Учреждение образования

«БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНОЛОГИЧЕСКИЙ

УНИВЕРСИТЕТ»

**Лабораторная работа 5. АЛГОРИТМЫ НА ГРАФАХ**

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

Студент: Бобрович Г.С.

ФИТ 2 курс 7 группа

Преподаватель: Барковский Е.В.

Минск 2022

**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 |
| 3 | |  | | --- | |  | |  |  | |

**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 – G W W W W W  D – 0 1 I 1 I I I  P - N 0 N 0 N N N |  | |  | |  | |  | |
| **Шаг 2** | | |
| Q – 2 3  C – GR W G G W W  D – 0 1 I 1 I I I  P - N N 0 0 N N N | |  |
| **Шаг 3** | | |
| Q – 5  C – GR W GR G W G W  D – 0 1 I 1 I I I  P - N N 0 0 N 2 N | |  |
| **Шаг 4** | | |
| Q – 5 6  C – GR N GR GR N G N  D – 0 1 2 1 2 I 2  P - N N 0 0 N 2 5 | |  |
| **Шаг 5** | | |
| Q – 4 6  C – GR N GR GR G GR G  D – 0 1 2 1 2 3 2  P - N N 0 0 6 2 5 | |  |
| **Шаг 6** | | |
| Q – 1 4  C – GR N GR GR G GR GR  D – 0 1 2 1 2 3 2  P - N 4 0 0 6 2 5 | |  |
| **Шаг 7** | | |
| Q – 1  C – GR G GR GR GR GR GR  D – 0 1 2 1 2 3 2  P - N 4 0 0 6 2 5 | |  |
| **Шаг 8** | | |
| Q –  C – GR GR GR GR GR GR GR  D – 0 1 2 1 2 3 2  P - N 4 0 0 6 2 5 | |  |

В итоге получаем: (P = N 4 0 0 6 2 5)

**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.

**Поиск 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

|  |  |
| --- | --- |
| **T = 1**  C - G W W W W W W  D - 1 I I I I I I  P - N N N N N N N  F - 0 0 0 0 0 0 0 |  |
| **T = 2**  C - G W G W W W W  D - 1 2 I I I I I  P - N N 0 N N N N  F - 0 0 0 0 0 0 0 |  |
| **T = 3**  C - G W G W W G W  D - 1 2 I I I I I  P - N N 0 N N 2 N  F - 0 0 0 0 0 0 0 |  |
| **T = 4**  C - G W G W W G G  D - 1 2 I 4 I I I  P - N N 0 N N 2 5  F - 0 0 0 0 0 0 0 |  |
| **T = 5**  C - G W G W G G G  D - 1 2 5 4 I I I  P - N N 0 N 6 2 5  F – 0 0 0 0 0 0 0 |  |
| **T = 6**  C - G G G W G G G  D - 1 2 5 4 I 6 I  P - N 4 0 N 6 2 5  F – 0 0 0 0 0 0 0 | с |
| **T = 7**  C - G G G G G G G  D - 1 2 5 4 I 6 7  P - N 4 0 1 6 2 5  F – 0 0 0 0 0 0 0 |  |
| **T = 8**  C - G G G B G G G  D - 1 2 5 4 I 6 7  P - N 4 0 1 6 2 5  F – 0 0 0 8 0 0 0 |  |
| **T = 9**  C - G B G B G G G  D - 1 2 5 4 I 6 7  P - N 4 0 1 6 2 5  F – 0 9 0 8 0 0 0 |  |
| **T = 10**  C - G B G B B G G  D - 1 2 5 4 I 6 7  P - N 4 0 1 6 2 5  F – 0 9 0 8 10 0 0 |  |
| **T = 11**  C - G B G B B G B  D - 1 2 5 4 11 6 7  P - N 4 0 1 6 2 5  F – 0 9 0 8 10 0 11 |  |
| **T = 12**  C - G B G B B B B  D - 1 2 5 4 11 6 7  P - N 4 0 1 6 2 5  F – 0 9 0 8 10 12 11 |  |
| **T = 13**  C - G B B B B B B  D - 1 2 5 4 11 6 7  P - N 4 0 1 6 2 5  F – 0 9 13 8 10 12 11 |  |
| **T = 14**  C - B B B B B B B  D - 1 2 5 4 11 6 7  P - N 4 0 1 6 2 5  F – 14 9 13 8 10 12 11 |  |

По итогу получаем: (P = N 0 3 0 3 2 5)

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 | 2 |
| 3 | 4 |
| 5 | 6 |
| 7 | 8 |
| 9 | 10 |
| 11 | 12 |
| 13 | 14 |

По итогу получаем:

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

**Задание 2.** Осуществить программную реализацию алгоритмов на C++. Разработать структуры **AMatrix** и **АList** для представления ориентированного графа матричным и списковым способом. Разработать функции преобразования из одного способа представления в другой. Разработать функцию **BFS** обхода вершин графа, используя метод поиска в ширину. Продемонстрировать работу функции. Копии экрана вставить в отчет.

**Задание 3.**  Разработать функцию **DFS** обхода вершин графа, используя метод поиска глубину. Продемонстрировать работу функции. Копии экрана вставить в отчет.

**Задание 4.**  Доработайте функцию **DFS**,для выполнения топологической сортировки графа. Продемонстрировать работу функции. Копии экрана вставить в отчет.

