**Лабораторная работа 5. АЛГОРИТМЫ НА ГРАФАХ**

**(алгоритмы поиска в ширину и глубину, топологическая сортировка)**

**ЦЕЛЬ РАБОТЫ:** Освоить сущность и программную реализацию: а) способов представления графов; б) алгоритмов поиска в ширину и глубину; в) алгоритма топологической сортировки графов.

**Задание для выполнения:**

|  |  |
| --- | --- |
| Вариант | Граф G |
| 4 |  |

**Задание 1.**  Ориентированный граф **G** взять в соответствии с вариантом. Осуществить алгоритмы поиска в ширину и глубину, а также алгоритма топологической сортировки аналогично примерам, рассмотренным на лекциях. Оформить отчет, включив в него **каждый** шаг выполнения алгоритмов.

**Поиск в ширину:**

массивы:

Q- для промежуточного хранения вершин(очередь)

С- массив окраски вершин

D- массив расстояний

Р- массив предшествующих вершин

Шаг 1

Q-0

C-GWWWW

D-0 | | | |

P-N N N N N

Шаг 2

Q-1

C-BGWWW

D-0 1 | | |

P-N0NNN

Шаг 3

Q-2 ,3

C-BBGWW

D-0 1 2 | |

P-N01NN

Шаг 4

Q-3

C-BBBGW

D-0 1 2 2 |

P-N0 1 1 N

Шаг 5

Q-4

C-BBBBG

D-0 1 2 2 3

P-N 0 1 1 3

Шаг 6

Q- -

C-BBBBB

D-0 1 2 2 3

P-N 0 1 1 3

**Алгоритм BFS** сводится к следующей последовательности шагов.

1. Инициализировать массивы **С**, **D**, **P**. Стартовую вершину **s** поместить в очередь **Q**. и окрасить в серый цвет: **C[s] = G**. Для стартовой вершины установить расстояние, равное нулю: **D[s] = 0**.
2. Если очередь **Q** пуста, то работа алгоритма завершена, в противном случае перейти к следующему шагу.
3. Выбрать из очереди **Q** вершину **k** и окрасить ее в черный цвет: **С[k] = B**.
4. Построить множества **J** вершин белого цвета смежных вершине **k**. Если таких вершин нет, то перейти к шагу 2, иначе – к следующему шагу.
5. Каждую вершину **j** из множества **J** поместить в очередь **Q**. Обычно (но не обязательно) в очередь вершины помещаются в порядке возрастания номеров.
6. Каждую вершину **j** из множества **J** окрасить в серый цвет: **С[j] = G**.
7. Для каждой вершины **j** из множества **J** вычислитьрасстояние: **D[j] = D[k] + 1**.
8. Для каждой вершины **j** из множества **J** указать предшествующую вершину: **P[j] = k**.
9. Перейти к шагу 3.

**Поиска в глубину:**

массивы:

T-шаг

С- массив окраски вершин

D- массив расстояний

Р- массив предшествующих вершин

F- массив шаг на котором вершина окрашивается в черный цвет

Шаг 1

C-G W W W W

D-1 N N N N

P- N N N N N

F- 0 0 0 0 0

Шаг 2

C-G G W W W

D-1 2 N N N

P- N 0 N N N

F- 0 0 0 0 0

Шаг 3

C-G G G W W

D-1 2 3 N N

P- N 0 1 N N

F- 0 0 0 0 0

Шаг 4

C-G G G G W

D-1 2 3 4 N

P- N 0 1 1 N

F- 0 0 0 0 0

Шаг 5

C-G G G G G

D-1 2 3 4 5

P- N 0 1 1 3

F- 0 0 0 0 0

Шаг 6

C-G G G G B

D-1 2 3 4 5

P- N 0 1 1 3

F- 0 0 0 0 6

Шаг 7

C-G G G B B

D-1 2 3 4 5

P- N 0 1 1 3

F- 0 0 0 7 6

Шаг 8

C-G G B B B

D-1 2 3 4 5

P- N 0 1 1 3

F- 0 0 8 7 6

Шаг 9

C-G B B B B

D-1 2 3 4 5

P- N 0 1 1 3

F- 0 9 8 7 6

Шаг 10

C-B B B B B

D-1 2 3 4 5

P- N 0 1 1 3

F-10 9 8 7 6

В основе алгоритма DFS лежит рекурсивная процедура **Visit**, имеющая один входной параметр **k** – вершину графа.

Опишем пошагово процедуру **Visit**.

1. Принять параметр **k** – вершину графа.
2. Вершину **k**  окрасить в серый цвет: **C[k] = G**.
3. Увеличить номер шага: **t = t + 1**.
4. Подсчитать расстояние до вершины: **D[k] = t**. Расстояние до вершины в алгоритме DFS совпадает с номером шага, на котором эта вершина была обнаружена (окрашена в серый цвет).
5. Построить множества **J** вершин белого цвета, смежных вершине **k**. Если таких вершин нет, то перейти к шагу 8.
6. Для каждой вершины **j** из множества **J** указать предшествующую вершину: **P[j] = k**.
7. Для каждой вершины **j** из множества **J** выполнить процедуру **Visit**.
8. Вершину **k**  окрасить в черный цвет: **C[k] = B**.
9. Увеличить номер шага: **t = t + 1**.
10. Отметить время фиксации вершины: **F[k] = t**.

**Топологическая сортировка**

**Топологическая сортировка −** это процедура упорядочивания вершин ориентированного графа, не имеющего циклов (ациклического графа). В результате топологической сортировки для вершин графа определяется такой порядок, что если их расположить на рисунке в соответствии с этим порядком сверху вниз, то дуги будут направлены только от верхних вершин к нижним**.**

**1**

**0**

**3**

**4**

**2**