < count; j++) {

if (mass[i][j] == 1) {

num\_reb[0][iter] = iter;

num\_reb[1][iter] = i;

num\_reb[2][iter] = j;

iter++;

cout << " " << iter << " " << i << " " << j << " \n";

}

}

# 5.Тестирование

Среда разработки MicrosoftVisualStudio 2022 предоставляет все средства, необходимые при разработке и отладке многомодульной программы.

Тестирование проводилось в рабочем порядке, в процессе разработки, после завершения написания программы. В ходе тестирования было выявлено и исправлено множество проблем, связанных с вводом данных, изменением дизайна выводимых данных, алгоритмом программы, взаимодействием функций.

Ниже продемонстрирован результат тестирования программы при вводе пользователем различных количеств вершин и вывод максимальных паросочетаний(Рис.6).

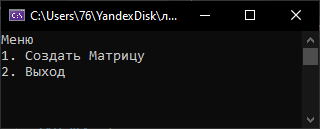


Рисунок 14 - Результат работы программы при загрузке матрицы из файла

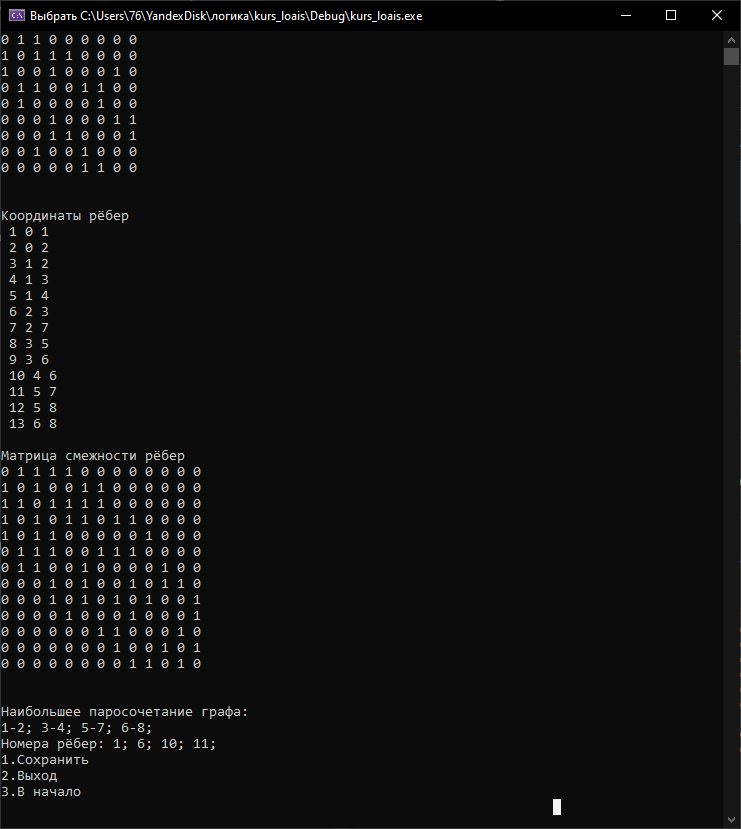


Рисунок 14 - Результат работы программы при загрузке матрицы из файла

Вывод результата работы программы для матрицы размерности 10 (Рис.7)

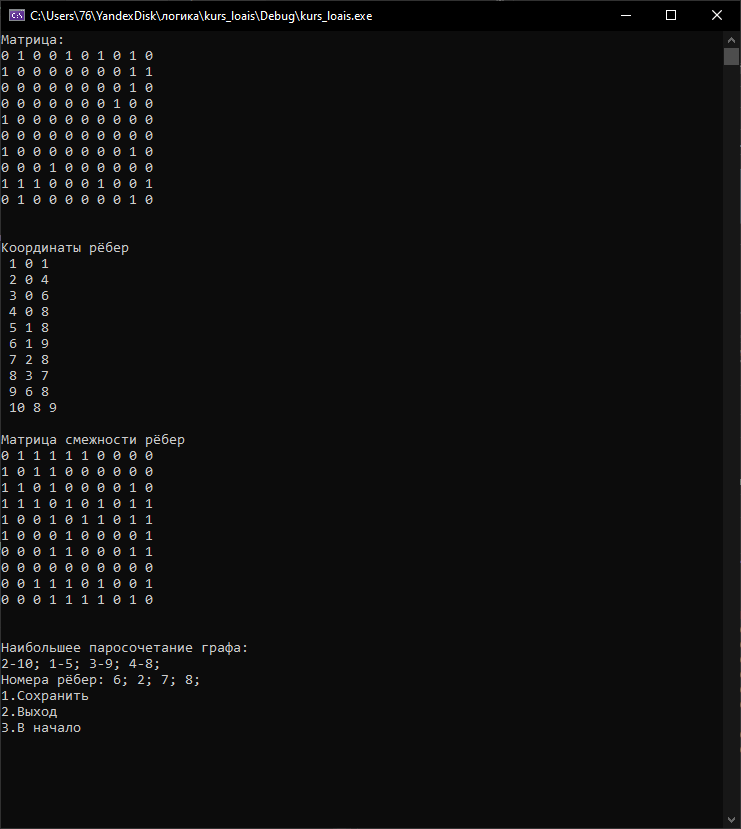


Рисунок 15 - Результат работы программы для матрицы размерности 10

Произведём вывод работы программы для размерности матрицы 12 (Рис 8)

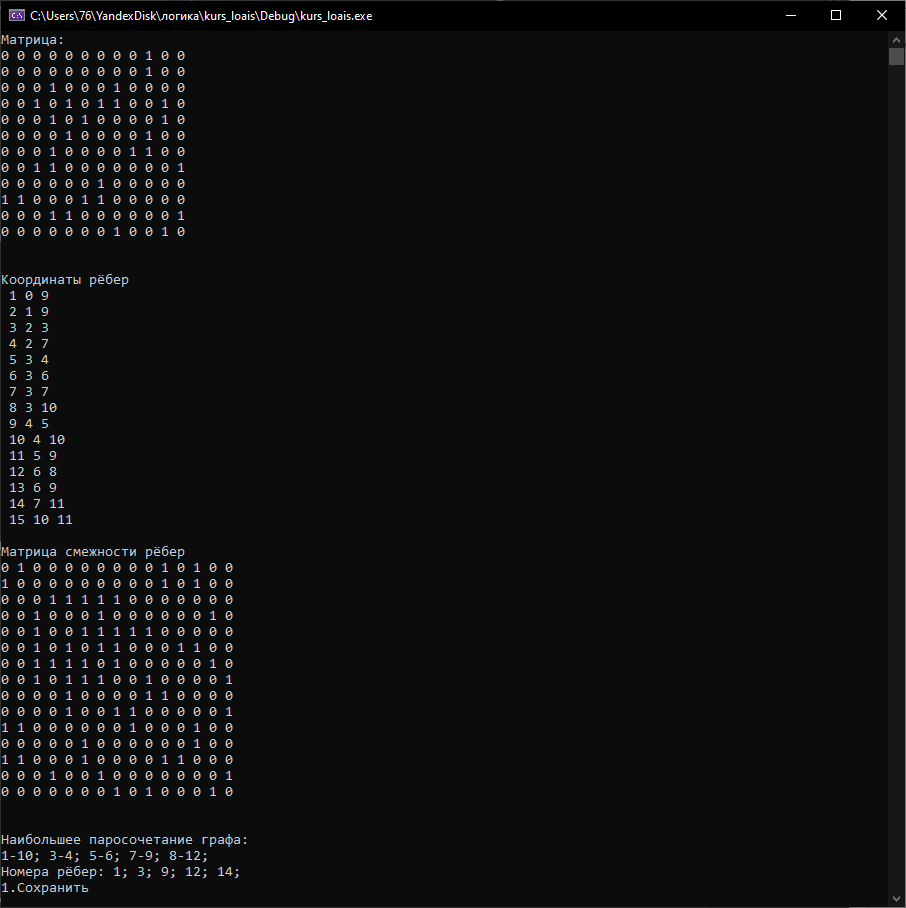


Рисунок 16 - вывод работы программы для размерности матрицы 12

|  |  |  |
| --- | --- | --- |
| Описание теста | Ожидаемый результат | Полученный результат |
| Запуск программы | Вывод главного меню. | Верно |
| Результат поиска наибольшего паросочетания | Вывод результата | Верно |

В результате тестирования было выявлено, что программа успешно работает и выводит правильный результат.

**6.Ручной расчѐт задачи**

Проведем проверку программы посредством ручных вычислений на примере графа с 5 вершинами(рис.6).

Представим матрицу в виде рисунка(Рис.9).

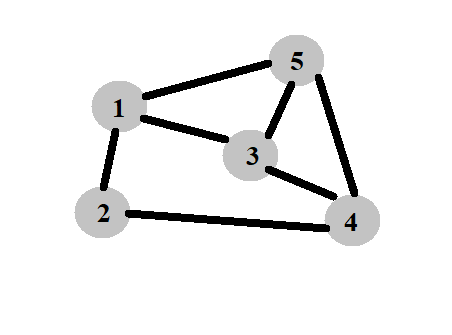


Рисунок 17 - граф загружаемого файла

Пронумеруем все рёбра в графе(Рис.10).

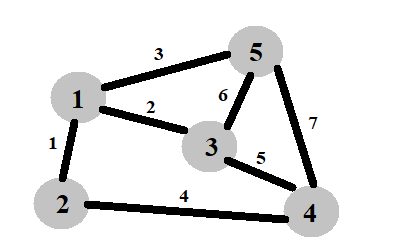


Рисунок 18 - нумерация рёбер графа

Соответственно рёбра и принадлежащие им инцидентные вершины получаются следующими:

1 ребро = 1-2

2 ребро = 1-3

3 ребро = 1-5

4 ребро = 2-4

5 ребро = 3-4

6 ребро = 3-5

7 ребро = 4-5

На основании расчётов составим матрицу смежности рёбер:

Смежные рёбра для первого: 0 1 1 0 1 0 0

Смежные рёбра для второго: 1 0 0 1 0 0 0

Смежные рёбра для третьего: 1 1 0 0 0 1 1

Смежные рёбра для четвёртого: 1 0 0 0 1 0 1

Смежные рёбра для пятого: 0 1 0 1 0 1 1

Смежные рёбра для шестого: 0 1 1 0 1 0 1

Смежные рёбра для седьмого: 0 0 1 1 1 1 0

Рассчитаем максимальное паросочетание:

Сравниваем 1 ребро с остальными. Оно смежно 2, 3, 4 ребру. Они не посещены, соответственно отмечаем вершины 1 и 2 в массиве. Ребра 2, 3 и 4 смежны 1, поэтому пропускаем их, ребро 5 не смежно 1, по этому отмечаем второе паросочетание 3-4. Ребра 6 и 7 смежны 5, следовательно они нам не подходят. В результате наше наибольшее паросочетание 1-2, 3-4.

# Заключение

Таким образом, в процессе создания данного проекта разработана программа, реализующая алгоритм поиска наибольшего паросочетания MicrosoftVisualStudio 2022.

При выполнении данной курсовой работы были получены навыки разработки программ и освоены приемы создания матриц смежностей, основанных на теории графов. Приобретены навыки по осуществлению поиска наибольшего паросочетания. Углублены знания языка программирования C++.

Недостатком разработанной программы является примитивный пользовательский интерфейс. Потому что программа работает в консольном режиме, не добавляющем к сложности языка сложность программного оконного интерфейса.

Программа имеет достаточный для использования функционал возможностей.

# Список литературы

* 1. <https://algorithmica.org/ru/matching>
  2. Стивен Прата. Язык программирования C++.
  3. https://e-maxx.ru/algo/kuhn\_matching
  4. 4. Роберт Седжвик: Алгоритмы на C++.
  5. Керниган Б. Ритчи Д. Язык программирования С. 1985 г.

# Приложение А.

**Листинг программы.**

Файл **head.h**

#pragma once

#define \_CRT\_SECURE\_NO\_WARNINGS

#include <conio.h>

#include <stdlib.h>

#include <stdio.h>

#include <time.h>

#include <stdbool.h>

#include <vector>

#include <iostream>

#include <clocale>

#include <fstream>

using namespace std;

int\*\* Create(size\_t n);

void save\_in\_file(string str, int\*\* M, size\_t n, int\*\* max\_mas, int max, int\*\* num\_reb, int max\_size);

void read\_from\_file(string str, int\*\* M, size\_t n);

void Free(int\*\* M1, size\_t n, int\* m1, int\* m2);

void Input(int\*\* M, size\_t n);

void FillRandomNumbers(int\*\* matrix, size\_t n);

void Print(int\*\* M, size\_t n);

void menu();

void search(vector<vector<int>>::iterator i, vector<vector<int>> most\_set, int max, int\*\* A, int sled, int max\_shag, int\* vis, int\* set, int\*\* num\_reb, int\*\* max\_mas, int n, int x, int &max\_size);

int\*\* matrix\_inc(int\*\* mass, int\*\* num\_reb, int count, int\* num);

int most(int i, int\*\* mass, int\* vis, int\* set, int\* num, int\* most\_step, int start, int sled, vector<vector<int>>& most\_set, int\* vec\_iter);

void null(int\*\* M1, int\* m1, int\* m2);

Файл **res.cpp**

#include "head.h"

void save\_in\_file(string str, int\*\* M, size\_t n, int\*\*max\_mas, int max, int\*\* num\_reb,int max\_size)

{

ofstream f(str);

ifstream fout("data.txt");

int\*\* write = new int\* [2];

for (int i = 0; i < 2; i++)

{

write[i] = new int[max\_size];

}

if (f.is\_open())

{

f << "Матрица" << endl;

for (size\_t i = 0; i < n; ++i)

{

for (size\_t j = 0; j < n; ++j)

{

f << M[i][j] << ' ';

}

f << endl;

}

f << "\nНаибольшее паросочетание графа:" << endl;

for (int i = 0; i < max\_size; i++)

{

fout >> write[0][max\_size];

f << write[0][max\_size] << "-";

fout >> write[1][max\_size];

f << write[1][max\_size] << endl;

}

}

else

{

cout << "Ошибка: файл(сохранение)" << endl;

}

fout.close();

f.close();

}

void read\_from\_file(string str, int\*\* M, size\_t n)

{

ifstream f(str);

size\_t trash = NULL;

if (f.is\_open())

{

f >> trash;

for (size\_t i = 0; i < n; ++i)

{

for (size\_t j = 0; j < n; ++j)

{

f >> M[i][j];

}

}

}

else

{

cout << "Ошибка: файл(запись)" << endl;

}

f.close();

}

int\*\* Create(size\_t n)

{

int\*\* M = new int\* [n];

for (size\_t i = 0; i < n; ++i) {

M[i] = new int[n];

}

return M;

}

void Free(int\*\* M1, size\_t n, int\* m1, int\* m2)

{

for (size\_t i = 0; i < n; ++i) {

delete[] M1[i];

}

delete[] m1;

delete[] m2;

delete[] M1;

}

void null(int\*\* M1, int\* m1, int\* m2)

{

M1 = NULL;

m1 = NULL;

m2 = NULL;

}

void Input(int\*\* M, size\_t n)

{

int z = 1;

for (size\_t i = 0; i < n; ++i)

{

M[i][i] = 0;

for (size\_t j = z; j < n; ++j)

{

cout << "M[" << i << "][" << j << "] = ";

cin >> M[i][j];

M[j][i] = M[i][j];

}

z++;

}

}

void FillRandomNumbers(int\*\* M, size\_t n)

{

srand(time(NULL));

int z = 1;

for (size\_t i = 0; i < n; i++)

{

M[i][i] = 0;

for (size\_t j = z; j < n; j++)

{

if (1 + rand() % 100 < 25)

{

M[i][j] = 1;

}

else

{

M[i][j] = 0;

}

M[j][i] = M[i][j];

}

z++;

}

}

void Print(int\*\* M, size\_t n)

{

for (size\_t i = 0; i < n; ++i)

{

for (size\_t j = 0; j < n; ++j)

{

cout << M[i][j] << ' ';

}

cout << endl;

}

cout << endl;

}

void search(vector<vector<int>>::iterator i, vector<vector<int>> most\_set, int max, int\*\* A, int sled, int max\_shag, int \*vis, int \*set, int\*\* num\_reb, int\*\*max\_mas, int n, int x, int &max\_size)

{

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

{

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

{

if (A[i][j] == 1) {

max++;

}

}

}

num\_reb = new int\* [3];

for (int i = 0; i < 3; i++)

{

num\_reb[i] = new int[max];

}

max\_mas = matrix\_inc(A, num\_reb, n, &max);

cout << "\nМатрица смежности рёбер" << endl;

Print(max\_mas, max);

vis = new int[max];

set = new int[max];

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

{

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

{

vis[j] = 0;

}

vis[i] = 1;

set[sled] = i;

sled = most(i, max\_mas, vis, set, &max, &max\_shag, i, sled, most\_set, &x);

sled = 0;

}

ofstream fout;

fout.open("data.txt");

cout << "\nНаибольшее паросочетание графа:" << endl;

i = most\_set.begin();

for (vector<int>::iterator j = (\*i).begin(); j != (\*i).end(); ++j)

{

cout << num\_reb[1][(\*j)] + 1 << "-" << num\_reb[2][(\*j)] + 1 << "; ";

fout << num\_reb[1][(\*j)] + 1 << " " << num\_reb[2][(\*j)] + 1 << endl;

max\_size++;

}

fout.close();

cout << "\nНомера рёбер: ";

for (vector<int>::iterator j = (\*i).begin(); j != (\*i).end(); ++j)

{

cout << \*j + 1 << "; ";

}

cout << endl;

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

delete[] max\_mas[i];

delete[] max\_mas;

max\_mas = NULL;

for (int i = 0; i < 3; i++)

delete[] num\_reb[i];

delete[] num\_reb;

num\_reb = NULL;

}

int\*\* matrix\_inc(int\*\* mass, int\*\* num\_reb, int count, int\* num)

{

int\*\* mass\_edge=new int\*[\*num];

for (int i = 0; i < \*num; i++)

{

mass\_edge[i] = new int[\*num];

}

for (int i = 0; i < \*num; i++) {

for (int j = 0; j < \*num; j++) {

mass\_edge[i][j] = 0;

}

}

cout << "\nКоординаты рёбер\n";

int iter = 0;

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

for (int j = i; j < count; j++) {

if (mass[i][j] == 1) {

num\_reb[0][iter] = iter;

num\_reb[1][iter] = i;

num\_reb[2][iter] = j;

iter++;

cout << " " << iter << " " << i << " " << j << " \n";

}

}

}

for (int i = 0; i < \*num; i++)

{

for (int j = i + 1; j < \*num; j++)

{

if ((num\_reb[1][i] == num\_reb[1][j]) || (num\_reb[1][i] == num\_reb[2][j]) || (num\_reb[2][i] == num\_reb[1][j]) || (num\_reb[2][i] == num\_reb[2][j]))

{

mass\_edge[i][j] = 1;

mass\_edge[j][i] = 1;

}

}

}

return mass\_edge;

}

int most(int i, int\*\* mass, int\* vis, int\* set, int\* num, int\* most\_step, int start, int sled, vector<vector<int>>& most\_set, int\* vec\_iter)

{

for (i = 0; i < \*num; i++) {

if (mass[start][i] == 0 && vis[i] == 0)

{

int schet = 0;

for (int j = 0; j < sled + 1; j++)

{

if (mass[set[j]][i] == 0)

{

schet++;

}

}

if (schet == sled + 1)

{

sled++;

set[sled] = i;

vis[i] = 1;

}

}

}

if (sled > \*most\_step)

{

most\_set.clear();

\*vec\_iter = 0;

most\_set.push\_back(vector<int>());

for (int i = 0; i < sled + 1; i++)

{

most\_set[\*vec\_iter].push\_back(set[i]);

}

\*most\_step = sled;

}

vis[set[sled]] = 0;

sled--;

return sled;

}

Файл **menu.cpp**

#include "head.h"

void menu()

{

vector<vector<int>>::iterator i;

vector<vector<int>> most\_set;

ifstream f;

int max = 0, sled = 0, max\_shag = -1, x = 0, max\_size=0;

int\* vis=NULL, \* set=NULL;

int\*\* num\_reb = NULL, \*\* max\_mas = NULL;

int exit\_key = 0, k = 0;

int\*\* A = NULL;

string str;

size\_t n = NULL;

while (exit\_key != 1)

{

exit\_key = 0;

cout << "Меню" << endl << "1. Создать Матрицу" << endl << "2. Выход" << endl;

cin >> k;

system("cls");

switch (k)

{

case 1:

while ((exit\_key != 2) && (exit\_key != 1) && (exit\_key != 3) && (exit\_key != 5))

{

cout << "Ввести размер" << endl << "1.Размер" << endl << "2.Из файла(без указания размера)" << endl << "3.Выход" << endl << "4.В начало" << endl;

cin >> k;

system("cls");

switch (k)

{

case 1:

cout << "Размер= " << endl;

cin >> n;

A = Create(n);

system("cls");

exit\_key = 2;

break;

case 2:

cout << "Имя файла" << endl;

cin >> str;

f.open(str);

f >> n;

A = Create(n);

read\_from\_file(str, A, n);

f.close();

system("cls");

exit\_key = 3;

break;

case 3:

cout << "Bye" << endl;

exit\_key = 1;

break;

case 4:

Free(A, n, vis, set);

null(A, vis, set);

exit\_key = 5;

}

}

while ((exit\_key != 1) && (exit\_key != 3) && (exit\_key != 5))

{

cout << "Заполнить" << endl << "1.Вручную" << endl << "2.Авто" << endl << "3.Выход" << endl << "4.В начало" << endl;

cin >> k;

system("cls");

switch (k)

{

case 1:

Input(A, n);

exit\_key = 3;

break;

case 2:

FillRandomNumbers(A, n);

exit\_key = 3;

break;

case 3:

cout << "Bye" << endl;

exit\_key = 1;

break;

case 4:

Free(A, n, vis, set);

null(A, vis, set);

exit\_key = 5;

}

}

while ((exit\_key != 1) && (exit\_key != 4) && (exit\_key != 5))

{

cout << "Работа с матрицей" << endl << "1.Отрисовать" << endl << "2.Поиск н.п." << endl << "3.Выход" << endl << "4.В начало" << endl;

cin >> k;

system("cls");

switch (k)

{

case 1:

cout << "Матрица:" << endl;

Print(A, n);

break;

case 2:

system("cls");

cout << "Матрица:" << endl;

Print(A, n);

search(i, most\_set, max, A, sled, max\_shag, vis, set, num\_reb, max\_mas, n, x, max\_size);

exit\_key = 4;

break;

case 3:

system("cls");

cout << "Bye" << endl;

exit\_key = 1;

break;

case 4:

Free(A, n, vis, set);

null(A, vis, set);

exit\_key = 5;

break;

}

}

while ((exit\_key != 1) && (exit\_key != 5))

{

cout << "1.Сохранить" << endl << "2.Выход" << endl << "3.В начало" << endl;

cin >> k;

system("cls");

switch (k)

{

case 1:

cout << "Имя файла: ";

cin >> str;

cout << "Сохранено!" << endl;

save\_in\_file(str, A, n, max\_mas, max, num\_reb, max\_size);

break;

case 2:

system("cls");

cout << "Bye" << endl;

exit\_key = 1;

break;

case 3:

Free(A, n, vis, set);

null(A, vis, set);

exit\_key = 5;

}

}

break;

case 2:

cout << "Bye" << endl;

exit\_key = 1;

}

}

}

Файл **main.cpp**

#include "head.h"

int main()

{

setlocale(LC\_ALL, "Russian");

menu();

return 0;

}