Учреждение образования

«БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНОЛОГИЧЕСКИЙ УНИВЕРСИТЕТ»

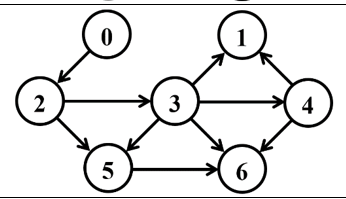
Дисциплина «Математическое программирование»

Отчёт по лабораторной работе №6

Студент: Буйко П.И.

ФИТ 2 курс 1 группа

Минск 2023



Исходный граф: (0,2), (2,3), (2,5), (3,1), (3,4), (3, 5), (3,6), (4,1), (4,6), (5,6).

Задание 1

Матрица смежности

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Вершины | 0 | 1 | 2 | 3 | 4 | 5 | 6 |
| 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 |
| 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 2 | 0 | 0 | 0 | 1 | 0 | 1 | 0 |
| 3 | 0 | 0 | 0 | 0 | 1 | 1 | 1 |
| 4 | 1 | 0 | 0 | 0 | 0 | 0 | 1 |
| 5 | 0 | 0 | 0 | 0 | 0 | 0 | 1 |
| 6 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Матрица инцидентности

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Связи Вершины | g(0,2) | g(2,3) | g(2,5) | g(3,1) | g(3,4) | g(3,5) | g(3,6) | g(4,1) | g(4,6) | g(5,6) |
| 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 1 | 0 | 0 | 0 | -1 | 0 | 0 | 0 | -1 | 0 | 0 |
| 2 | -1 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 3 | 0 | -1 | 0 | 1 | 1 | 1 | 1 | 0 | 0 | 0 |
| 4 | 0 | 0 | 0 | 0 | -1 | 0 | 0 | 1 | 1 | 0 |
| 5 | 0 | 0 | -1 | 0 | 0 | -1 | 0 | 0 | 0 | 1 |
| 6 | 0 | 0 | 0 | 0 | 0 | 0 | -1 | 0 | -1 | -1 |

Список смежных вершин

0 2

1

2 3,5

3 1,4,5,6

4 6

5 6

6

Задание 2.

Алгоритм поиска в ширину (BFS).

Алгоритм графа в ширину (BFS) используется для обхода графа и нахождения кратчайшего пути от начальной вершины до всех остальных вершин графа. Он работает следующим образом:

Создать очередь и добавить в нее начальную вершину.

Создать массив посещенных вершин и пометить начальную вершину как посещенную.

Пока очередь не пуста, извлечь из очереди первую вершину и для каждой смежной с ней вершины, которая еще не была посещена, пометить ее как посещенную и добавить в очередь.

Повторять шаг 3 до тех пор, пока очередь не станет пустой.

В результате работы алгоритма, для каждой вершины графа будет найден кратчайший путь от начальной вершины до нее.

Поиск в ширину: 0 2 3 5 4 6

0: q = (0),

1: 0, q = (2),

2: 2, q = (3,5),

3: 3, q = (1,4,5,6),

4: 5, q = (6),

5: 1, q = (),

6: 4, q = (1,6),

7: 6, q = ().

Очередь графа пуста, поэтому обход графа закончен.

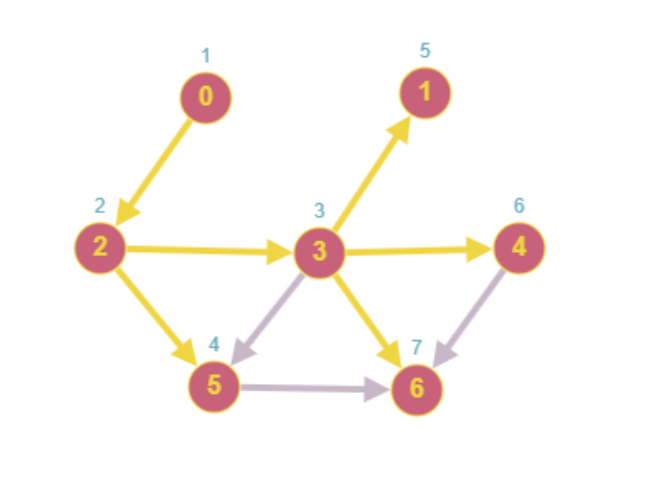


Рисунок 2 – Поиск в ширину

Алгоритм обхода в глубину (DFS)

Алгоритм обхода графа в глубину (Depth-First Search, DFS) — это алгоритм, который используется для обхода всех вершин графа. Он начинает с одной вершины и идет вглубь графа, пока не достигнет конца пути. Затем он возвращается на предыдущий уровень и продолжает обход до тех пор, пока не пройдет все вершины.

Поиск в глубину: 0 2 3 1 4 6 5

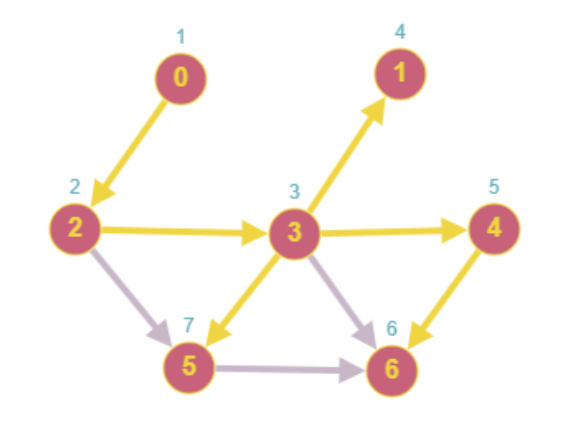


Рисунок 3 – Поиск в глубину

Так как все вершины помечены как посещённые, то мы заканчиваем обход графа в глубину.

Алгоритм топологической сортировки.

Топологическая сортировка (Topological sort) — один из основных алгоритмов на графах, который применяется для решения множества более сложных задач.

Задача топологической сортировки графа состоит в следующем: указать такой линейный порядок на его вершинах, чтобы любое ребро вело от вершины с меньшим номером к вершине с большим номером. Очевидно, что если в графе есть циклы, то такого порядка не существует.

Топологическая сортировка:

6,4, 5, 3, 2, 0, 1

Задание 3

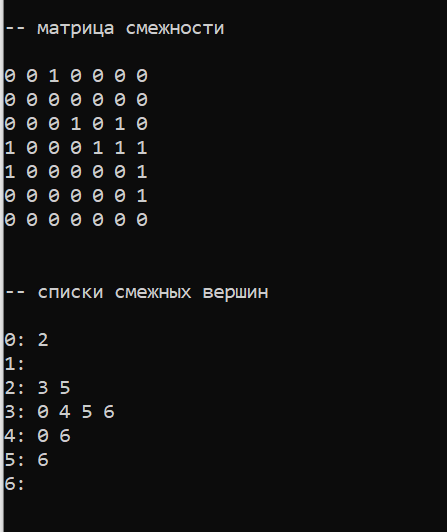


Рисунок 4 – Демонстрация программы с функцией BFS

Задание 4

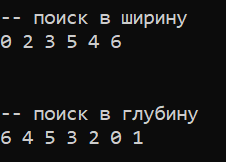


Рисунок 5 – Демонстрация работы функции DFS

Задание 5



Рисунок 6 – Демонстрация работы функции DFS

============================================

Задание 6

На вход алгоритма подаётся связный неориентированный граф. Для каждого ребра задаётся его стоимость.

Сначала берётся произвольная вершина и находится ребро, инцидентное данной вершине и обладающее наименьшей стоимостью. Найденное ребро и соединяемые им две вершины образуют дерево. Затем, рассматриваются рёбра графа, один конец которых — уже принадлежащая дереву вершина, а другой — нет; из этих рёбер выбирается ребро наименьшей стоимости. Выбираемое на каждом шаге ребро присоединяется к дереву. Рост дерева происходит до тех пор, пока не будут исчерпаны все вершины исходного графа.

Результатом работы алгоритма является остовное дерево минимальной стоимости.

Начальная вершина 0:

1. {0} => ({0,2} = 2). Идем {0,2}

2. {1} => ({2,3} = 7)

3. {3} => ({3,4} = 1), ({3,5} = 6), ({3,1} = 4). Идем {3,4}

4. {4} => ({4,6} = 10)

5. {6} => ({6,5} = 2)

1 + 7 + 1 +10 + 2 = 21

Вес минимального остовного дерева равен 21.

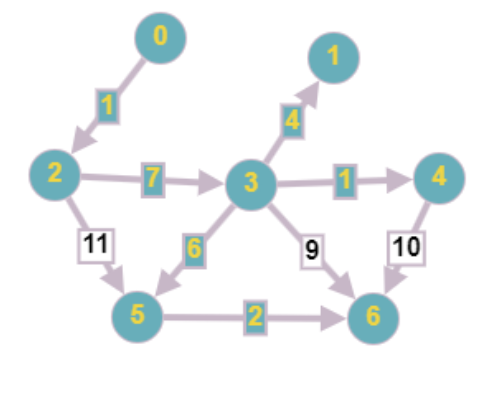


Рисунок 6 – Завершенный алгоритм Прима

Задание 7

В начале текущее множество рёбер устанавливается пустым. Затем, пока это возможно, проводится следующая операция: из всех рёбер, добавление которых к уже имеющемуся множеству не вызовет появление в нём цикла, выбирается ребро минимального веса и добавляется к уже имеющемуся множеству. Когда таких рёбер больше нет, алгоритм завершён. Подграф данного графа, содержащий все его вершины и найденное множество рёбер, является его остовным деревом минимального веса.

Начальное ребро {0,2} = 1

1. {0,2} => ({3,4} = 1),

2. {3,4} => ({5,6} = 2),

3. {5,6} => ({3,1} = 4),

4. {3,1} => ({3,2} = 7),

5.

1+1+2+4+6+7=21

Вес минимального остовного дерева равен 21.

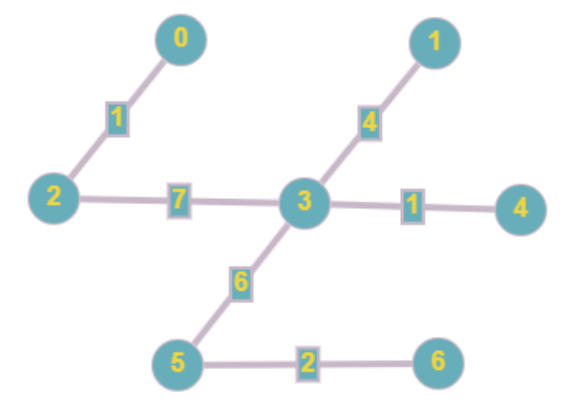


Рисунок 7 – Завершенный алгоритм Краскала