**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 |
| 12 |  |

**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.

**Алгоритм поиска в ширину (англ. breadth-first search, BFS)** позволяет найти кратчайшие пути из одной вершины невзвешенного графа до всех остальных вершин

**Поиск в ширину:**

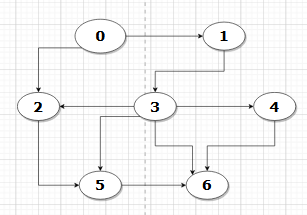
Q- для промежуточного хранения вершин(очередь)

массивы:

С- массив окраски вершин ( Б – не добавлена в очередь , С – добавлена в очередь , Ч – вышла из очереди )

D- массив расстояний

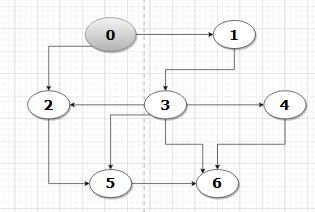
Р- массив предшествующих вершин



Шаг 1

Очередь : 0

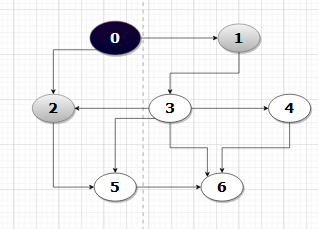
|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  | 0 | 1 | 2 | 3 | 4 | 5 | 6 |
| Окраска вершин | С | Б | Б | Б | Б | Б | Б |
| Расстояния | 0 | - | - | - | - | - | - |
| Массив предыдущих вершин | - | - | - | - | - | - | - |



Шаг 2

Очередь : 1 2

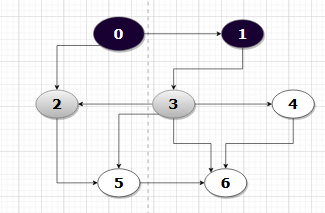
|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  | 0 | 1 | 2 | 3 | 4 | 5 | 6 |
| Окраска вершин | Ч | С | С | Б | Б | Б | Б |
| Расстояния | 0 | 1 | 1 | - | - | - | - |
| Массив предыдущих вершин | - | 0 | 0Ы | - | - | - | - |



Шаг 3

Очередь : 2 3

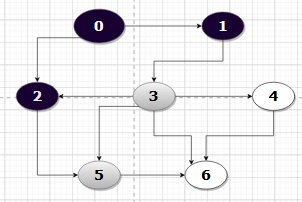
|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  | 0 | 1 | 2 | 3 | 4 | 5 | 6 |
| Окраска вершин | Ч | Ч | С | С | Б | Б | Б |
| Расстояния | 0 | 1 | 1 | 2 | - | - | - |
| Массив предыдущих вершин | - | 0 | 0 | 1 | - | - | - |



Шаг 4

Очередь : 3 5

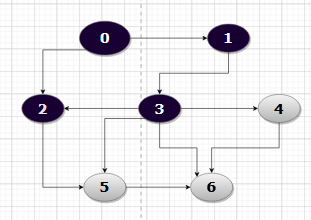
|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  | 0 | 1 | 2 | 3 | 4 | 5 | 6 |
| Окраска вершин | Ч | Ч | Ч | С | Б | С | Б |
| Расстояния | 0 | 1 | 1 | 2 | - | - | - |
| Массив предыдущих вершин | - | 0 | 0 | 1 | - | 2 | - |



Шаг 5

Очередь : 5 4 6

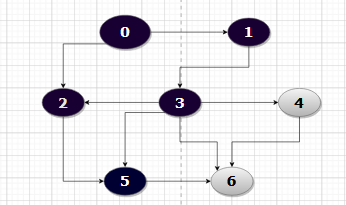
|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  | 0 | 1 | 2 | 3 | 4 | 5 | 6 |
| Окраска вершин | Ч | Ч | Ч | Ч | С | С | С |
| Расстояния | 0 | 1 | 1 | 2 | 1 | 1 | 1 |
| Массив предыдущих вершин | - | 0 | 0 | 1 | 3 | 2 | 3 |



Шаг 6

Очередь : 4 6

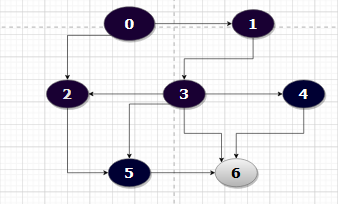
|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  | 0 | 1 | 2 | 3 | 4 | 5 | 6 |
| Окраска вершин | Ч | Ч | Ч | Ч | С | Ч | С |
| Расстояния | 0 | 1 | 1 | 2 | 1 | 1 | 1 |
| Массив предыдущих вершин | - | 0 | 0 | 1 | 3 | 2 | 3 |

ы

Шаг 7

Очередь : 6

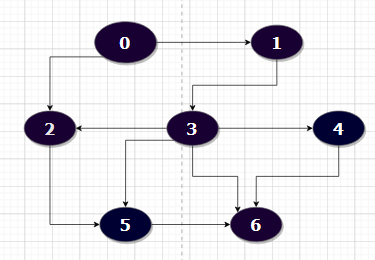
|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  | 0 | 1 | 2 | 3 | 4 | 5 | 6 |
| Окраска вершин | Ч | Ч | Ч | Ч | Ч | Ч | С |
| Расстояния | 0 | 1 | 1 | 2 | 1 | 1 | 1 |
| Массив предыдущих вершин | - | 0 | 0 | 1 | 3 | 2 | 3 |



Шаг 8

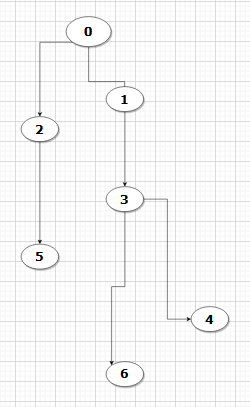
Очередь : 0

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  | 0 | 1 | 2 | 3 | 4 | 5 | 6 |
| Окраска вершин | Ч | Ч | Ч | Ч | Ч | Ч | Ч |
| Расстояния | 0 | 1 | 1 | 2 | 1 | 1 | 1 |
| Массив предыдущих вершин | - | 0 | 0 | 1 | 3 | 2 | 3 |



Очередь пуста – следует конец обхода.

Результат :



**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.

**Алгоритм поиска (или обхода) в глубину** (англ. depth-first search, DFS) позволяет построить обход графа, при котором посещаются все вершины, доступные из начальной вершины.

**Алгоритм поиска в глубину:**

T-шаг

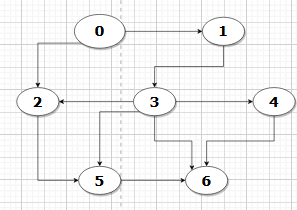
Стек посещённых вершин

Массив окраски вершин ( Б – не посещена , С – в стеке , Ч – вышла из стека)

Массив расстояний

Массив предшествующих вершин

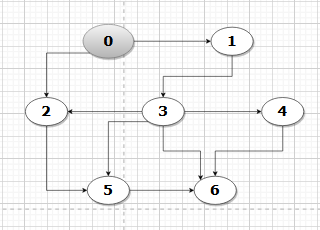
Массив шаг на котором вершина окрашивается в черный цвет



Шаг 1

Стек :

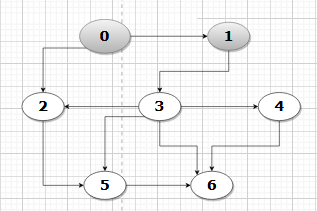
|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  | 0 | 1 | 2 | 3 | 4 | 5 | 6 |
| Окраска вершин | С | Б | Б | Б | Б | Б | Б |
| Расстояния | 0 |  |  |  |  |  |  |
| Номера предыдущих вершин |  |  |  |  |  |  |  |
| Номер шага , когда вершина выходит из алогоритма |  |  |  |  |  |  |  |



Шаг 2

Стек : 0

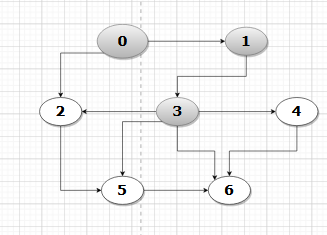
|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  | 0 | 1 | 2 | 3 | 4 | 5 | 6 |
| Окраска вершин | С | С | Б | Б | Б | Б | Б |
| Расстояния | 0 | 1 |  |  |  |  |  |
| Номера предыдущих вершин |  | 0 |  |  |  |  |  |
| Номер шага , когда вершина выходит из алогоритма |  |  |  |  |  |  |  |



Шаг 3

Стек : 0 1

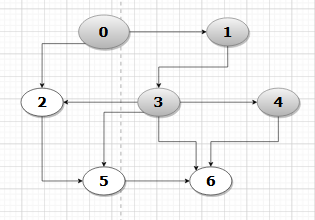
|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  | 0 | 1 | 2 | 3 | 4 | 5 | 6 |
| Окраска вершин | С | С | Б | С | Б | Б | Б |
| Расстояния | 0 | 1 |  | 2 |  |  |  |
| Номера предыдущих вершин |  | 0 |  | 1 |  |  |  |
| Номер шага , когда вершина выходит из алогоритма |  |  |  |  |  |  |  |



Шаг 4

Стек : 0 1 3

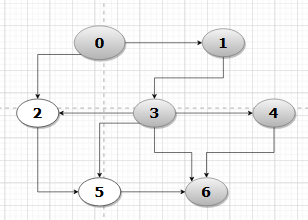
|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  | 0 | 1 | 2 | 3 | 4 | 5 | 6 |
| Окраска вершин | С | С | Б | С | С | Б | Б |
| Расстояния | 0 | 1 |  | 2 | 3 |  |  |
| Номера предыдущих вершин |  | 0 |  | 1 | 3 |  |  |
| Номер шага , когда вершина выходит из алогоритма |  |  |  |  |  |  |  |



Шаг 5

Стек : 0 1 3 4

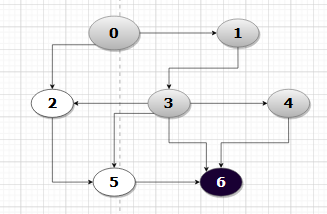
|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  | 0 | 1 | 2 | 3 | 4 | 5 | 6 |
| Окраска вершин | С | С | Б | С | С | Б | С |
| Расстояния | 0 | 1 |  | 2 | 3 |  | 4 |
| Номера предыдущих вершин |  | 0 |  | 1 | 3 |  | 4 |
| Номер шага , когда вершина выходит из алогоритма |  |  |  |  |  |  |  |



Шаг 6

Стек : 0 1 3

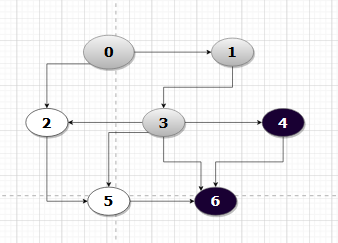
|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  | 0 | 1 | 2 | 3 | 4 | 5 | 6 |
| Окраска вершин | С | С | Б | С | С | Б | Ч |
| Расстояния | 0 | 1 |  | 2 | 3 |  | 4 |
| Номера предыдущих вершин |  | 0 |  | 1 | 3 |  | 4 |
| Номер шага , когда вершина выходит из алогоритма |  |  |  |  |  |  | 6 |



Шаг 7

Стек : 0 1

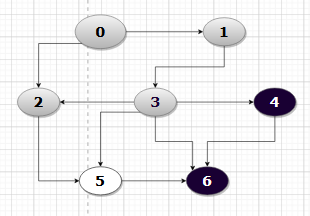
|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  | 0 | 1 | 2 | 3 | 4 | 5 | 6 |
| Окраска вершин | С | С | Б | С | Ч | Б | Ч |
| Расстояния | 0 | 1 |  | 2 | 3 |  | 4 |
| Номера предыдущих вершин |  | 0 |  | 1 | 3 |  | 4 |
| Номер шага , когда вершина выходит из алогоритма |  |  |  |  | 7 |  | 6 |

Ы

Шаг 8

Стек : 0 1 3

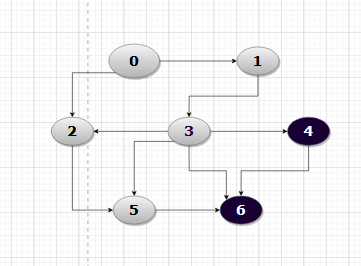
|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  | 0 | 1 | 2 | 3 | 4 | 5 | 6 |
| Окраска вершин | С | С | С | С | Ч | Б | Ч |
| Расстояния | 0 | 1 | 3 | 2 | 3 |  | 4 |
| Номера предыдущих вершин |  | 0 | 2 | 1 | 3 |  | 4 |
| Номер шага , когда вершина выходит из алгоритма |  |  |  |  | 7 |  | 6 |



Шаг 9

Стек : 0 1 3 2

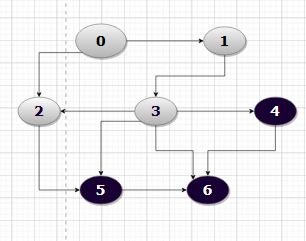
|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  | 0 | 1 | 2 | 3 | 4 | 5 | 6 |
| Окраска вершин | С | С | С | С | Ч | С | Ч |
| Расстояния | 0 | 1 | 3 | 2 | 3 | 4 | 4 |
| Номера предыдущих вершин |  | 0 | 2 | 1 | 3 | 2 | 4 |
| Номер шага , когда вершина выходит из алгоритма |  |  |  |  | 7 |  | 6 |



Шаг 10

Стек : 0 1 3

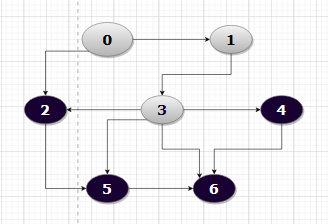
|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  | 0 | 1 | 2 | 3 | 4 | 5 | 6 |
| Окраска вершин | С | С | С | С | Ч | Ч | Ч |
| Расстояния | 0 | 1 | 3 | 2 | 3 | 4 | 4 |
| Номера предыдущих вершин |  | 0 | 2 | 1 | 3 | 2 | 4 |
| Номер шага , когда вершина выходит из алгоритма |  |  |  |  | 7 | 10 | 6 |



Шаг 11

Стек : 0 1

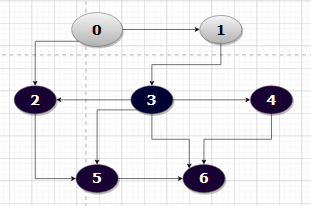
|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  | 0 | 1 | 2 | 3 | 4 | 5 | 6 |
| Окраска вершин | С | С | Ч | С | Ч | Ч | Ч |
| Расстояния | 0 | 1 | 3 | 2 | 3 | 4 | 4 |
| Номера предыдущих вершин |  | 0 | 2 | 1 | 3 | 2 | 4 |
| Номер шага , когда вершина выходит из алгоритма |  |  | 11 |  | 7 | 10 | 6 |



Шаг 12

Стек : 0

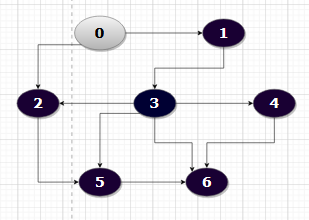
|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  | 0 | 1 | 2 | 3 | 4 | 5 | 6 |
| Окраска вершин | С | С | Ч | Ч | Ч | Ч | Ч |
| Расстояния | 0 | 1 | 3 | 2 | 3 | 4 | 4 |
| Номера предыдущих вершин |  | 0 | 2 | 1 | 3 | 2 | 4 |
| Номер шага , когда вершина выходит из алгоритма |  |  | 11 | 12 | 7 | 10 | 6 |



Шаг 13

Стек :

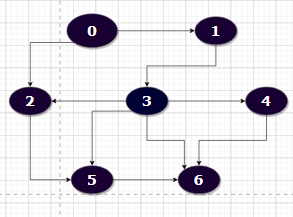
|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  | 0 | 1 | 2 | 3 | 4 | 5 | 6 |
| Окраска вершин | С | Ч | Ч | Ч | Ч | Ч | Ч |
| Расстояния | 0 | 1 | 3 | 2 | 3 | 4 | 4 |
| Номера предыдущих вершин |  | 0 | 2 | 1 | 3 | 2 | 4 |
| Номер шага , когда вершина выходит из алгоритма |  | 13 | 11 | 12 | 7 | 10 | 6 |



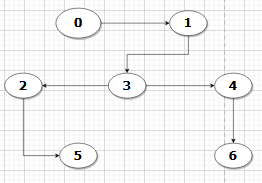
Шаг 14

Стек :

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  | 0 | 1 | 2 | 3 | 4 | 5 | 6 |
| Окраска вершин | Ч | Ч | Ч | Ч | Ч | Ч | Ч |
| Расстояния | 0 | 1 | 3 | 2 | 3 | 4 | 4 |
| Номера предыдущих вершин |  | 0 | 2 | 1 | 3 | 2 | 4 |
| Номер шага , когда вершина выходит из алгоритма | 14 | 13 | 11 | 12 | 7 | 10 | 6 |



Результат



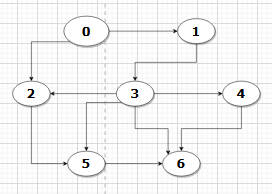
В основе алгоритма DFS лежит рекурсивная процедура **Visit**, имеющая один входной параметр **k** – вершину графа.

Опишем пошагово процедуру **Visit**.

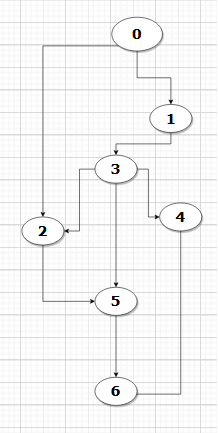
1. Принять параметр **k** – вершину графа.
2. Вершину **k**  окрасить в серый цвет: **C[k] = G**.
3. Увеличить номер шага: **t = t + 1**.
4. Подсчитать расстояние до вершины: **D[k] = t**. Расстояние до вершины в алгоритме DFS совпадает с номером шага, на котором эта вершина была обнаружена (окрашена 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**

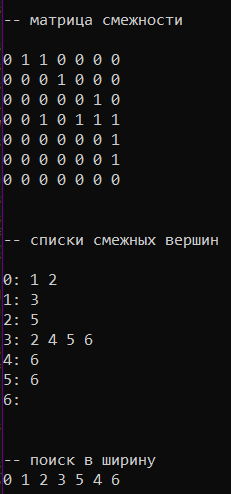
**Топологическая сортировка −** это процедура упорядочивания вершин бесконтурного ориентированного графа, не имеющего циклов (ациклического графа). В результате топологической сортировки для вершин графа определяется такой порядок, что если их расположить на рисунке в соответствии с этим порядком сверху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

****

**Результат**

****

**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.

