Министерство науки и высшего образования Российской Федерации

Пензенский государственный университет

Кафедра «Вычислительная техника»

**ОТЧЕТ**

по лабораторной работе №8

по курсу «Логика и основы алгоритмизации в инженерных задачах»

на тему «Обход графа в ширину»

**Выполнили студенты группы 21вв1:**

Нечаев А.Д.

Киреев Д.А.

**Приняли**

Юрова О.В.

Акифьев И.В.

Пенза 2022

**Методические указания.**

**Задание**

1. Сгенерируйте (используя генератор случайных чисел) матрицу смежности для неориентированного графа *G*. Выведите матрицу на экран.
2. Для сгенерированного графа осуществите процедуру обхода в ширину, реализованную в соответствии с приведенным выше описанием. При реализации алгоритма в качестве очереди используйте класс **queue** из стандартной библиотеки С++.

**Описание метода решения задачи:**

**Вход**: G – матрица смежности графа.

**Выход**: номера вершин в порядке их прохождения на экране.

**Алгоритм ПОШ**

1.1. для всех i положим NUM[i] =  False пометим как "не посещенную";

1.2. **ПОКА** существует "новая" вершина v

1.3. **ВЫПОЛНЯТЬ** BFS (v).

**Алгоритм** BFS(v):

2.1. Создать пустую очередь Q = {};

2.2. Поместить v в очередь Q.push(v);

2.3. пометить v как "посещенную" NUM[v] = True;

2.4. **ПОКА**  Q != ∅ очередь не пуста **ВЫПОЛНЯТЬ**

2.5. v = Q.front() установить текущую вершину;

2.6. Удалить первый элемент из очереди Q.pop();

2.7. вывести на экран v;

2.8. **ДЛЯ** i = 1 **ДО** size\_G **ВЫПОЛНЯТЬ**

2.9.   **ЕСЛИ**  G(v,i) = = 1**И** NUM[i] = = False

2.10. **ТО**

2.11.       Поместить i в очередь Q.push(i);

2.12.       пометить v как "посещенную" NUM[v] = True;

**Листинг:**

//8

#include <queue>

#include <iostream>

#include <locale>

using namespace std;

void BFS(int \*(\*mass),bool \*be, int pos,int n)

{

queue <int> q;

q.push(pos);

be[pos] = true;

while(!q.empty())

{

pos=q.front();

q.pop();

printf("%d ",pos+1);

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

{

if(mass[pos][i]==1 && be[i]==false)

{

q.push(i);

be[i] = true;

}

}

}

}

int main()

{

setlocale(LC\_ALL, "Rus");

int\*\* a, n, k = 0;

int i = 0, j = 0,w,t=0;

printf("Введите кол-во строк и столбцов\n");

scanf\_s("%d", &n);

a = (int\*\*)malloc(n \* sizeof(int\*));

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

{

a[i] = (int\*)malloc(n \* sizeof(int));

}

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

{

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

{

w = rand() % 2;

if (w == 1)

{

a[i][j] = 1;// rand() % 10;

a[j][i] = a[i][j];

}

else {

a[i][j] = 0;

a[j][i] = a[i][j];

}

if (i == j)

{

a[i][j] = 0;

}

}

}

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

{

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

{

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

if (a[i][j] != 0)

{

t++;

}

k++;

}

printf("\n");

}

printf("%d\n", k);

int pos=0;

bool\* be = (bool\*)malloc(n \* sizeof(bool));

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

{

be[i] = false;

}

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

{

if(be[i] == false)

{

BFS(a,be,i,n);

}

}

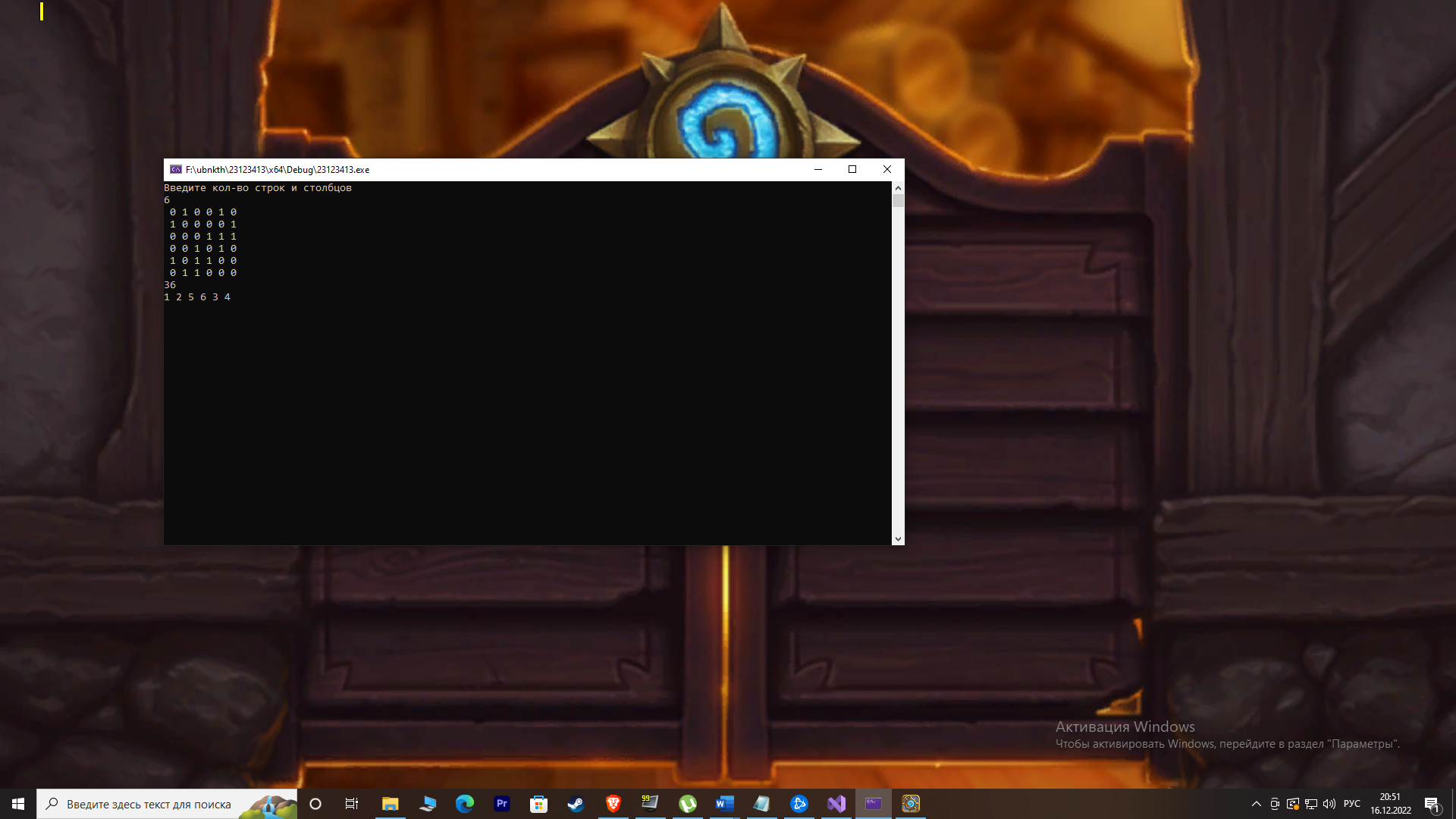
free(a);

getchar();

getchar();

}

**Результаты работы программы:**



**Вывод:**

в ходе выполнения лабораторной работы мы реализовали обход графа в ширину. Результаты работы программ совпали с результатами трассировки, следовательно программа работает без ошибок.