Учреждение образования

«Белорусский государственный технологический университет»

Факультет информационных технологий

Кафедра информационных систем и технологий

Электронный конспект по дисциплине

**Лабораторная работа № 5**

**АЛГОРИТМЫ НА ГРАФАХ**

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

Выполнила:

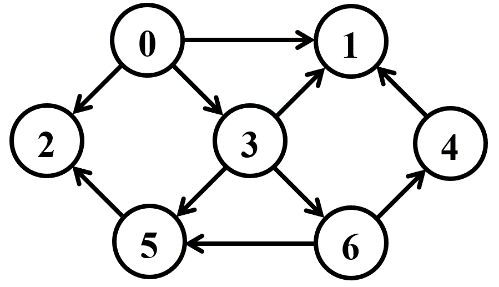
Студентка 2 курса 3 группы

Кохнюк Александра Сергеевна

**Вариант 5**

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

**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.



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

**Алгоритм поиска в ширину**

**Поиск в ширину (BFS)** – один из алгоритмов обхода графа. Метод лежит в основе некоторых других алгоритмов близкой тематики. Поиск в ширину подразумевает поуровневое исследование графа: вначале посещается корень – произвольно выбранный узел, затем – все потомки данного узла, после этого посещаются потомки потомков и т.д. Вершины просматриваются в порядке возрастания их расстояния от корня.

**Q** – очередь вершин,

**C** – массив окраски вершин,

**D** – массив расстояний,

**P** – массив предшествующих вершин.

Шаг 1.

В качестве стартовой вершины выбираем вершину с номером 0. Сначала она помечается серым как обнаруженный, а затем черным, поскольку обнаружены смежные с ней узлы.

0 1

2 3 4

5 6

|  |  |
| --- | --- |
| **Q** | 0 |
| **C** | G | W | W | W | W | W | W |
| **D** | 0 | I | I | I | I | I | I |
| **P** | N | N | N | N | N | N | N |

G – gray

W – вершина не посещалась

B – black

Шаг 2. Вершина 0 имеет смежные вершины 1, 2, 3, окрашиваем их серым. Вершину 0 закрашиваем в черный как пройденную.

0 1

2 3 4

5 6

|  |  |  |  |
| --- | --- | --- | --- |
| **Q** | 1 | 2 | 3 |
| **C** | В | G | G | G | W | W | W |
| **D** | 0 | 1 | 1 | 1 | I | I | I |
| **P** | N | 0 | 0 | 0 | N | N | N |

Шаг 3. Для следующего пути выбираем вершину с наименьшим весом – 1, а 2ая и 3 в очередь. Вершина 1 не имеет никаких смежных вершин, окрашиваем ее в черный цвет.

0 1

2 3 4

5 6

|  |  |  |
| --- | --- | --- |
| **Q** | 2 | 3 |
| **C** | В | B | G | G | W | W | W |
| **D** | 0 | 1 | 1 | 1 | I | I | I |
| **P** | N | 0 | 0 | 0 | N | N | N |

Шаг 4. Переходим в вершину 2. Она также не имеет смежных вершин, закрашиваем черным как пройденную.

0 1

2 3 4

5 6

|  |  |
| --- | --- |
| **Q** | 3 |
| **C** | В | B | B | G | W | W | W |
| **D** | 0 | 1 | 1 | 1 | I | I | I |
| **P** | N | 0 | 0 | 0 | N | N | N |

Шаг 5. Перейдем в следующую вершину с наименьшим весом – 3. Она имеет смежные вершины 1, 5 и 6. Но вершина 1 уже посещалась. Вершину 3 закрашиваем как пройденную.

0 1

2 3 4

5 6

|  |  |  |
| --- | --- | --- |
| **Q** | 5 | 6 |
| **C** | В | B | B | B | W | G | G |
| **D** | 0 | 1 | 1 | 1 | I | 2 | 2 |
| **P** | N | 0 | 0 | 0 | N | 3 | 3 |

Шаг 6. Так как мы уже были в вершине 1, то переходим в вершину 5. Она имеет единственную смежную вершину 2, но мы уже были в ней. Вершину 5 закрашиваем как пройденную.

0 1

2 3 4

5 6

|  |
| --- |
| **Q** |
| **C** | В | B | B | B | W | В | G |
| **D** | 0 | 1 | 1 | 1 | I | 2 | 2 |
| **P** | N | 0 | 0 | 0 | N | 3 | 3 |

Шаг 7. Переходим обратно в вершину 3, а из нее уже в 6ую вершину. Смежная вершина с ней только вершина 4. Закрашиваем вершину 6 как пройденную.

0 1

2 3 4

5 6

|  |  |
| --- | --- |
| **Q** | 4 |
| **C** | В | B | B | B | G | В | В |
| **D** | 0 | 1 | 1 | 1 | 3 | 2 | 2 |
| **P** | N | 0 | 0 | 0 | 6 | 3 | 3 |

Шаг 8. Переходим в вершину 4. Так как смежная вершина только одна – 1 и она уже пройдена, алгоритм окончен.

0 1

2 3 4

5 6

|  |  |
| --- | --- |
| **Q** | 1 |
| **C** | В | B | B | B | В | В | В |
| **D** | 0 | 1 | 1 | 1 | 3 | 2 | 2 |
| **P** | N | 0 | 0 | 0 | 6 | 3 | 3 |

**В результате получаем дерево:**

0 1

2 3 4

5 6

**Алгоритм поиска в глубину**

**Алгоритм поиска (или обхода) в глубину** (DFS) позволяет построить обход ориентированного или неориентированного графа, при котором посещаются все вершины, доступные из начальной вершины.

Отличие поиска в глубину от поиска в ширину заключается в том, что поиск в глубину строит дерево путей из начальной вершины во все доступные из нее.

**t** – номер шага алгоритма

**F** – время фиксации (шага окраски в черный цвет)

**C** – массив окраски вершин,

**D** – время, шаг окраски в серый цвет

**P** – массив предшествующих вершин.

Шаг 1. В качестве стартовой вершины выбираем вершину с номером 0. Далее будем осуществлять проход по смежным вершинам, пока не сможет достичь того, чтобы не было возможности осуществить дальнейший проход.

0 1

2 3 4

5 6

|  |  |
| --- | --- |
| **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 |

Шаг 2. Вершина 0 имеет 3 смежные вершины – 1, 2, 3. Переходим в 1ую, так как она наименьшего веса и окрашиваем ее в серый.

0 1

2 3 4

5 6

|  |  |
| --- | --- |
| **t** | 2 |
| **C** | G | G | W | W | W | W | W |
| **D** | 1 | 2 | I | I | I | I | I |
| **P** | N | 0 | N | N | N | N | N |
| **F** | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Шаг 3. Вершина 1 не имеет смежных вершин. Поэтому закрашиваем ее в черный.

0 1

2 3 4

5 6

|  |  |
| --- | --- |
| **t** | 3 |
| **C** | G | B | W | W | W | W | W |
| **D** | 1 | 2 | I | I | I | I | I |
| **P** | N | 0 | N | N | N | N | N |
| **F** | 0 | 3 | 0 | 0 | 0 | 0 | 0 |

Шаг 4. Возвращаемся в вершину 0. Следующая вершина по весу – 2.

0 1

2 3 4

5 6

|  |  |
| --- | --- |
| **t** | 4 |
| **C** | G | B | G | W | W | W | W |
| **D** | 1 | 2 | 3 | I | I | I | I |
| **P** | N | 0 | 0 | N | N | N | N |
| **F** | 0 | 3 | 0 | 0 | 0 | 0 | 0 |

Шаг 5. Вершина 2 также не имеет смежных вершин, поэтому окрашиваем ее в черный цвет.

0 1

2 3 4

5 6

|  |  |
| --- | --- |
| **t** | 5 |
| **C** | G | B | B | W | W | W | W |
| **D** | 1 | 2 | 3 | I | I | I | I |
| **P** | N | 0 | 0 | N | N | N | N |
| **F** | 0 | 3 | 5 | 0 | 0 | 0 | 0 |

Шаг 6. Возвращаемся в вершину 0. Следующая вершина по весу – 3.

0 1

2 3 4

5 6

|  |  |
| --- | --- |
| **t** | 6 |
| **C** | G | B | B | G | W | W | W |
| **D** | 1 | 2 | 3 | 4 | I | I | I |
| **P** | N | 0 | 0 | 0 | N | N | N |
| **F** | 0 | 3 | 5 | 0 | 0 | 0 | 0 |

Шаг 7. Вершина 3 имеет две смежных вершины – 5, 6. Переходим в наименьшую по весу – 5 и окрашиваем ее в серый.

0 1

2 3 4

5 6

|  |  |
| --- | --- |
| **t** | 7 |
| **C** | G | B | B | G | W | G | W |
| **D** | 1 | 2 | 3 | 4 | I | 5 | I |
| **P** | N | 0 | 0 | 0 | N | 3 | N |
| **F** | 0 | 3 | 5 | 0 | 0 | 0 | 0 |

Шаг 8. 5ая вершина имеет одну смежную – 2, но мы уже там были. Следовательно мы погрузились в глубину.

0 1

2 3 4

5 6

|  |  |
| --- | --- |
| **t** | 8 |
| **C** | G | B | B | G | W | B | W |
| **D** | 1 | 2 | 3 | 4 | I | 5 | I |
| **P** | N | 0 | 0 | 0 | N | 3 | N |
| **F** | 0 | 3 | 5 | 0 | 0 | 8 | 0 |

Шаг 9. Возвращаемся в вершину 3 и переходим в 6ую. Окрашиваем ее в серый.

0 1

2 3 4

5 6

|  |  |
| --- | --- |
| **t** | 9 |
| **C** | G | B | B | G | W | B | G |
| **D** | 1 | 2 | 3 | 4 | I | 5 | 6 |
| **P** | N | 0 | 0 | 0 | N | 3 | 3 |
| **F** | 0 | 3 | 5 | 0 | 0 | 8 | 0 |

Шаг 10. Вершина 6 имеет единственную смежную вершину – 4. Закрашиваем ее в серый.

0 1

2 3 4

5 6

|  |  |
| --- | --- |
| **t** | 10 |
| **C** | G | B | B | G | G | B | G |
| **D** | 1 | 2 | 3 | 4 | 7 | 5 | 6 |
| **P** | N | 0 | 0 | 0 | 6 | 3 | 3 |
| **F** | 0 | 3 | 5 | 0 | 0 | 8 | 0 |

Шаг 11. Вершина 4 не имеет смежных вершин, значит мы добрались до глубины, поэтому окрашиваем ее в черный.

0 1

2 3 4

5 6

|  |  |
| --- | --- |
| **t** | 11 |
| **C** | G | B | B | G | B | B | G |
| **D** | 1 | 2 | 3 | 4 | 7 | 5 | 6 |
| **P** | N | 0 | 0 | 0 | 6 | 3 | 3 |
| **F** | 0 | 3 | 5 | 0 | 11 | 8 | 0 |

Шаг 12. Возвращаемся в вершину 6. И так как она больше не имеет смежных вершин окрашиваем ее в черный.

0 1

2 3 4

5 6

|  |  |
| --- | --- |
| **t** | 12 |
| **C** | G | B | B | G | B | B | В |
| **D** | 1 | 2 | 3 | 4 | 7 | 5 | 6 |
| **P** | N | 0 | 0 | 0 | 6 | 3 | 3 |
| **F** | 0 | 3 | 5 | 0 | 11 | 8 | 12 |

Шаг 13. Возвращаемся в вершину 3. И так как она больше не имеет смежных вершин окрашиваем ее в черный.

0 1

2 3 4

5 6

|  |  |
| --- | --- |
| **t** | 13 |
| **C** | G | B | B | В | B | B | В |
| **D** | 1 | 2 | 3 | 4 | 7 | 5 | 6 |
| **P** | N | 0 | 0 | 0 | 6 | 3 | 3 |
| **F** | 0 | 3 | 5 | 13 | 11 | 8 | 12 |

Шаг 14. Возвращаемся в вершину 0. И окрашиваем ее в черный.

0 1

2 3 4

5 6

|  |  |
| --- | --- |
| **t** | 14 |
| **C** | В | B | B | В | B | B | В |
| **D** | 1 | 2 | 3 | 4 | 7 | 5 | 6 |
| **P** | N | 0 | 0 | 0 | 6 | 3 | 3 |
| **F** | 14 | 3 | 5 | 13 | 11 | 8 | 12 |

**В результате получили DFS-дерево:**

0 1

2 3 4

5 6

**Алгоритм топологической сортировки:**

**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.

Шаг 1. Начинаем с вершины 0. Окрашиваем ее в серый цвет.

0 1

2 3 4

5 6

Шаг 2. Вершина 0 имеет три смежных вершины – 1, 2, 3. Идем к вершине 1. Закрашиваем ее серым цветом.

0 1

2 3 4

5 6

Шаг 3. Вершина 6 не имеет смежных вершин, поэтому закрашиваем ее черным цветом, кладем в стек и возвращаемся в вершину 0.

0 1

2 3 4

5 6

Шаг 4. Из вершины 0 переходим в следующую вершину по весу – 2. Закрашиваем ее серым цветом.

0 1

2 3 4

5 6

Шаг 5. Переходим во вторую и красим ее в черный цвет, так как она не имеет смежных вершин. Кладем ее в стек.

0 1

2 3 4

5 6

Шаг 6. Возвращаемся в вершину 0 и переходим к 3ей. Закрашиваем ее серым цветом.

0 1

2 3 4

5 6

Шаг 7. Вершина 3 имеет две смежных вершины – 5, 6. Закрашиваем 5ую вершину в черный, так как она не имеет белых смежных вершин. Кладем ее в стек.

0 1

2 3 4

5 6

Шаг 8. Переходим в вершину 6 и закрашиваем ее серым цветом.

0 1

2 3 4

5 6

Шаг 9. Вершина 6 имеет единственную смежную вершину – 4. Закрашиваем ее серым цветом.

0 1

2 3 4

5 6

Шаг 10. Так как вершина 4 не имеет смежных вершин, закрашиваем ее черным и кладем в стек.

0 1

2 3 4

5 6

Шаг 11. Возвращаемся в вершину 6, закрашиваем ее черным цветом и кладем в стек.

0 1

2 3 4

5 6

Шаг 12. Возвращаемся в вершину 3, закрашиваем ее черным цветом и кладем в стек.

0 1

2 3 4

5 6

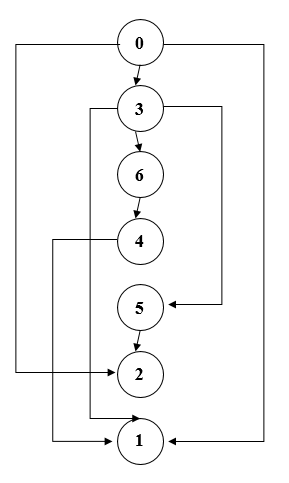
Шаг 13. Возвращаемся в вершину 0, закрашиваем ее черным цветом и кладем в стек.

0 1

2 3 4

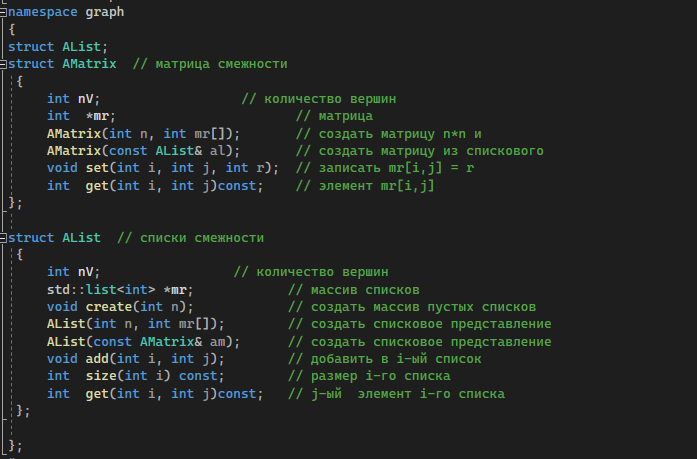
5 6

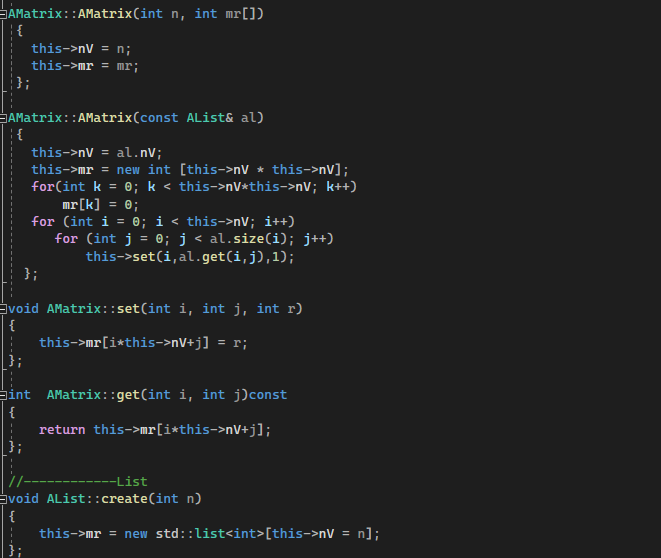
**Получим в итоге:** 1 2 5 4 6 3 0

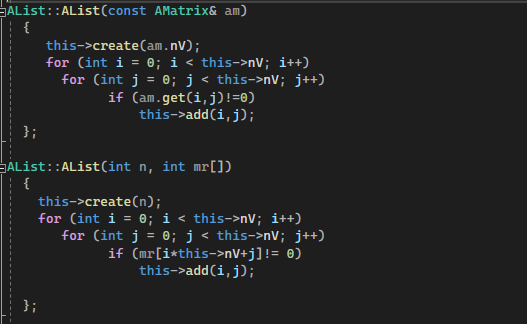
****

**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.

Ниже на *рисунках* представлен код выполнения программы.







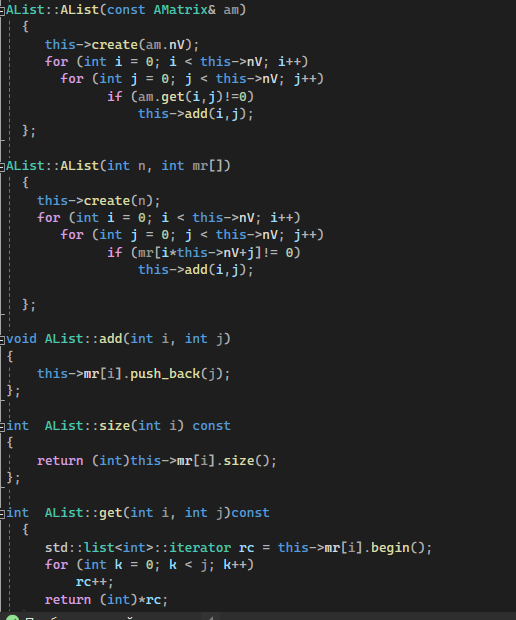
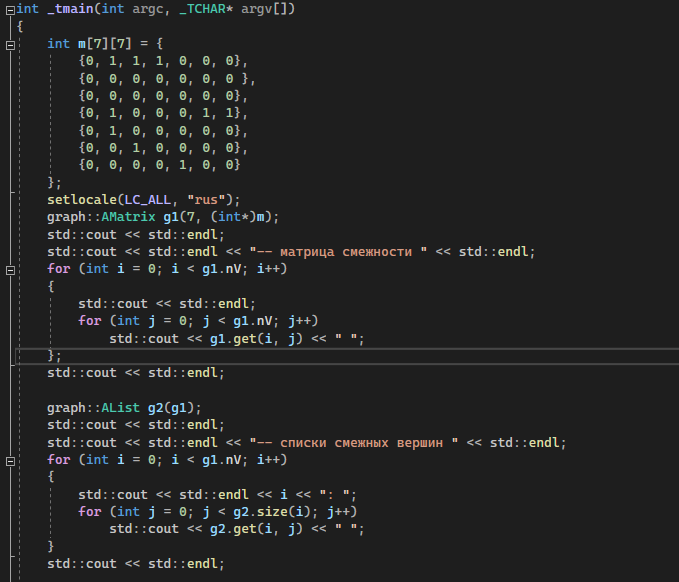


Рис. *1* – Код файла Graph.cpp



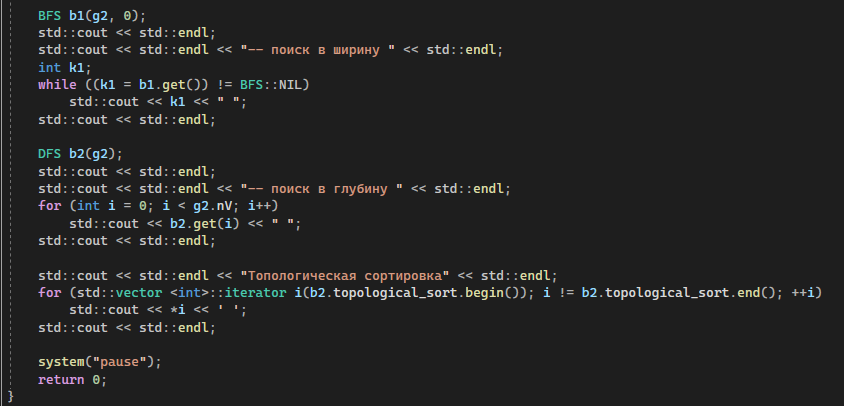
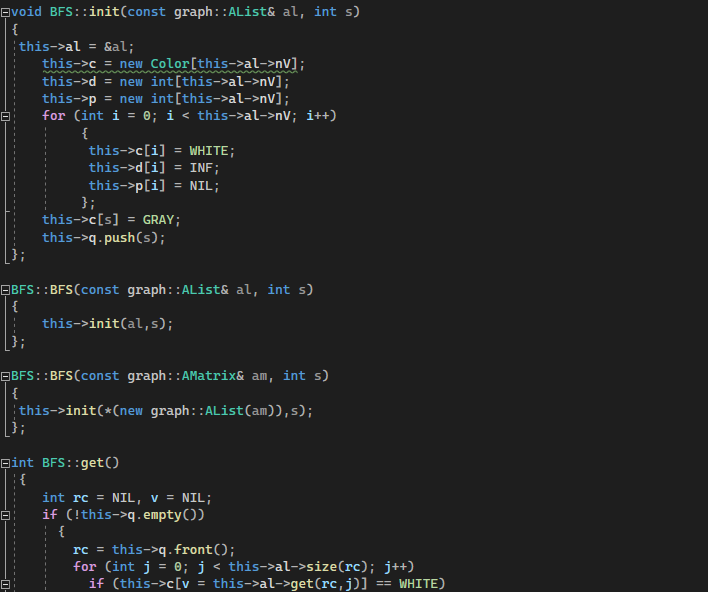


Рис. *2* – Код файла Lab5.cpp



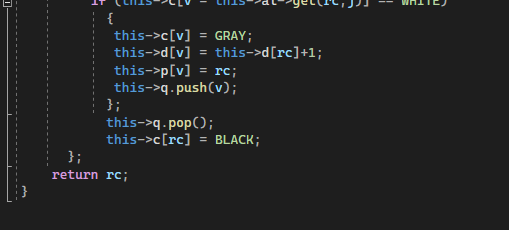


Рис. *3* – Код файла BFS.cpp

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

Ниже на *рисунках* представлен код выполнения программы.

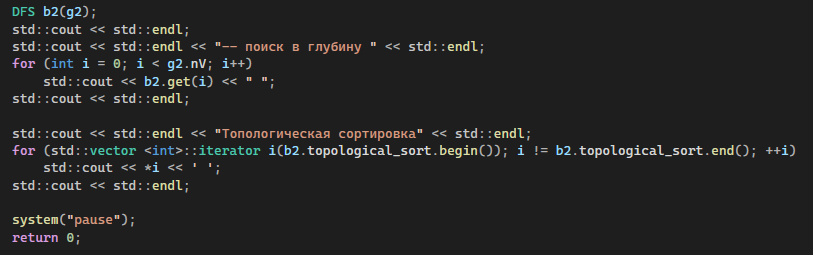
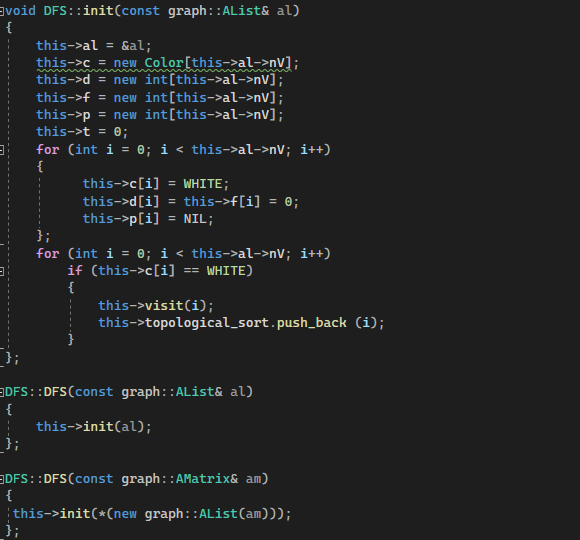


Рис. *4* – Код файла Lab5.cpp



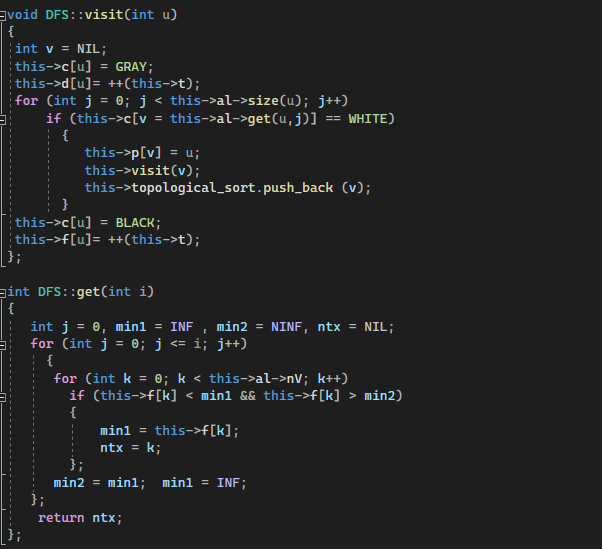


Рис. *5* – Код файла DFS.cpp

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

В предыдущем задании приведен код.

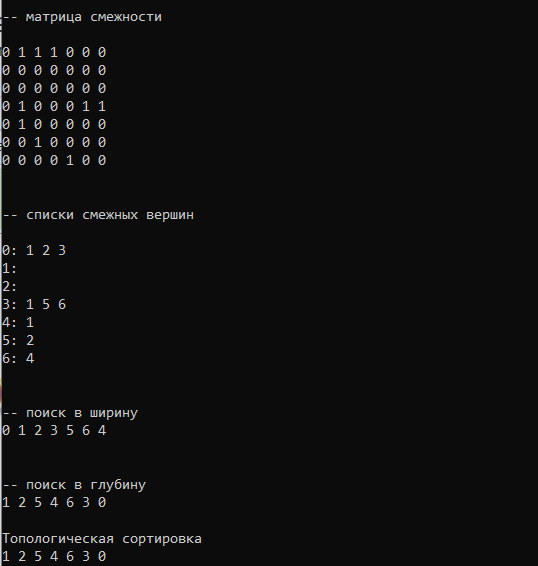
****

Рис. *6* – Результат выполнения программы

**Вывод**: Мы освоили сущность и программную реализацию способов представления графов, алгоритмов поиска в ширину и глубину, алгоритма топологической сортировки графов.

.