Министерство науки и высшего образования

Пензенский государственный университет

Кафедра «Вычислительная техника»

Отчет

По лабораторной работе №7

по курсу «Логика и основы алгоритмизации в инженерных задачах»

на тему «Обход графа в глубину»

Выполнили студенты группы 21ВВ2:

Алёшина А.В.

Мубаряков А.А.

Асянов И. К.

Приняли:

Юрова О.В.

Митрохин М.А.

Пенза 2022

Ход работы:

Сгенерировали с помощью генератора случайных чисел матрицу  
смежности для неориентированного графа. Матрица выводится на экран. Осуществили процедуру обхода графа в глубину.

Листинг:

#include <iostream>

#include <windows.h>

using namespace std;

void Generator(bool\*\* \_matrix, int \_SIZE)

{

for (int row = 0; row < \_SIZE; ++row)

{

\_matrix[row] = new bool[\_SIZE];

}

for (int row = 0; row < \_SIZE; ++row)

{

for (int col = 0; col < \_SIZE; ++col)

{

\_matrix[row][col] = 0;

}

}

bool buffer = 0;

for (int row = 0; row < \_SIZE; ++row)

{

for (int col = 0; col < \_SIZE; ++col)

{

buffer = rand() % 2;

\_matrix[row][col] = buffer;

\_matrix[col][row] = buffer;

}

}

for (int counter = 0; counter < \_SIZE; ++counter)

{

\_matrix[counter][counter] = 0;

}

}

void Output(bool\*\* \_matrix, int \_SIZE)

{

for (int row = 0; row < \_SIZE; ++row)

{

for (int col = 0; col < \_SIZE; ++col)

{

std::cout << \_matrix[row][col] << ' ';

}

std::cout << std::endl;

}

}

void ZeroingOut(bool\* \_matrix, int \_SIZE)

{

for (int counter = 0; counter < \_SIZE; ++counter)

{

\_matrix[counter] = 0;

}

}

void Bypass(int \_num, bool\* \_visits, bool\*\* \_matrix, int \_SIZE)

{

\_visits[\_num] = 1;

printf("%d, ", \_num);

for (int i = 0; i < \_SIZE; ++i)

{

if (\_matrix[\_num][i] == 1 && \_visits[i] == 0)

{

Bypass(i, \_visits, \_matrix, \_SIZE);

}

}

}

int main()

{

SetConsoleCP(1251);

SetConsoleOutputCP(1251);

cout << "Введите размер"<<endl;

int SIZE;

cin >> SIZE;

bool\*\* adjacencyMatrix = new bool\* [SIZE];

bool\* visits = new bool [SIZE];

ZeroingOut(visits, SIZE);

Generator(adjacencyMatrix, SIZE);

Output(adjacencyMatrix, SIZE);

std::cout << std::endl << "Пройденные вершины: ";

Bypass(0, visits, adjacencyMatrix, SIZE);

delete [] adjacencyMatrix;

delete [] visits;

return 0;

}

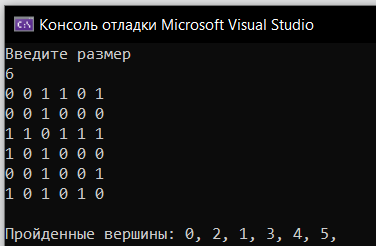


Рис.1 - Результат работы программы

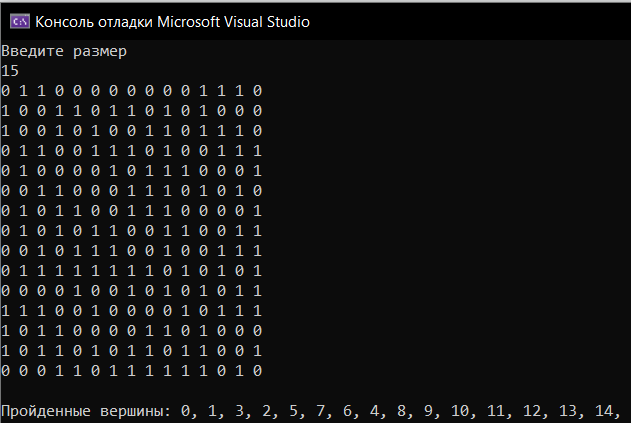


Рис.2 - Результат работы программы

Вывод:

В ходе работы была написана программа, которая генерирует неориентированный граф и осуществляет его обход в глубину. Задачей данного обхода является прохождение всех вершин графа. Применить данный обход можно для проверки связности, поиска пути и т.д.