Учреждение образования

«Белорусский государственный технологический университет»

**Лабораторная работа №1**

Вспомогательные функции

Выполнил:

Студент 2 курса 4 группы ФИТ

Станчик Максим Андреевич

2024 г.

**ЦЕЛЬ РАБОТЫ:** приобретение навыков составления и отладки программ с использованием пользовательских функций для замера продолжительности процесса вычисления

**Задание 1**

Используя спецификации, приведенные в условии данной лабораторной работы, мной было разработаны три функции (start, dget и iget). Результат выполнения задания приведен на рисунке 1:

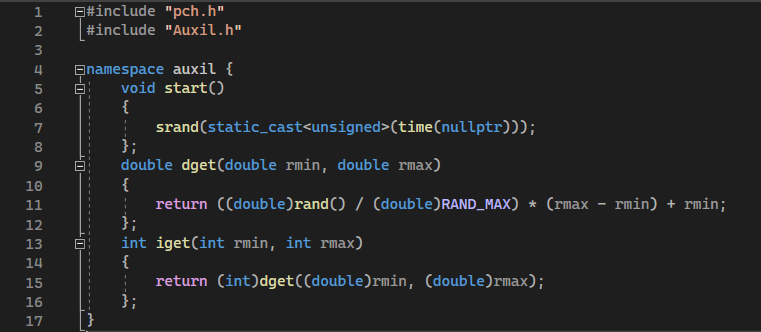


Рисунок 1 – Реализация функций

**Задание 2**

В целях проверки работоспособности разработанных функций и приобретения навыков замера продолжительности процесса вычисления мной также была реализована программа, основанная на примере, приведенном в лабораторной работе. Результат выполнения задания приведен на рисунке 2:

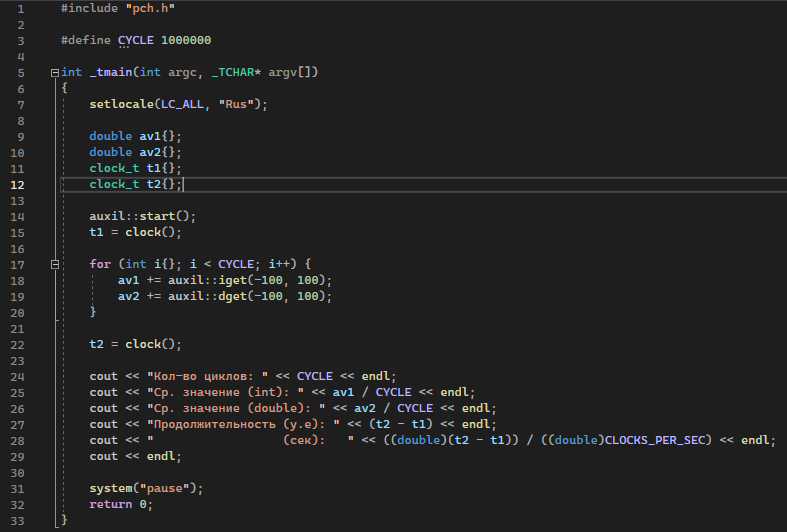


Рисунок 2 – вызов функции и измерение времени выполнения

**Задание 3**

Для выполнения задания 3 мной были проведены необходимые эксперименты и построен график зависимости продолжительности процесса вычисления от количества циклов в программе, реализованной на рисунках 1 и 2. Мной была использована программа для работы с электронными таблицами Microsoft Excel. Проанализировал характер зависимости. Результат выполнения приведен на рисунке 3:

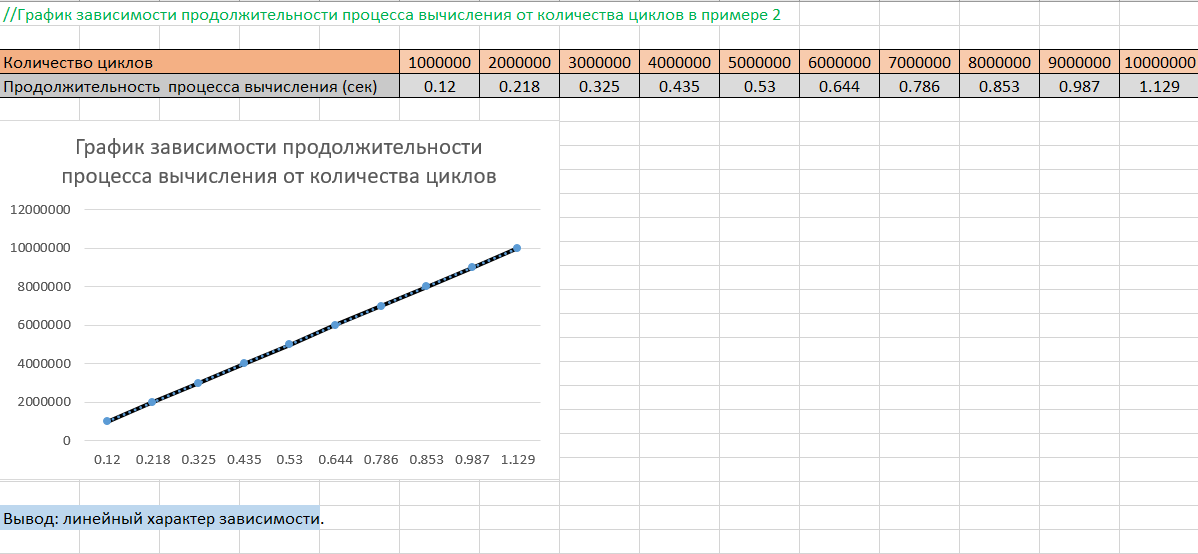


Рисунок 3 – График зависимости продолжительности процесса вычисления от количества циклов в примере 2

Также мной был реализован алгоритм для нахождения факториала. Содержимое файла, где находятся функции для работы данного алгоритма представлен на рисунке 4:

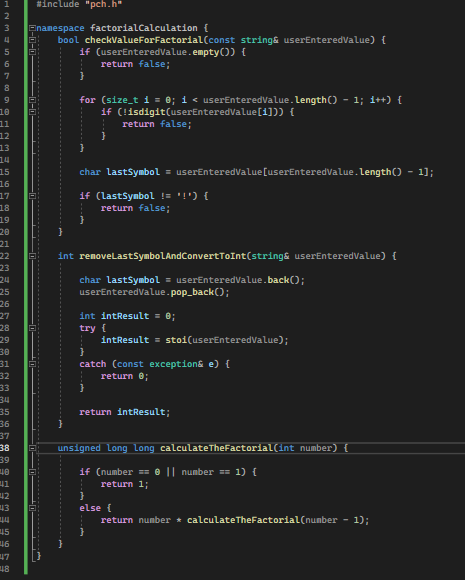


Рисунок 4 – Реализация алгоритма нахождения факториала

Вызов функций и измерение времени выполнения алгоритма находится в файле Main.cpp, после выполнения заданий 1 и 2. Содержимое файла, где находится вызов функций представлен на рисунке 5:



Рисунок 5 – Вызовы функций и измерение времени выполнения алгоритма

Также необходимо было построить график зависимости продолжительности процесса вычисления (сек) от количества итераций. Результат представлен на рисунке 6:

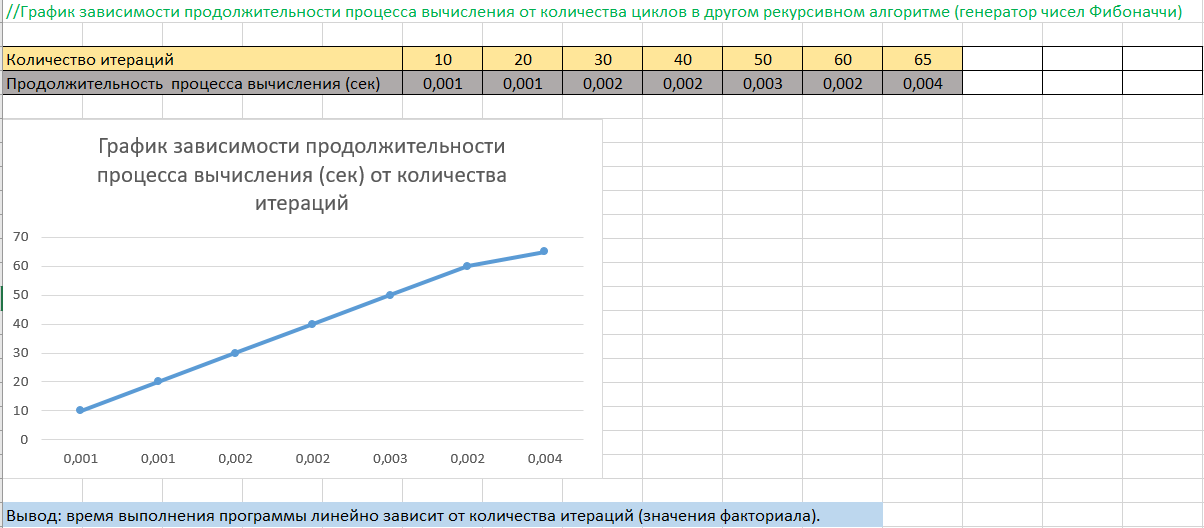


Рисунок 6 – График зависимости продолжительности процесса вычисления (сек) от количества итераций

Работоспособность программы представлена на рисунке 7:

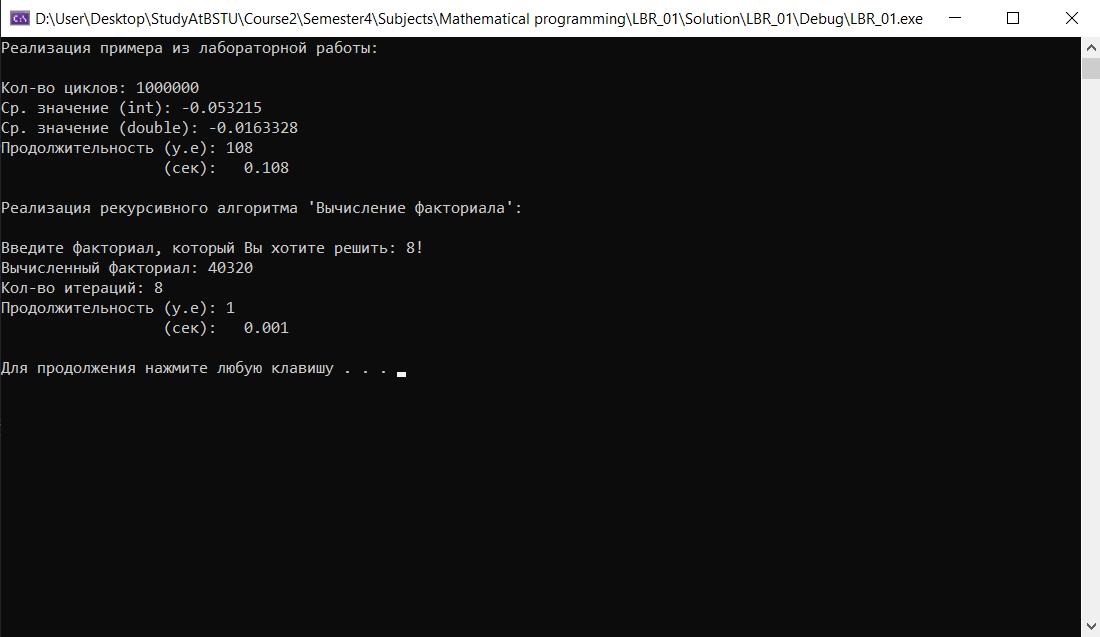


Рисунок 7 – Запуск программы

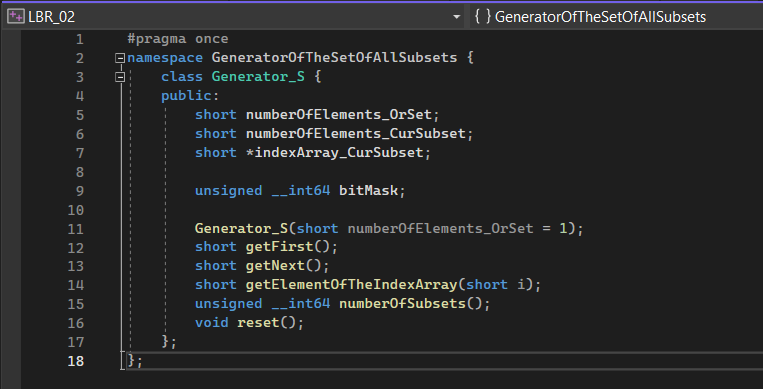
Вывод: в ходе данной лабораторной работы были использованы функции времени, которые нужны для замера чего-либо, например работы алгоритма или программы в целом, разработан алгоритм нахождения факториала, разработаны графики зависимости, первый--график зависимости продолжительности процесса вычисления от количества циклов, второй--график зависимости продолжительности процесса вычисления (сек) от количества итераций.

**Лабораторная работа №2**

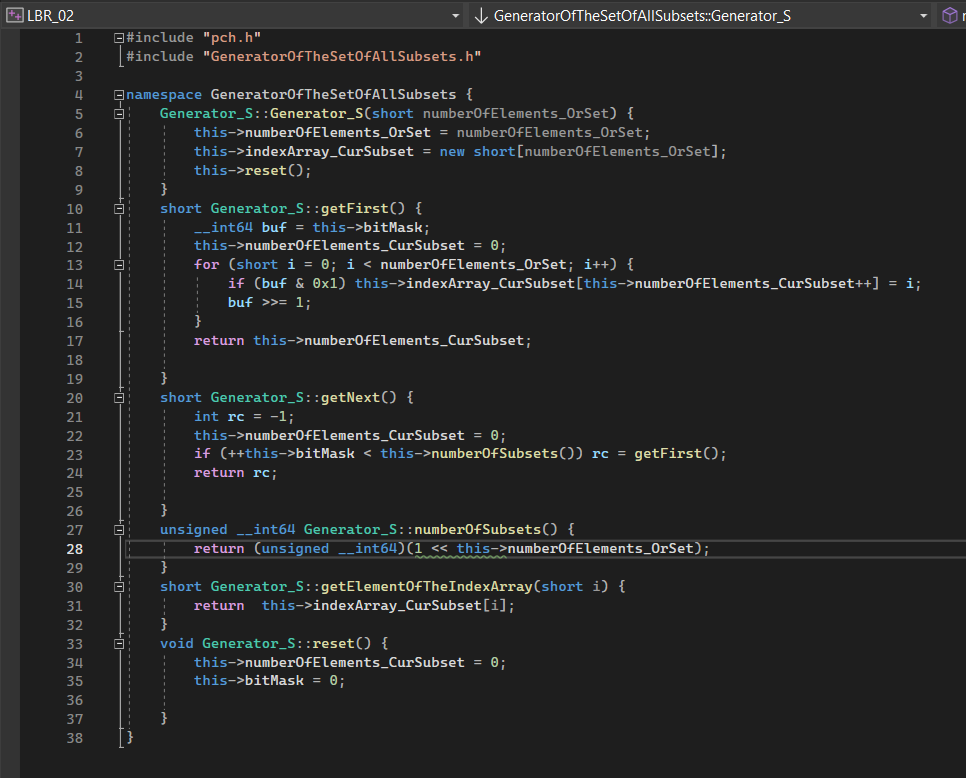
Комбинаторные алгоритмы решения оптимизационных задач

**Задание 1.** Разработать генератор подмножеств заданного множества.

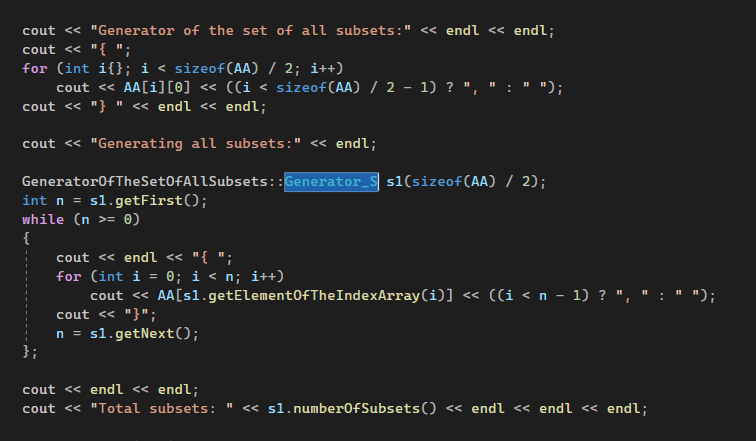
В этом задании было необходимо разработать генератор подмножеств заданного множества. Ниже на рисунках представлены заголовочные файлы и файлы cpp с реализацией поставленной задачи.



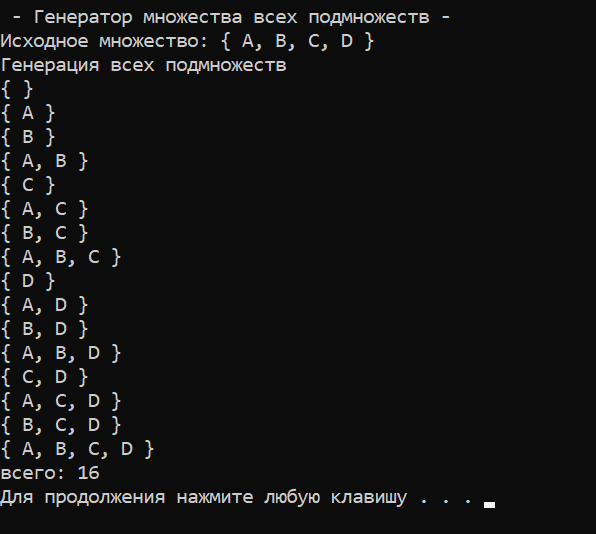
Файл GeneratorOfTheSetOfAllSubsets.h



Файл GeneratorOfTheSetOfAllSubsets.cpp



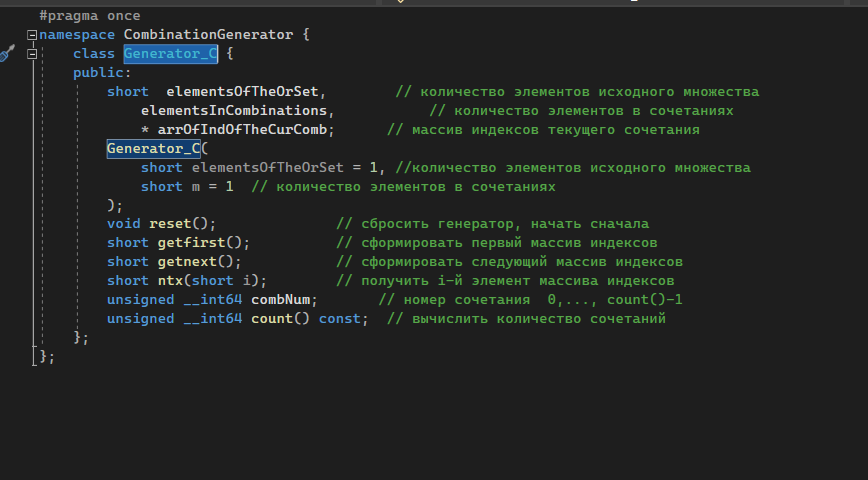
Использование в Main()



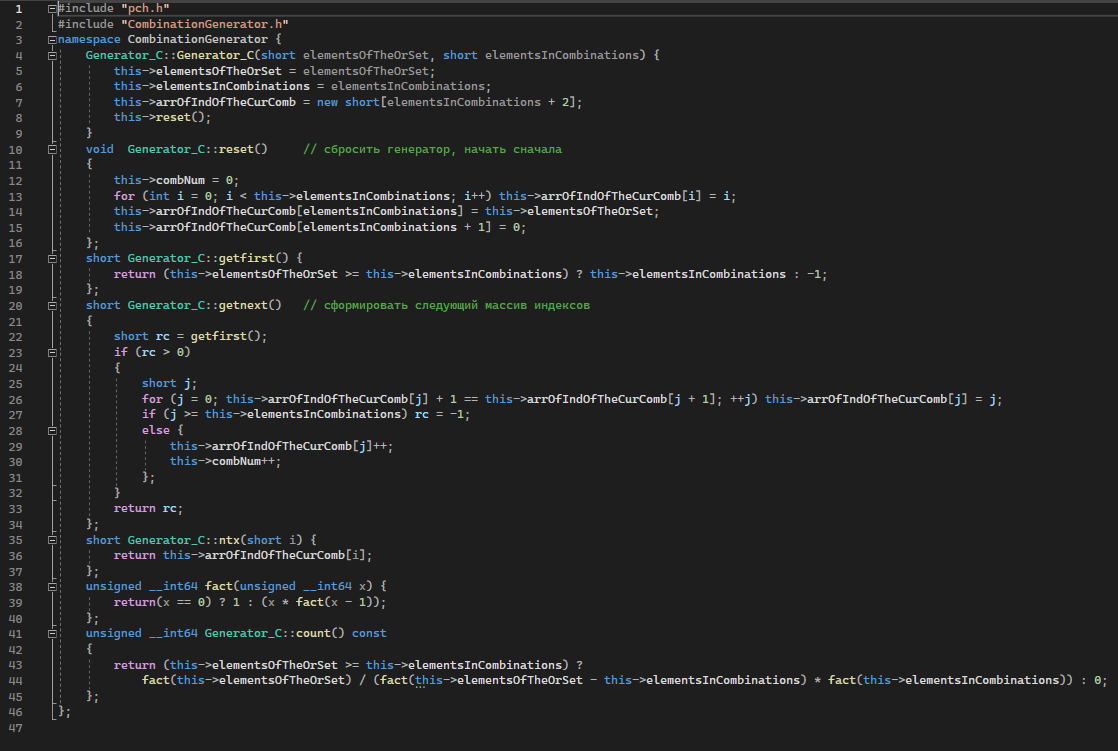
Результат выполнения

**Задание 2.** Разработать генератор сочетаний.

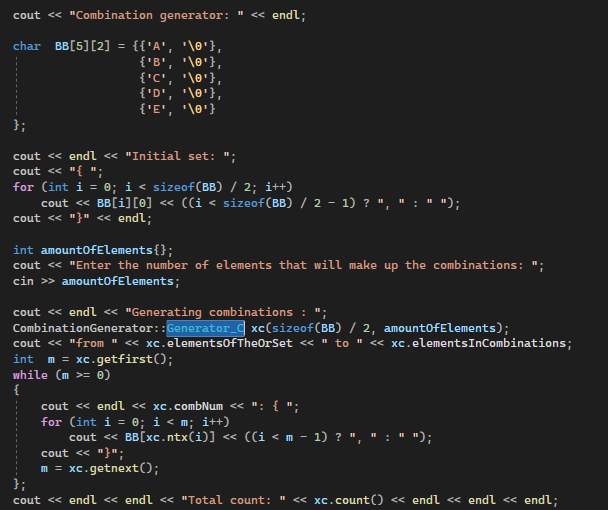
В этом задании было необходимо разработать генератор сочетаний. Ниже на рисунках представлены заголовочные файлы и файлы cpp с реализацией поставленной задачи.



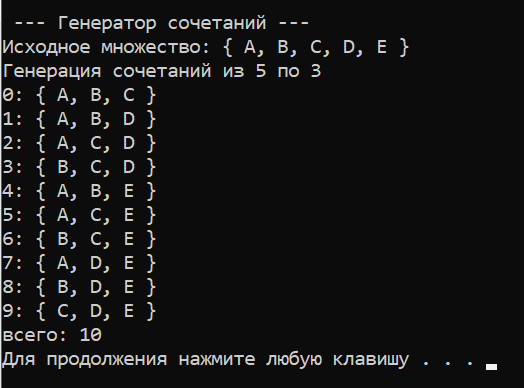
Файл CombinationGenerator.h



Файл CombinationGenerator.cpp



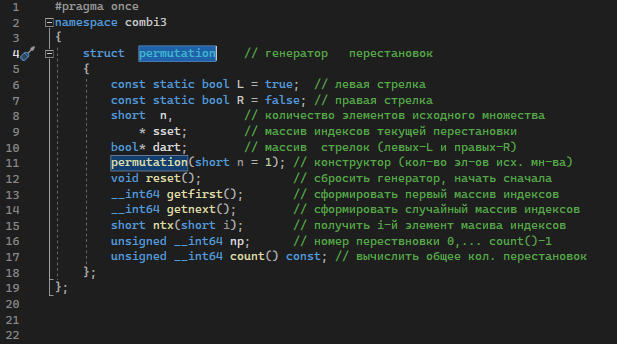
Использование в Main()



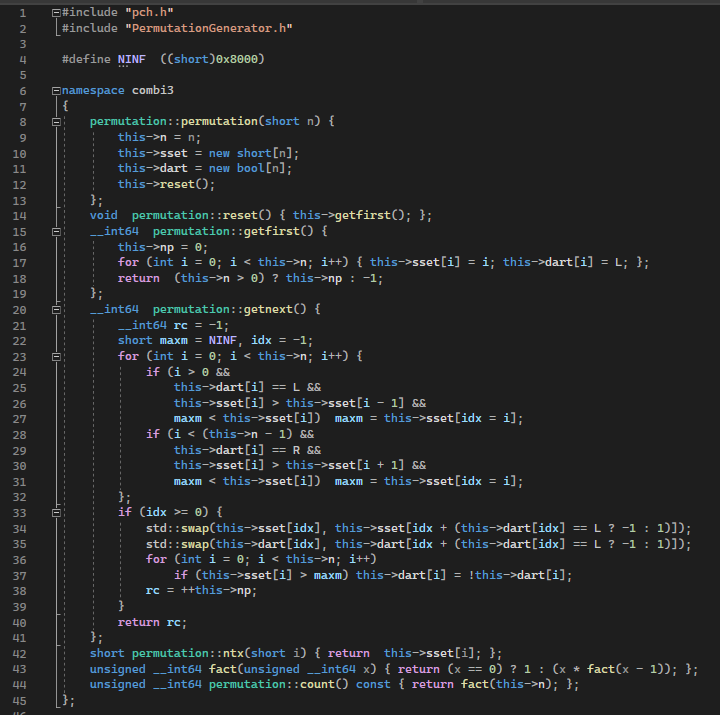
Результат выполнения

**Задание 3.** Разработать генератор перестановок.

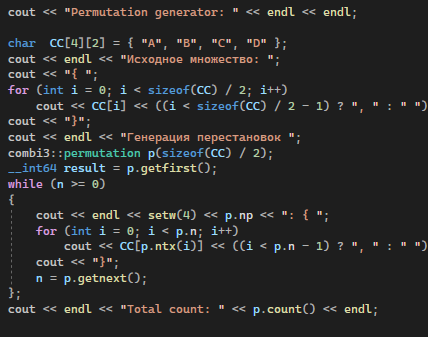
В этом задании было необходимо разработать генератор перестановок. Ниже на рисунках представлены заголовочные файлы и файлы cpp с реализацией поставленной задачи.



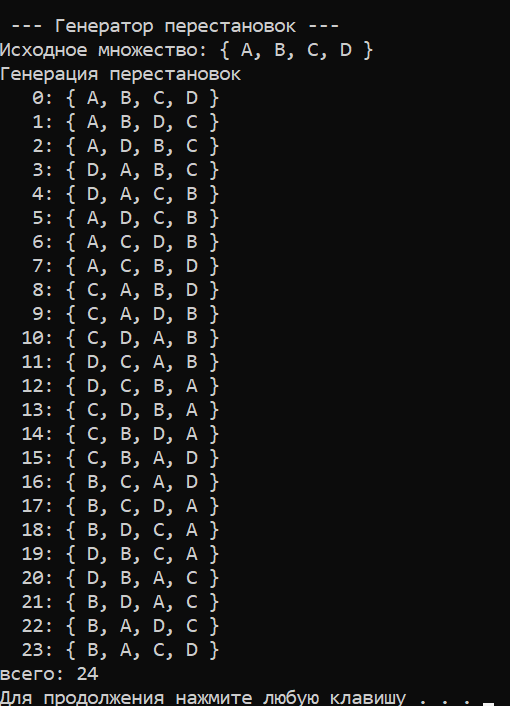
Файл PermutationGenerator.h



Файл PermutationGenerator.cpp



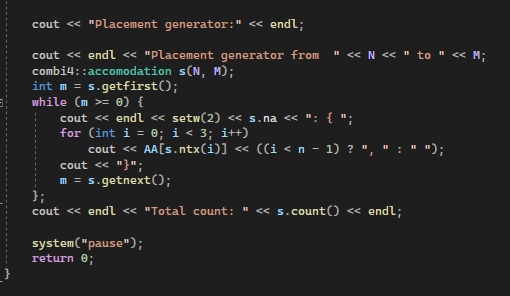
Использование в Main()



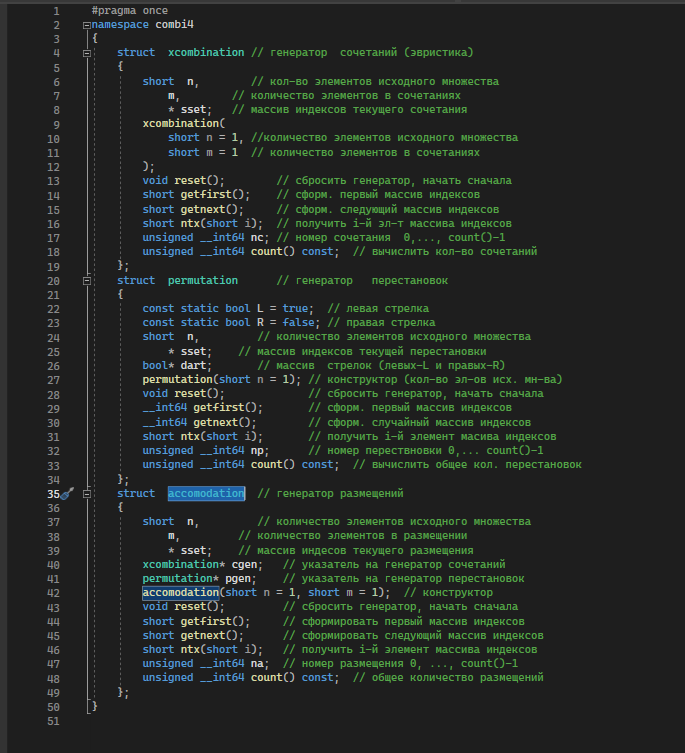
Результат выполнения

**Задание 4.** Разработать генератор размещений.

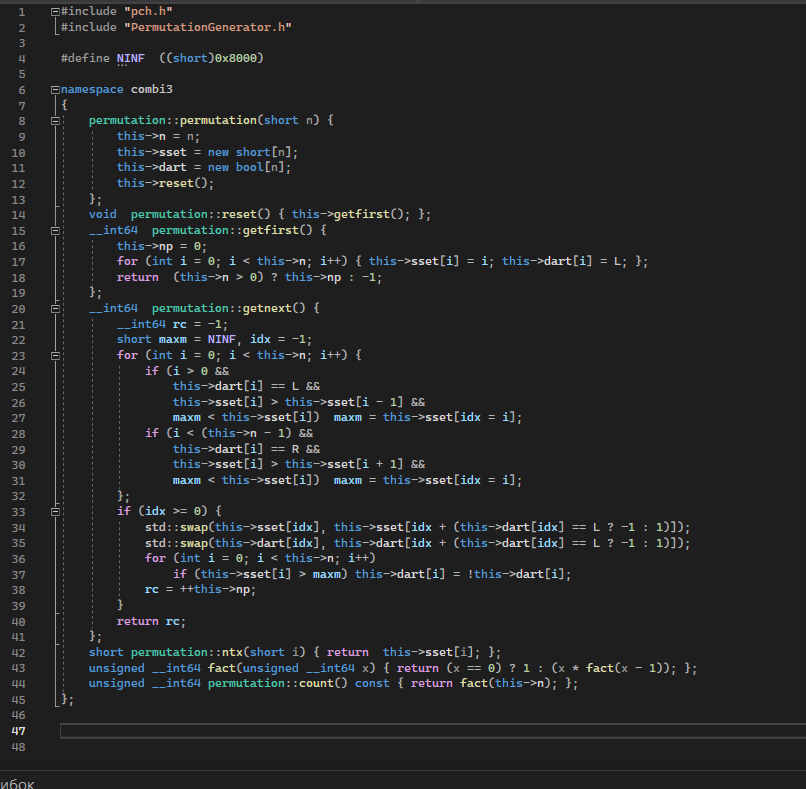
В этом задании было необходимо разработать генератор размещений. Ниже на рисунках представлены заголовочные файлы и файлы cpp с реализацией поставленной задачи.



Использование генератора размещений в Main()



Файл PermutationGenerator.h



Файл PermutationGenerator.cpp

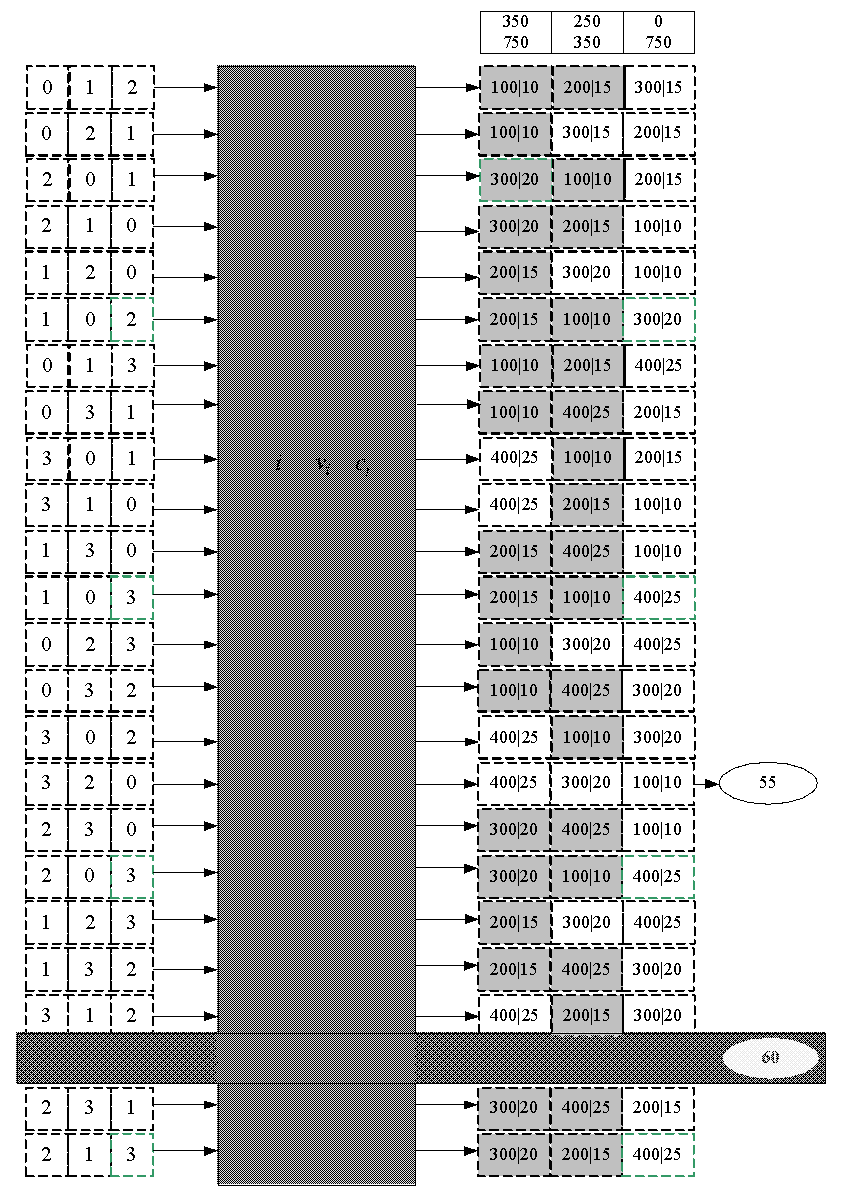


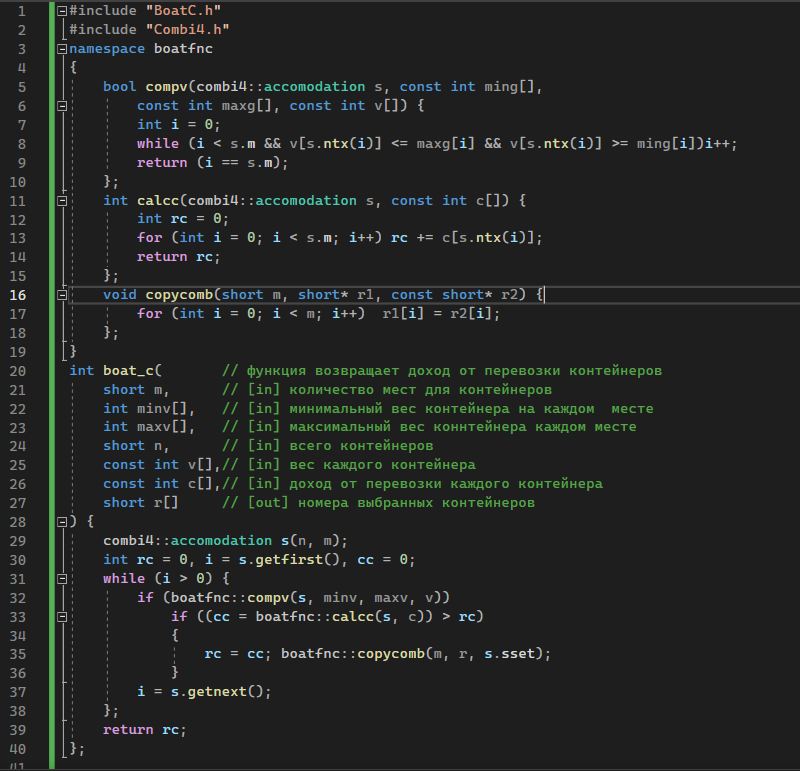
Результат выполнения

**Задание 5.** Решить в соответствии с вариантом задачу и результат занести в отчет. У меня восьмой вариант, поэтому условие задачи следующее:

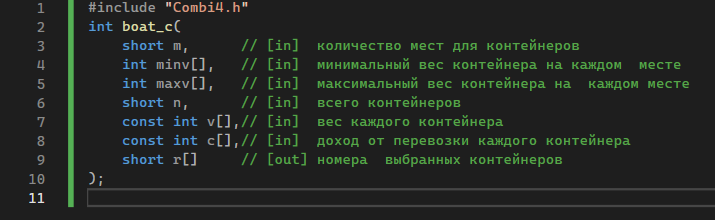
об оптимальной загрузке судна с условием центровки (веса контейнеров сгенерировать случайным образом: количество мест на судне для контейнеров – 5, количество контейнеров 8, веса контейнеров 100 – 200 кг., доход от перевозки 10 – 100 у.е.; минимальный вес контейнера для каждого места 50 – 120 кг, максимальный вес контейнера для каждого места 150 – 850 кг);

Для решения данного задания использовался генератор размещений, схема которого представлена на рисунке:





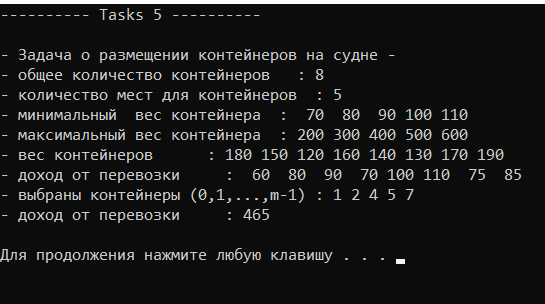
Файл BoatC.cpp



Файл BoatC.h



Использование в Main()



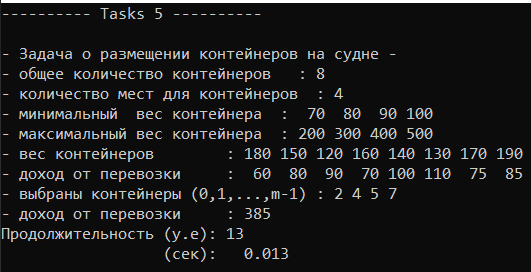
Результат решения задачи

**Задание 6.** Исследовать зависимость времени вычисления необходимое для решения задачи от размерности задачи и результат в виде графика:

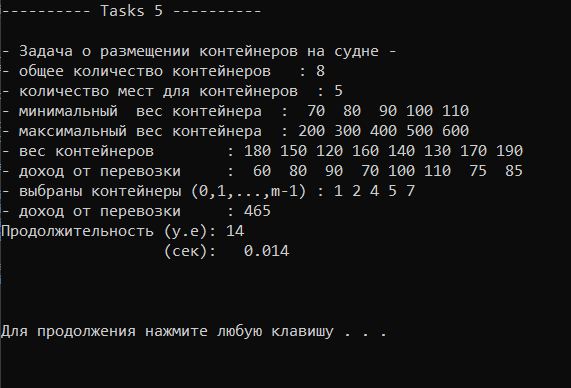
Об оптимальной загрузке судна с условием центровки (количество мест на судне для контейнеров 4 – 8);

Пусть x = количество мест на судне для контейнеров. Тогда:

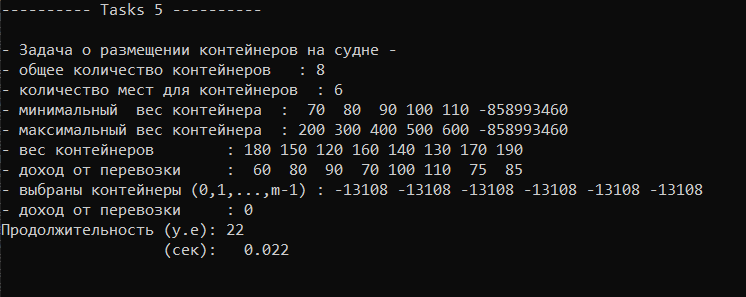
При х = 4:



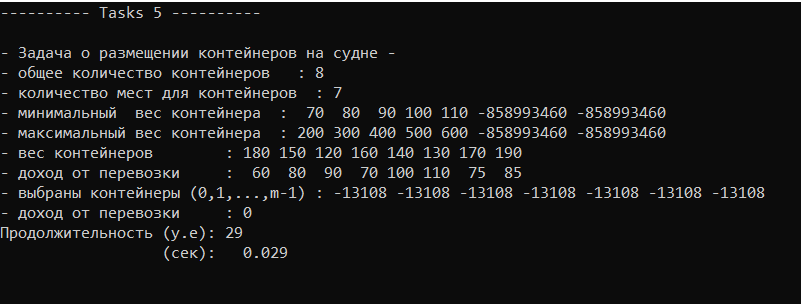
При х = 5:



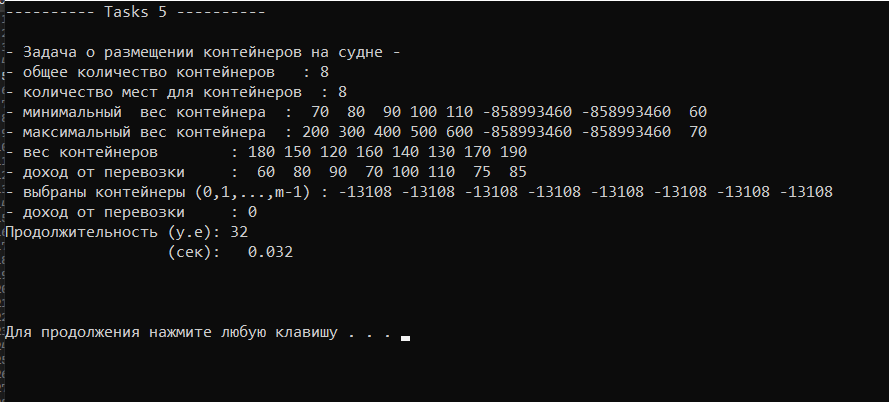
При х = 6:

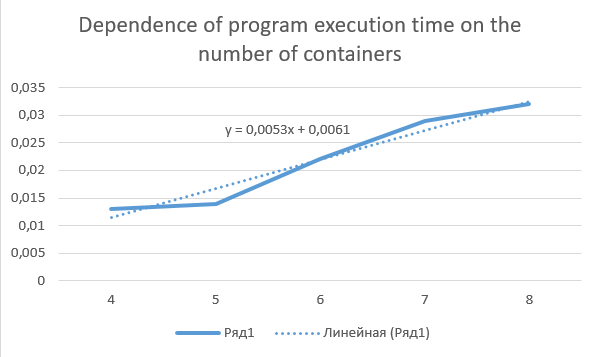


При х = 7:



При х = 8:





Вывод: исходя из полученных данных и графика, можно заметить, что скорость выполнения программы линейно возрастает при добавлении количества мест для контейнеров. Уравнение функции: y = kx +b.

//сочетание размещение перестановки и что такое генератор

Учреждение образования

«Белорусский государственный технологический университет»

**Лабораторная работа №3**

Метод ветвей и границ. Задача коммивояжера и методы её решения.

Выполнил:

Студент 2 курса 4 группы ФИТ

Станчик Максим Андреевич

2024 г.

**ЦЕЛЬ РАБОТЫ:** освоить общие принципы решения задач методом ветвей и границ, решить задачу о коммивояжере данным методом, сравнить полученное решение задачи с комбинаторным методом перестановок.

**Задание 1.**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Город** | **1** | **2** | **3** | **4** | **5** |
| **1** |  | 16 | 29 |  | 8 |
| **2** | 8 |  | 23 | 60 | 76 |
| **3** | 10 | 24 |  | 86 | 57 |
| **4** | 25 | 50 | 32 |  | 24 |
| **5** | 85 | 74 | 52 | 21 |  |

*Вариант 8*

**Задание 2.**

В каждой строке (городе) необходимо найти минимальное число. После чего от каждого элемента рассматриваемой строки необходимо отнять минимальное число этой строки.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Город** | **1** | **2** | **3** | **4** | **5** |
| **1** |  | 16 | 29 |  | 8 |
| **2** | 8 |  | 23 | 60 | 76 |
| **3** | 10 | 24 |  | 86 | 57 |
| **4** | 25 | 50 | 32 |  | 24 |
| **5** | 85 | 74 | 52 | 21 |  |

|  |  |
| --- | --- |
| **Строка** | **Минимальное значение (константа приведения)** |
| **1** | 8 |
| **2** | 8 |
| **3** | 10 |
| **4** | 24 |
| **5** | 21 |

Сумма минимальных значений в строках: 8 + 8 + 10 + 24 + 21 = 71 – константа приведения;

От каждого элемента отняли минимальное значение строки, в которой находится рассматриваемый элемент. Это называется приведение таблицы по строке.

Изменение всех элементов столбца матрицы расстояний на одно и то же число не влияет на выбор оптимального маршрута коммивояжера.

Преобразованная таблица:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Город** | **1** | **2** | **3** | **4** | **5** |
| **1** |  | 8 | 21 |  | 0 |
| **2** | inf |  | 15 | 52 | 68 |
| **3** | 0 | 14 |  | 76 | 47 |
| **4** | 1 | 26 | 8 |  | 0 |
| **5** | 64 | 53 | 31 | 0 |  |

В каждом столбце нашли минимальное число. От каждого элемента в столбце отняли это число.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Номер столбца** | **1** | **2** | **3** | **4** | **5** |
| **Минимальное значение (константа приведения)** | 0 | 8 | 8 |  | 0 |

Сумма минимальных значений в столбцах: 16 – константа приведения;

От каждого элемента отняли минимальное значение столбца, в котором находится рассматриваемый элемент. Это называется приведение таблицы по столбцу.

Полностью приведенная таблица:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Город** | **1** | **2** | **3** | **4** | **5** |
| **1** |  | 0 | 13 |  | 0 |
| **2** | 0 |  | 7 | 52 | 68 |
| **3** | 0 | 6 |  | 76 | 47 |
| **4** | 1 | 18 | 0 |  | 0 |
| **5** | 64 | 45 | 23 | 0 |  |

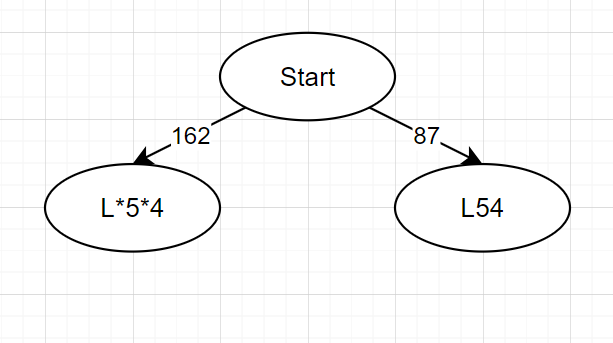
Складывая константы приведения таблицы получаем нижнюю границу

Сумма констант приведения: 71 + 16 = 87 – нижняя граница;

Кол-во расстояний между городами = кол-во нулевых элементов в таблице.

|  |  |  |
| --- | --- | --- |
| **Расстояния между городами** | **Вычисление значения** | **Значение** |
| **L21** | 0 + 7 | 7 |
| **L31** | 0 + 6 | 6 |
| **L12** | 0 + 6 | 6 |
| **L43** | 7 + 0 | 7 |
| **L54** | 52 + 23 | 75 |
| **L15** | 0 + 0 | 0 |
| **L45** | 0 + 0 | 0 |

Самое большое значение имеет L54



Start – начало

Если выберем L54, необходимо будет потратить 87. Если выберем не L54, то есть L\*5\*4, нужно будет потратить 162. Т.к. 87 < 162, выбираем L54. В таком случае удаляем из таблицы строку 5 и столбец 4. Значение на строке 4 и столбце 5 станет INF. Тогда таблица имеет вид:

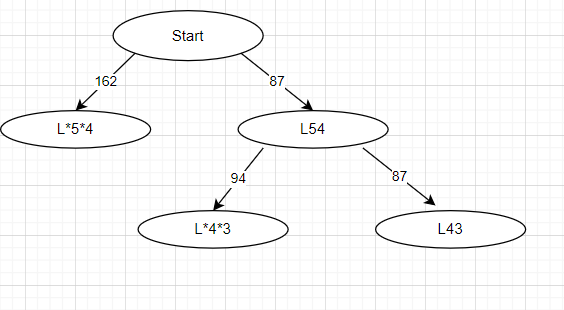
|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Город** | **1** | **2** | **3** | **5** |
| **1** |  | 0 | 13 | 0 |
| **2** | 0 |  | 7 | 68 |
| **3** | 0 | 6 |  | 47 |
| **4** | 1 | 18 | 0 | 0 |

|  |  |  |
| --- | --- | --- |
| **Расстояния между городами** | **Вычисление значения** | **Значение** |
| **L21** | 0 + 0 | 0 |
| **L31** | 6 + 0 | 6 |
| **L12** | 6 + 0 | 6 |
| **L43** | 0 + 7 | 7 |
| **L15** | 0 + 0 | 0 |
| **L45** | 0 + 0 | 0 |

Сумма минимальных значений по столбцам: 0;

Сумма минимальных значений по строкам: 0;

Необходимо использовать L43



Получили таблицу:

|  |  |  |  |
| --- | --- | --- | --- |
| **Город** | **1** | **2** | **5** |
| **1** |  | 0 | 0 |
| **2** | 0 |  | 68 |
| **3** | 0 | 6 | 47 |

|  |  |
| --- | --- |
| **Строка** | **Минимальное значение (константа приведения)** |
| **2** | 0 |
| **3** | 0 |
| **4** | 0 |

|  |  |
| --- | --- |
| **Столбец** | **Минимальное значение (константа приведения)** |
| **1** | 0 |
| **2** | 0 |
| **5** | 0 |

Сумма минимальных значений в строках: 0 + 0 + 0 = 0;

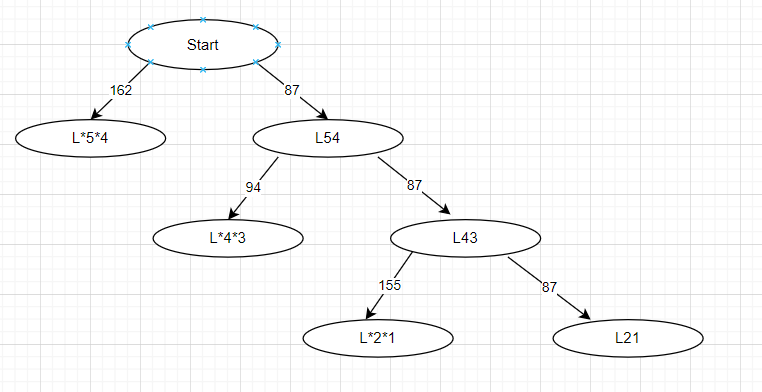
Сумма минимальных значений в столбцах: 0 + 0 + 0 = 0;

Сумма констант приведения: 0 + 0 = 0;

87 + 0 = 87;

|  |  |  |
| --- | --- | --- |
| **Расстояния между городами** | **Вычисление значения** | **Значение** |
| **L21** | 68 + 0 | 68 |
| **L31** | 0 + 6 | 6 |
| **L12** | 0 + 6 | 6 |
| **L15** | 47 + 0 | 47 |

Необходимо использовать L21.



Получили таблицу:

|  |  |  |
| --- | --- | --- |
| **Город** | **2** | **5** |
| **1** | 0 | 0 |
| **3** | 6 | 47 |

|  |  |
| --- | --- |
| **Строка** | **Минимальное значение (константа приведения)** |
| **1** | 0 |
| **3** | 6 |

|  |  |
| --- | --- |
| **Столбец** | **Минимальное значение (константа приведения)** |
| **1** | 0 |
| **2** | 0 |

Сумма минимальных значений в строках: 0 + 6 = 6;

Сумма минимальных значений в столбцах: 0 + 0 = 0

Сумма констант приведения: 0 + 6 = 6;

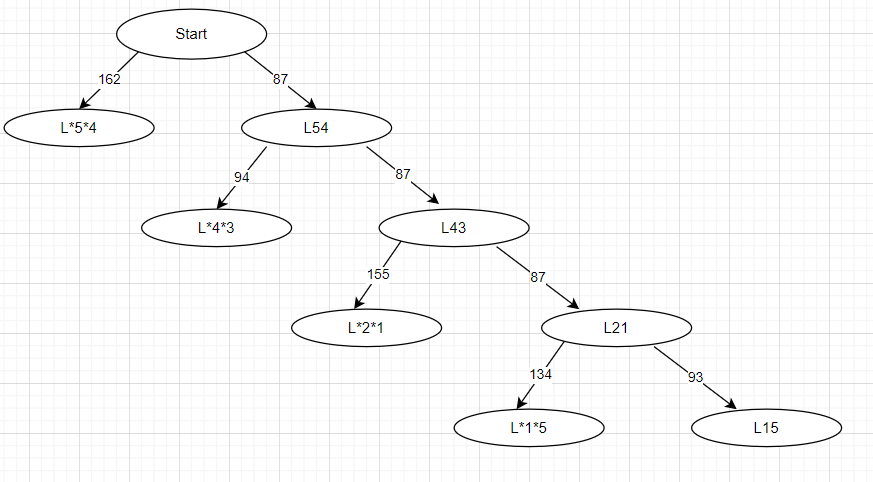
87 + 6 = 93;

Получили таблицу:

|  |  |  |
| --- | --- | --- |
| **Город** | **2** | **5** |
| **1** | 0 | 0 |
| **3** | 0 | 41 |

|  |  |  |
| --- | --- | --- |
| **Расстояния между городами** | **Вычисление значения** | **Значение** |
| **L12** | 0 + 0 | 0 |
| **L15** | 0 + 41 | 41 |
| **L32** | 0 + 41 | 41 |

Выбираем L15.

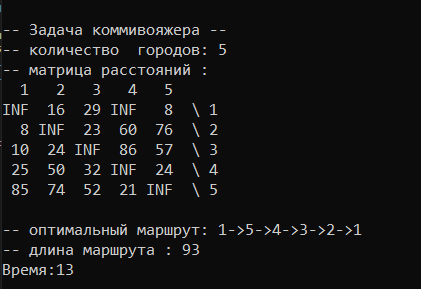


Получили таблицу:

|  |  |
| --- | --- |
| **Город** | **2** |
| **3** | 0 |

Ответ: 93.

После проверки результата при помощи генератора перестановок из лабораторной работы 2, полученные ответы совпадали с исходным решением.



Вывод: алгоритм ветвей и границ подходит для решения общей задачи целочисленного линейного программирования. Метод является вариацией полного перебора с отсевом подмножеств допустимых решений, заведомо не содержащих оптимальных решений. Применяется в решении задачи о коммивояжере, минимаксных задачах о назначениях, задачи календарного планирования, задачи о трех станках (вероятностные задачи).

**Лабораторная работа №4**

Динамическое программирование.

Выполнил:

Студент 2 курса 4 группы ФИТ

Станчик Максим Андреевич

2024 г.

**Задание 1**

На языке С++ сгенерировать случайным образом строку букв латинского алфавита S1 длиной 300 символов и S2 длиной 200.

На рисунке 1 представлена функция для генерации строк:

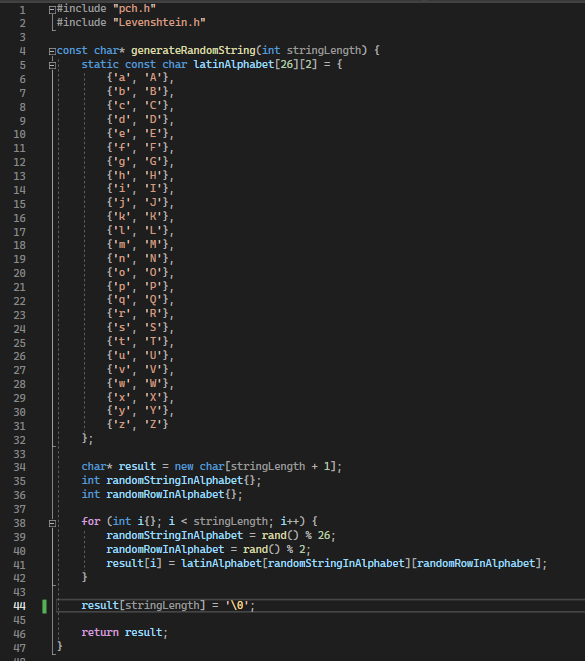


Рисунок 1 – Функция для генерации строк

На рисунке 2 представлен код, где эта функция вызывается для заполнения двух строковых переменных нужными значениями.

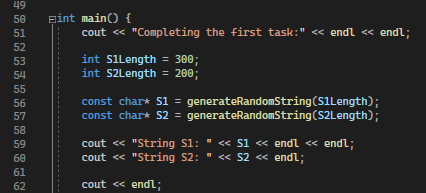
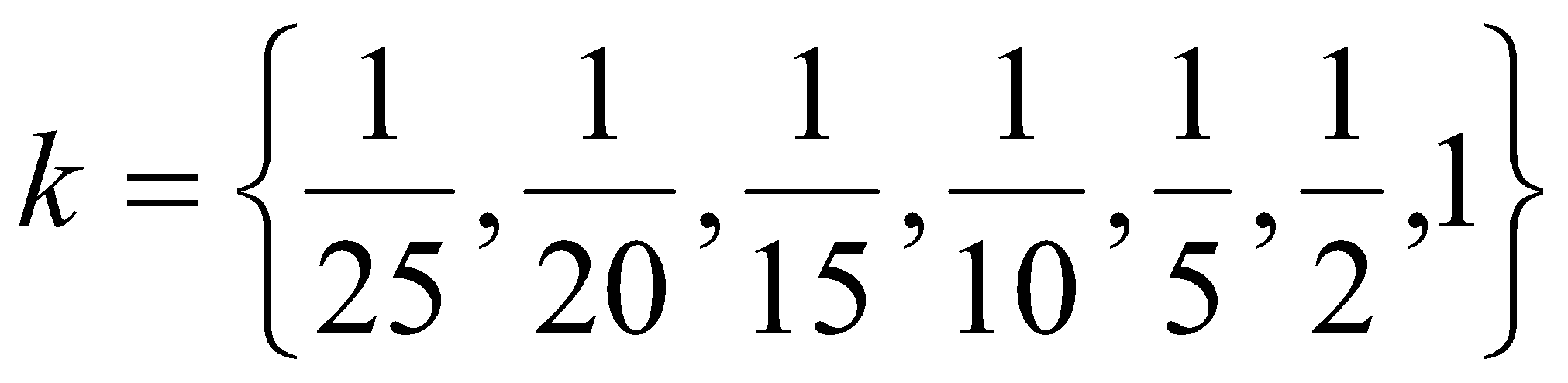


Рисунок 2 – Вызов функций в int main()

**Задание 2**

Необходимо при помощи двух способов вычислить дистанцию Левенштейна для . Код выполнение программы представлен на рисунках ниже.

На рисунке 3 представлено содержание файла Levenshtein.h

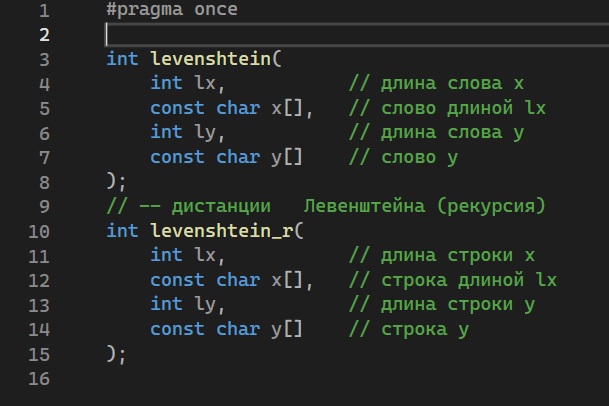


Рисунок 3 – прототипы функций

На рисунке 4 представлено содержание файла Levenshtein.cpp

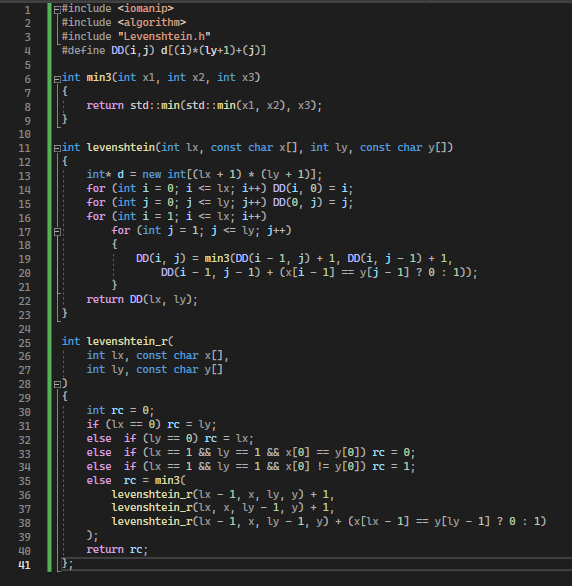


Рисунок 4 – реализация функций для вычисления расстояния Левенштейна

Результат выполнения программы представлен на рисунке 5.

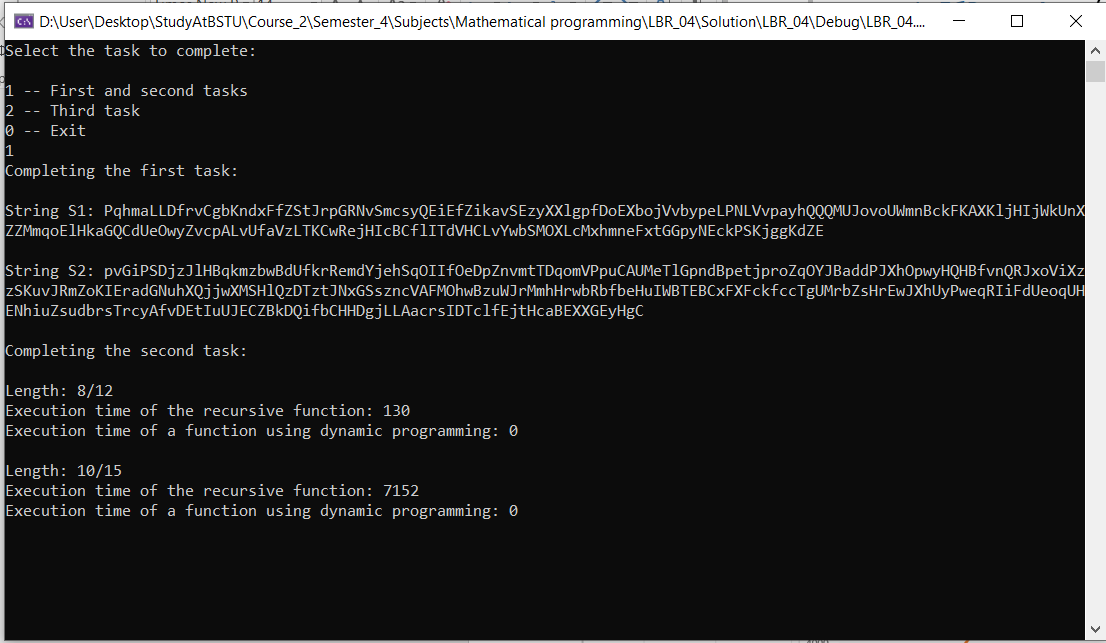


Рисунок 5 – результат выполнения программы

**Задание 3**

Выполнить сравнительный анализ времени затраченного на вычисление дистанции Левенштейна для двух методов решения. Построить графики зависимости времени вычисления от k. (копии экрана и график вставить в отчет).

На рисунке 6 представлен график зависимости выполнения.

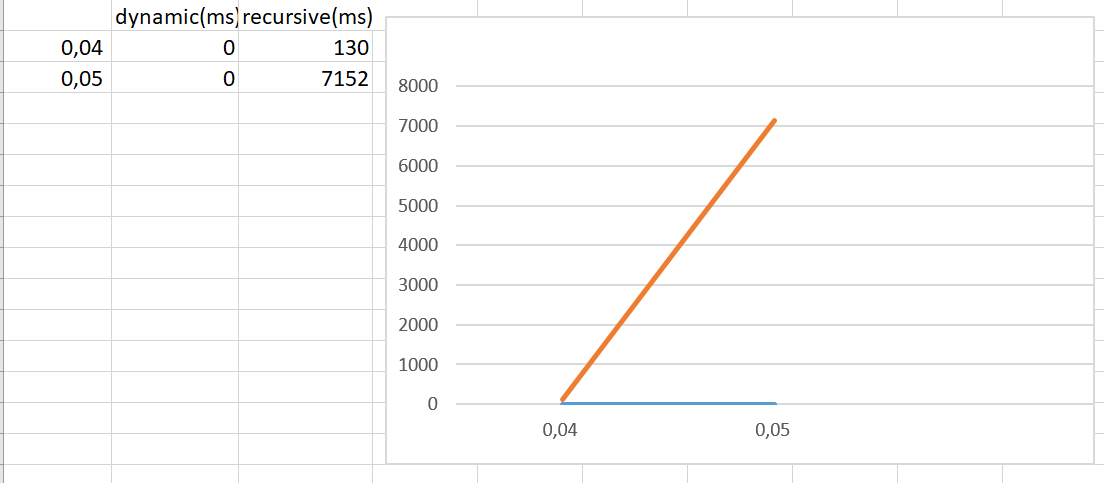


Рисунок 6 – график зависимости выполнения

***Задание 4.***

Реализовать вручную пример вычисления дистанции Левенштейна при помощи рекурсивного алгоритма. Ниже представлен алгоритм решения последовательно:



Рассмотрим 3 варианта:

1. Изменение первого
2. Изменение второго
3. Удаление обоих слов

= 5.

= 4.



= 3.

= 2.

= 2.

= 1.

= 3.

= 2.

= 1.

= 0.

16. L(“B”, “Б”) = min(2,2,1) = 1.

17. L(“Ва”,”Б”) = min(2,3,2) = 2.

18. L(“Вар”,”Б”) = min(3,4,3) = 3.

19. L(“В”,”Ба”) = min(3,2,2) = 2.

20. L(“Ва”,”Ба”) = min(3,3,1) = 1.

21. L(“Вар”,”Ба”) = min(2,4,3) = 2.

22. L(“В”,”Бар”) = min(4,3,3) = 3.

23. L(“Ва”,”Бар”) = min(4,2,3) = 2.

24. L(“Вар”,”Бар”) = min(3,3,1) = 1.

1. L(“В”,”Бара”) = min(5,4,4) = 4.
2. L(“В”,”Баран”) = min(6,5,5) = 5.
3. L(“Ва”,”Бара”) = min(5,3,3) = 3.
4. L(“Вар”,”Бара”) = min(4,2,3) = 2.
5. L(“Ва”,”Баран”) = min(6,4,5) = 4.
6. L(“Вар”,”Баран”) = min(5,3,4) = 3.

Условие: Вар Баран

Решение:

1. Вар Варан
2. Вара Варан
3. Варан Варан

Ответ: 3.

***Задание 5.***

Выполнить сравнительный анализ времени затраченного на решение задачи об оптимальной расстановке скобок при умножении нескольких матриц для двух методов решения (рекурсивное решение, динамическое программирование). Размерность матриц взять в соответствии с вариантом. Объяснить в отчете принцип расставления скобок по итоговой матрице + код + копии экрана.





Рисунок 7 – вызов функций в main()

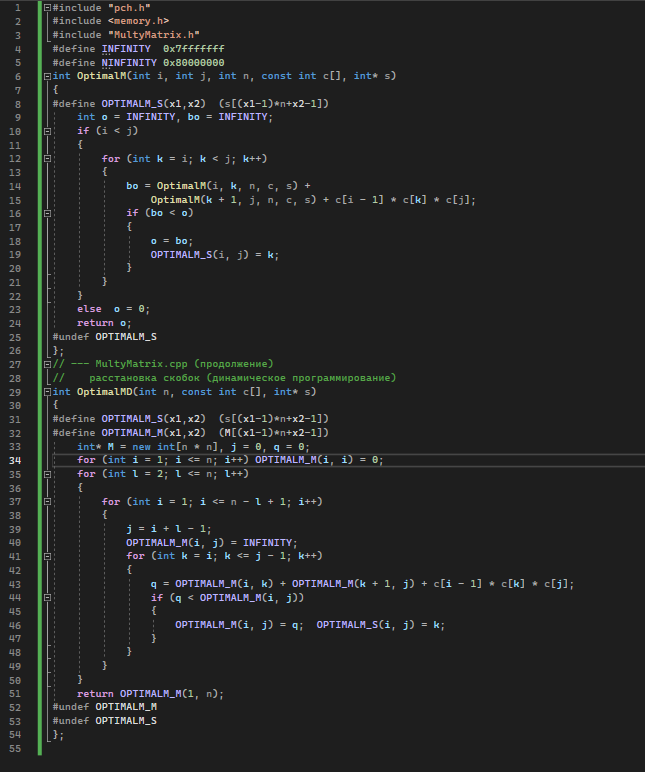


Рисунок 8 – содержание файла MultyMatrix.cpp

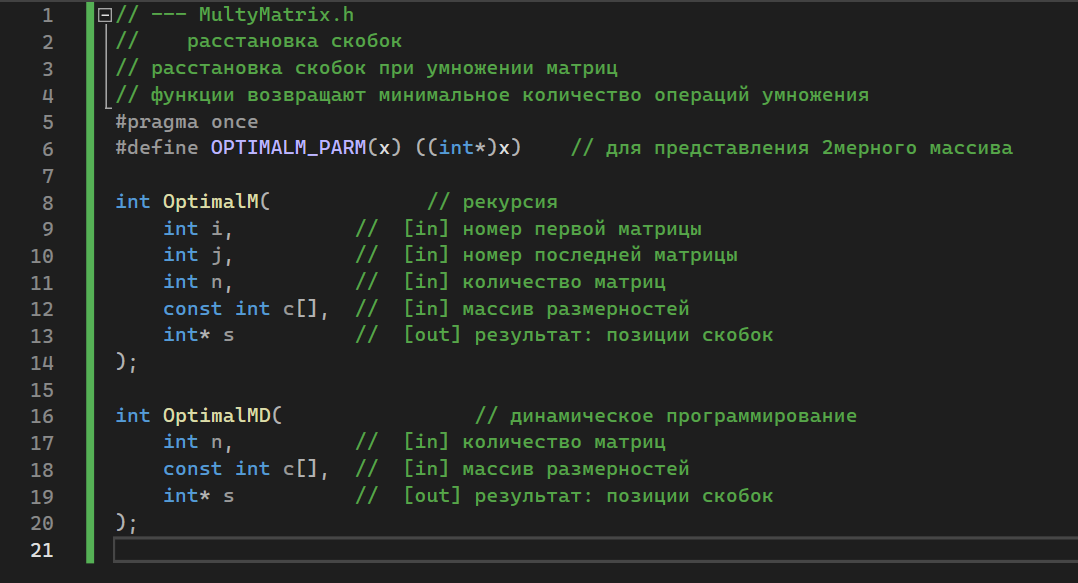


Рисунок 9 – содержание файла MultyMatrix.h

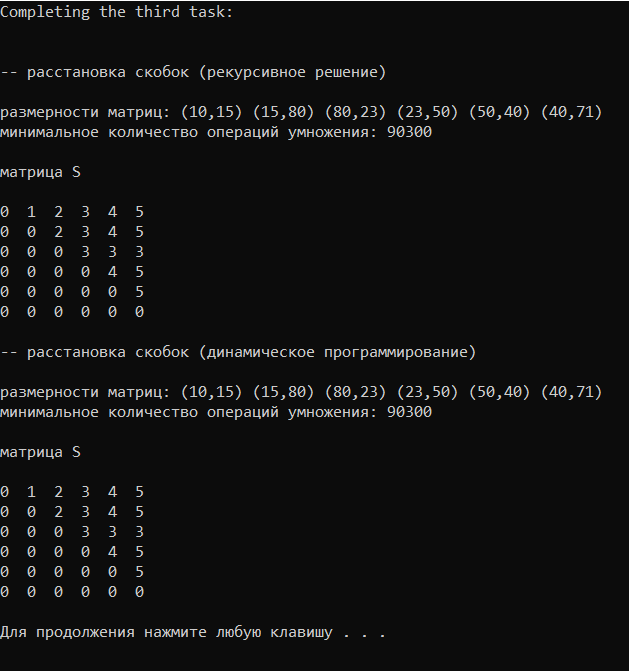


Рисунок 10 – результат выполнения программы

Протестировав программу на работоспособность, а также выполнив сравнительный анализ времени затраченного на решение задачи о расстановке скобок при перемножении матриц для двух методов решения, можно заметить, что динамический алгоритм работает с той же скоростью, что и рекурсивный алгоритм.

**Вывод:** динамический подход к решению задач позволяет выполнять их значительно быстрее, чем рекурсивный, особенно это будет заметно при решении задач с большим объёмом информации. Так же, я сделал следующие выводы:

1. Динамическое программирование - это мощный метод решения задач, позволяющий эффективно решать широкий спектр задач, которые не могут быть решены простыми алгоритмами.

2. Метод динамического программирования заключается в разбиении сложной задачи на более простые подзадачи, решение которых затем комбинируется в общее решение задачи.

3. Решение задач методом динамического программирования отличается высокой скоростью выполнения благодаря использованию кэширования вычислений и быстрой обработке данных.

4. В ходе выполнения лабораторной работы был исследован один из наиболее популярных методов динамического программирования - расстояние Левенштейна. Результаты экспериментов показали, что данная техника может быть очень эффективна для решения задач, связанных с обработкой текстовых данных.

5. Решение задач методом динамического программирования может быть осуществлено не только с помощью программирования на языке C++, но и на других языках, таких как Python или Java.

Учреждение образования

«Белорусский государственный технологический университет»

**Лабораторная работа №5**

Транспортная задача

Выполнил:

Студент 2 курса 4 группы ФИТ

Станчик Максим Андреевич

2024 г.

Вариант: 8.

Исходные данные:

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| ПОТРЕБИТЕЛИ  ПОСТАВЩИКИ | 1 | 2 | 3 | 4 | 5 | 6 | ЗАПАСЫ |
| 1 | **N+12** | **N+2** | **N+6** | **N+3** | **N+11** | **N+1** | **168+N** |
| 2 | **N+10** | **N** | **N+8** | **N+5** | **N+7** | **N+13** | **113+N** |
| 3 | **N+1** | **N+5** | **N+11** | **N+8** | **N+2** | **N+11** | **150+N** |
| 4 | **N+4** | **N+10** | **N+10** | **N+3** | **N+13** | **N+2** | **159+N** |
| 5 | **N+3** | **N+11** | **N+9** | **N** | **N+10** | **N+4** | **100+N** |
| ПОТРЕБНОСТИ | **143+N** | **107+N** | **131+N** | **193+N** | **95+N** | **163+N** |  |

Подставили N и получили таблицу:

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| ПОТРЕБИТЕЛИ  ПОСТАВЩИКИ | 1 | 2 | 3 | 4 | 5 | 6 | ЗАПАСЫ |
| 1 | **20** | **10** | **14** | **11** | **19** | **9** | **176** |
| 2 | **18** | **8** | **16** | **13** | **15** | **21** | **121** |
| 3 | **9** | **13** | **19** | **16** | **10** | **19** | **158** |
| 4 | **12** | **18** | **18** | **11** | **21** | **10** | **167** |
| 5 | **11** | **19** | **17** | **8** | **18** | **12** | **108** |
| ПОТРЕБНОСТИ | **151** | **115** | **139** | **201** | **103** | **171** |  |

**Ход решения:**

* Проверить, открытая задача или закрытая;
* Составить опорный план;
* Применить метод потенциалов;

**Проверка на открытость задачи**

m = 5 – количество поставщиков,

n = 6 – количество потребителей;

∑a = 176 + 121 + 158 + 167 + 108 = 730 -- запасы груза на базах;

∑b = 151 + 115 + 139 + 201 + 103 + 171 = 880 -- суммарная потребность груза в пунктах назначения.

∑b = 880 и ∑a = 730, то есть ∑b > ∑a, следовательно, суммарный объем продукции, которую необходимо отправить не удовлетворяет потребностям пунктов назначения. Исходя из чего можно сделать вывод, что транспортная задача является открытой (с нарушенным балансом).

**Приведение открытой задачи к закрытой**

Чтобы получить закрытую модель, введем дополнительную (фиктивную) базу с запасом груза, равным |730-880| = 150.

Тарифы перевозки единицы груза из базы ко всем потребителям полагаем равны нулю

Введем новые обозначения:

Ai – объем продукции в пункте отправления

Bj – потребность в пункте назначения

Получили таблицу:

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| ПОТРЕБИТЕЛИ  ПОСТАВЩИКИ | B1 | B2 | B3 | B4 | B5 | B6 | ЗАПАСЫ |
| A1 | **20** | **10** | **14** | **11** | **19** | **9** | **176** |
| A2 | **18** | **8** | **16** | **13** | **15** | **21** | **121** |
| A3 | **9** | **13** | **19** | **16** | **10** | **19** | **158** |
| A4 | **12** | **18** | **18** | **11** | **21** | **10** | **167** |
| A5 | **11** | **19** | **17** | **8** | **18** | **12** | **108** |
| A6 | **0** | **0** | **0** | **0** | **0** | **0** | **150** |
| ПОТРЕБНОСТИ | **151** | **115** | **139** | **201** | **103** | **171** |  |

**Решение транспортной задачи**

**Этап 1. Построение начального базисного решения**

Используем метод наименьшей стоимости

Cij – стоимость доставки единицы продукции от i-го поставщика к j-му потребителю

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| ПОТРЕБИТЕЛИ  ПОСТАВЩИКИ | B1 | B2 | B3 | B4 | B5 | B6 | ЗАПАСЫ |
| A1 | **20** | **10** | **14** | **11** | **19** | **9** | **176** |
| A2 | **18** | **8** | **16** | **13** | **15** | **21** | **121** |
| A3 | **9** | **13** | **19** | **16** | **10** | **19** | **158** |
| A4 | **12** | **18** | **18** | **11** | **21** | **10** | **167** |
| A5 | **11** | **19** | **17** | **8** | **18** | **12** | **108** |
| A6 | **0** | **0** | **0** | **0** | **0** | **0** | **150** |
| ПОТРЕБНОСТИ | **151** | **115** | **139** | **201** | **103** | **171** |  |

Суть метода заключается в том, что из всей таблицы стоимостей выбирают наименьшую, и в клетку, которая ей соответствует, помещают меньшее из чисел ai, или bj.  
Затем, из рассмотрения исключают либо строку, соответствующую поставщику, запасы которого полностью израсходованы, либо столбец, соответствующий потребителю, потребности которого полностью удовлетворены, либо и строку и столбец, если израсходованы запасы поставщика и удовлетворены потребности потребителя.  
Из оставшейся части таблицы стоимостей снова выбирают наименьшую стоимость, и процесс распределения запасов продолжают, пока все запасы не будут распределены, а потребности удовлетворены.

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| ПОТРЕБИТЕЛИ  ПОСТАВЩИКИ | B1 | B2 | B3 | B4 | B5 | B6 | ЗАПАСЫ |
| A1 | **20** | **10** | **14** | **11** | **19** | **9** | **176** |
| A2 | **18** | **8** | **16** | **13** | **15** | **21** | **121** |
| A3 | **9** | **13** | **19** | **16** | **10** | **19** | **158** |
| A4 | **12** | **18** | **18** | **11** | **21** | **10** | **167** |
| A5 | **11** | **19** | **17** | **8** | **18** | **12** | **108** |
| A6 | **0** | **0** | **0** | **0** | **0** | **0** | **150** |
| ПОТРЕБНОСТИ | **151** | **115** | **139** | **201** | **103** | **171** |  |

Искомый элемент равен C22=8. Для этого элемента запасы равны 121, потребности 115. Поскольку минимальным является 115, то вычитаем его.  
x54 = min(115,121) = 115

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| ПОТРЕБИТЕЛИ  ПОСТАВЩИКИ | B1 | B2 | B3 | B4 | B5 | B6 | ЗАПАСЫ |
| A1 | **20** | **10** | **14** | **11** | **19** | **9** | **176** |
| A2 | **18** | **8|115** | **16** | **13** | **15** | **21** | **6** |
| A3 | **9** | **13** | **19** | **16** | **10** | **19** | **158** |
| A4 | **12** | **18** | **18** | **11** | **21** | **10** | **167** |
| A5 | **11** | **19** | **17** | **8** | **18** | **12** | **108** |
| A6 | **0** | **0** | **0** | **0** | **0** | **0** | **150** |
| ПОТРЕБНОСТИ | **151** | **0** | **139** | **201** | **103** | **171** |  |

Искомый элемент равен C54=8. Для этого элемента запасы равны 108, потребности 201. Поскольку минимальным является 108, то вычитаем его.  
x54 = min(108, 201) = 108

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| ПОТРЕБИТЕЛИ  ПОСТАВЩИКИ | B1 | B2 | B3 | B4 | B5 | B6 | ЗАПАСЫ |
| A1 | **20** | **10** | **14** | **11** | **19** | **9** | **176** |
| A2 | **18** | **8|115** | **16** | **13** | **15** | **21** | **6** |
| A3 | **9** | **13** | **19** | **16** | **10** | **19** | **158** |
| A4 | **12** | **18** | **18** | **11** | **21** | **10** | **167** |
| A5 | **11** | **19** | **17** | **8|108** | **18** | **12** | **0** |
| A6 | **0** | **0** | **0** | **0** | **0** | **0** | **150** |
| ПОТРЕБНОСТИ | **151** | **0** | **139** | **93** | **103** | **171** |  |

Искомый элемент равен C16=9. Для этого элемента запасы равны 176, потребности 171. Поскольку минимальным является 171, то вычитаем его.  
x54 = min(176, 171) = 171

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| ПОТРЕБИТЕЛИ  ПОСТАВЩИКИ | B1 | B2 | B3 | B4 | B5 | B6 | ЗАПАСЫ |
| A1 | **20** | **10** | **14** | **11** | **19** | **9|171** | **5** |
| A2 | **18** | **8|115** | **16** | **13** | **15** | **21** | **6** |
| A3 | **9** | **13** | **19** | **16** | **10** | **19** | **158** |
| A4 | **12** | **18** | **18** | **11** | **21** | **10** | **167** |
| A5 | **11** | **19** | **17** | **8|108** | **18** | **12** | **0** |
| A6 | **0** | **0** | **0** | **0** | **0** | **0** | **150** |
| ПОТРЕБНОСТИ | **151** | **0** | **139** | **93** | **103** | **0** |  |

Искомый элемент равен C31=9. Для этого элемента запасы равны 158, потребности 151. Поскольку минимальным является 151, то вычитаем его.  
x54 = min(158, 151) = 151

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| ПОТРЕБИТЕЛИ  ПОСТАВЩИКИ | B1 | B2 | B3 | B4 | B5 | B6 | ЗАПАСЫ |
| A1 | **20** | **10** | **14** | **11** | **19** | **9|171** | **5** |
| A2 | **18** | **8|115** | **16** | **13** | **15** | **21** | **6** |
| A3 | **9|151** | **13** | **19** | **16** | **10** | **19** | **7** |
| A4 | **12** | **18** | **18** | **11** | **21** | **10** | **167** |
| A5 | **11** | **19** | **17** | **8|108** | **18** | **12** | **0** |
| A6 | **0** | **0** | **0** | **0** | **0** | **0** | **150** |
| ПОТРЕБНОСТИ | **0** | **0** | **139** | **93** | **103** | **0** |  |

Искомый элемент равен C35=10. Для этого элемента запасы равны 7, потребности 103. Поскольку минимальным является 7, то вычитаем его.  
x54 = min(7, 103) = 7

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| ПОТРЕБИТЕЛИ  ПОСТАВЩИКИ | B1 | B2 | B3 | B4 | B5 | B6 | ЗАПАСЫ |
| A1 | **20** | **10** | **14** | **11** | **19** | **9|171** | **5** |
| A2 | **18** | **8|115** | **16** | **13** | **15** | **21** | **6** |
| A3 | **9|151** | **13** | **19** | **16** | **10|7** | **19** | **0** |
| A4 | **12** | **18** | **18** | **11** | **21** | **10** | **167** |
| A5 | **11** | **19** | **17** | **8|108** | **18** | **12** | **0** |
| A6 | **0** | **0** | **0** | **0** | **0** | **0** | **150** |
| ПОТРЕБНОСТИ | **0** | **0** | **139** | **93** | **96** | **0** |  |

Искомый элемент равен C14=11. Для этого элемента запасы равны 5, потребности 93. Поскольку минимальным является 5, то вычитаем его.  
x54 = min(5, 93) = 7

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| ПОТРЕБИТЕЛИ  ПОСТАВЩИКИ | B1 | B2 | B3 | B4 | B5 | B6 | ЗАПАСЫ |
| A1 | **20** | **10** | **14** | **11|5** | **19** | **9|171** | **0** |
| A2 | **18** | **8|115** | **16** | **13** | **15** | **21** | **6** |
| A3 | **9|151** | **13** | **19** | **16** | **10|7** | **19** | **0** |
| A4 | **12** | **18** | **18** | **11** | **21** | **10** | **167** |
| A5 | **11** | **19** | **17** | **8|108** | **18** | **12** | **0** |
| A6 | **0** | **0** | **0** | **0** | **0** | **0** | **150** |
| ПОТРЕБНОСТИ | **0** | **0** | **139** | **88** | **96** | **0** |  |

Искомый элемент равен C44=11. Для этого элемента запасы равны 167, потребности 88. Поскольку минимальным является 88, то вычитаем его.  
x54 = min(167, 88) = 88

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| ПОТРЕБИТЕЛИ  ПОСТАВЩИКИ | B1 | B2 | B3 | B4 | B5 | B6 | ЗАПАСЫ |
| A1 | **20** | **10** | **14** | **11|5** | **19** | **9|171** | **0** |
| A2 | **18** | **8|115** | **16** | **13** | **15** | **21** | **6** |
| A3 | **9|151** | **13** | **19** | **16** | **10|7** | **19** | **0** |
| A4 | **12** | **18** | **18** | **11|88** | **21** | **10** | **79** |
| A5 | **11** | **19** | **17** | **8|108** | **18** | **12** | **0** |
| A6 | **0** | **0** | **0** | **0** | **0** | **0** | **150** |
| ПОТРЕБНОСТИ | **0** | **0** | **139** | **0** | **96** | **0** |  |

Искомый элемент равен C25=15. Для этого элемента запасы равны 6, потребности 96. Поскольку минимальным является 88, то вычитаем его.  
x54 = min(6, 96) = 6

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| ПОТРЕБИТЕЛИ  ПОСТАВЩИКИ | B1 | B2 | B3 | B4 | B5 | B6 | ЗАПАСЫ |
| A1 | **20** | **10** | **14** | **11|5** | **19** | **9|171** | **0** |
| A2 | **18** | **8|115** | **16** | **13** | **15|6** | **21** | **0** |
| A3 | **9|151** | **13** | **19** | **16** | **10|7** | **19** | **0** |
| A4 | **12** | **18** | **18** | **11|88** | **21** | **10** | **79** |
| A5 | **11** | **19** | **17** | **8|108** | **18** | **12** | **0** |
| A6 | **0** | **0** | **0** | **0** | **0** | **0** | **150** |
| ПОТРЕБНОСТИ | **0** | **0** | **139** | **0** | **90** | **0** |  |

Искомый элемент равен C43=18. Для этого элемента запасы равны 79, потребности 139. Поскольку минимальным является 79, то вычитаем его.  
x54 = min(79, 139) = 79

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| ПОТРЕБИТЕЛИ  ПОСТАВЩИКИ | B1 | B2 | B3 | B4 | B5 | B6 | ЗАПАСЫ |
| A1 | **20** | **10** | **14** | **11|5** | **19** | **9|171** | **0** |
| A2 | **18** | **8|115** | **16** | **13** | **15|6** | **21** | **0** |
| A3 | **9|151** | **13** | **19** | **16** | **10|7** | **19** | **0** |
| A4 | **12** | **18** | **18|79** | **11|88** | **21** | **10** | **0** |
| A5 | **11** | **19** | **17** | **8|108** | **18** | **12** | **0** |
| A6 | **0** | **0** | **0** | **0** | **0** | **0** | **150** |
| ПОТРЕБНОСТИ | **0** | **0** | **60** | **0** | **90** | **0** |  |

Искомый элемент равен C63=0. Для этого элемента запасы равны 150, потребности 60. Поскольку минимальным является 60, то вычитаем его.  
x54 = min(60, 150) = 60

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| ПОТРЕБИТЕЛИ  ПОСТАВЩИКИ | B1 | B2 | B3 | B4 | B5 | B6 | ЗАПАСЫ |
| A1 | **20** | **10** | **14** | **11|5** | **19** | **9|171** | **0** |
| A2 | **18** | **8|115** | **16** | **13** | **15|6** | **21** | **0** |
| A3 | **9|151** | **13** | **19** | **16** | **10|7** | **19** | **0** |
| A4 | **12** | **18** | **18|79** | **11|88** | **21** | **10** | **0** |
| A5 | **11** | **19** | **17** | **8|108** | **18** | **12** | **0** |
| A6 | **0** | **0** | **0|60** | **0** | **0** | **0** | **90** |
| ПОТРЕБНОСТИ | **0** | **0** | **0** | **0** | **90** | **0** |  |

Искомый элемент равен C65=0. Для этого элемента запасы равны 90, потребности 90. Запасы и потребности равны. Вычитаем 90.

x54 = min(90, 90) = 60

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| ПОТРЕБИТЕЛИ  ПОСТАВЩИКИ | B1 | B2 | B3 | B4 | B5 | B6 | ЗАПАСЫ |
| A1 | **20** | **10** | **14** | **11|5** | **19** | **9|171** | **0** |
| A2 | **18** | **8|115** | **16** | **13** | **15|6** | **21** | **0** |
| A3 | **9|151** | **13** | **19** | **16** | **10|7** | **19** | **0** |
| A4 | **12** | **18** | **18|79** | **11|88** | **21** | **10** | **0** |
| A5 | **11** | **19** | **17** | **8|108** | **18** | **12** | **0** |
| A6 | **0** | **0** | **0|60** | **0** | **0|90** | **0** | **0** |
| ПОТРЕБНОСТИ | **0** | **0** | **0** | **0** | **0** | **0** |  |

Произведем проверку. Если количество выделенных ячеек (базисных значений) равно n + m - 1 то задача решена верно.

n + m - 1 = 6 + 6 - 1 = 11 и все потребности равны 0.

Следовательно, опорный план является невырожденным.

В результате получен первый опорный план, который является допустимым, так как потребность потребителей удовлетворена, все запасы истощены, а план соответствует системе ограничений транспортной задачи.

C31 = 151

C22 = 115

C43 = 79

C63 = 60

C14 = 5

C44 = 88

C54 = 108

C25 = 6

C35 = 7

C65 = 90

C16 = 171

Значение целевой функции для этого опорного плана:

F(x) = (151 \* 9) + (115 \* 8) + (79 \* 18) + (60 \* 0) + (5 \* 11) + (88 \* 11) + (108 \* 8) + (6 \* 15) + (7 \* 10) + (90 \* 0) + (171 \* 9) = 7287

**Применение метода потенциалов**

Проверим оптимальность опорного плана. Найдем *предварительные потенциалы* ui, vj. по занятым клеткам таблицы, в которых ui + vj = cij, полагая, что U1 = 0.

Ui – поставщики

Vj – потребители

Определяем потенциалы для всех базисных переменных

U3 + V1 = 9 => V1 = 17

U2 + V2 = 8 => V2 = 11

U4 + V3 = 18 => V3 = 18

U6 + V3 = 0 => U6 = -18

U1 + V4 = 11 => V4 = 11

U4 + V4 = 11 => U4 = 0

U5 + V4 = 8 => U5 = -3

U2 + V5 = 15 => U2 = -3

U3 + V5 = 10 => U3 = -8

U6 + V5 = 0 => V5 = 18

U1 + V6 = 9 => V6 = 9

Сумма потенциалов:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | V1=17 | V2=11 | V3=18 | V4=11 | V5=18 | V6=9 |
| U1=0 | **17** | **11** | **18** | **11** | **18** | **9** |
| U2=-3 | **14** | **8** | **15** | **8** | **15** | **6** |
| U3=-8 | **9** | **3** | **10** | **3** | **10** | **1** |
| U4=0 | **17** | **11** | **18** | **11** | **18** | **9** |
| U5=-3 | **14** | **8** | **15** | **8** | **16** | **6** |
| U6=-18 | **-1** | **-7** | **0** | **-7** | **0** | **-9** |

Чтобы решение было оптимальным нужно чтобы разность суммы потенциалов Ui+Vi и стоимость доставки Сi,j была равна нулю и меньше.

Xij = Ui + Vj – Cij.

//нужно три цифры

|  |  |  |
| --- | --- | --- |
| Небазисные переменные | Расчет | Результат |
| X11 | 17-20 | -3 |
| X12 | 11-10 | 1 |
| X13 | 18-14 | 4 |
| X14 | 11-11 | 0 |
| X15 | 18-19 | -1 |
| X16 | 9-9 | 0 |
| X21 | 14-18 | -4 |
| X22 | 8-8 | 0 |
| X23 | 15-16 | -1 |
| X24 | 8-13 | -5 |
| X25 | 15-15 | 0 |
| X26 | 6-21 | -15 |
| X31 | 9-9 | 0 |
| X32 | 3-13 | -10 |
| X33 | 10-19 | -9 |
| X34 | 3-16 | -13 |
| X35 | 10-10 | 0 |
| X36 | 1-19 | -18 |
| X41 | 17-12 | 5 |
| X42 | 11-18 | -7 |
| X43 | 18-18 | 0 |
| X44 | 11-11 | 0 |
| X45 | 18-21 | -3 |
| X46 | 9-10 | -1 |
| X51 | 14-11 | 3 |
| X52 | 8-19 | -11 |
| X53 | 15-17 | -2 |
| X54 | 8-8 | 0 |
| X55 | 16-18 | -2 |
| X56 | 6-12 | -6 |
| X61 | 0-1 | -1 |
| X62 | 0-7 | -7 |
| X63 | 0-0 | 0 |
| X64 | 0-7 | -7 |
| X65 | 0-0 | 0 |
| X66 | 0-9 | -9 |

Матрица сумм потенциалов

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | **1** | **2** | **3** | **4** | **5** | **6** |
| **1** | **-3** | **1** | **4** | **0** | **-1** | **0** |
| **2** | **-4** | **0** | **-1** | **-5** | **0** | **-15** |
| **3** | **0** | **-10** | **-9** | **-13** | **0** | **-18** |
| **4** | **5** | **-7** | **0** | **0** | **-3** | **-1** |
| **5** | **3** | **-11** | **-2** | **0** | **-2** | **-6** |
| **6** | **-1** | **-7** | **0** | **-7** | **0** | **-9** |

Опорный план не является оптимальным, так как существуют оценки свободных клеток, для которых Ui + Vj > Cij.

max(1,4,5,3) = 5.

Выбираем максимальную оценку свободной клетки (4;1).

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| ПОТРЕБИТЕЛИ  ПОСТАВЩИКИ | B1 | B2 | B3 | B4 | B5 | B6 | ЗАПАСЫ |
| A1 | **20** | **10** | **14** | **11|5** | **19** | **9|171** | **176** |
| A2 | **18** | **8|115** | **16** | **13** | **15|6** | **21** | **121** |
| A3 | **9|72 [-]** | **13** | **19** | **16** | **10|86 [+]** | **19** | **158** |
| A4 | **12|79 [+]** | **18** | **18|0**  **[-]** | **11|88** | **21** | **10** | **167** |
| A5 | **11** | **19** | **17** | **8|108** | **18** | **12** | **108** |
| A6 | **0** | **0** | **0|139[+]** | **0** | **0|11 [-]** | **0** | **150** |
| ПОТРЕБНОСТИ | **151** | **115** | **139** | **201** | **103** | **171** |  |

Получился цикл: 41—43—63—65—35—31—41.

A4B3 больше не является базисом. A4B1 теперь базис.

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| ПОТРЕБИТЕЛИ  ПОСТАВЩИКИ | B1 | B2 | B3 | B4 | B5 | B6 | ЗАПАСЫ |
| A1 | **20** | **10** | **14** | **11|5** | **19** | **9|171** | **176** |
| A2 | **18** | **8|115** | **16** | **13** | **15|6** | **21** | **121** |
| A3 | **9|72** | **13** | **19** | **16** | **10|86** | **19** | **158** |
| A4 | **12|79** | **18** | **18** | **11|88** | **21** | **10** | **167** |
| A5 | **11** | **19** | **17** | **8|108** | **18** | **12** | **108** |
| A6 | **0** | **0** | **0|139** | **0** | **0|11** | **0** | **150** |
| ПОТРЕБНОСТИ | **151** | **115** | **139** | **201** | **103** | **171** |  |

C31 = 72

C41 = 79

C22 = 115

C63 = 139

C14 = 5

C44 = 88

C54 = 108

C25 = 6

C35 = 86

C65 = 11

C16 = 171

Проверим оптимальность опорного плана. Найдем *предварительные потенциалы* ui, vj. по занятым клеткам таблицы, в которых ui + vj = cij, полагая, что U1 = 0.

U3 + V1 = 9 => U3 = -3

U4 + V1 = 12 => V1 = 12

U2 + V2 = 8 => V2 = 6

U6 + V3 = 0 => V3 = 13

U1 + V4 =11 => V4 = 11

U4 + V4 =11 => U4 = 0

U5 + V4 = 8 => U5 = -3

U2 + V5 = 15 => U2 = 2

U3 + V5 = 10 => V5 = 13

U6 + V5 = 0 => U6 = -13

U1 + V6 = 9 => V6 = 9

Сумма потенциалов:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | V1 = 12 | V2 = 6 | V3 = 13 | V4 = 11 | V5 = 13 | V6 = 9 |
| U1 = 0 | **12** | **6** | **13** | **11** | **13** | **9** |
| U2 = 2 | **14** | **8** | **15** | **13** | **15** | **11** |
| U3 = -3 | **9** | **3** | **10** | **8** | **10** | **6** |
| U4 = 0 | **12** | **6** | **13** | **11** | **13** | **9** |
| U5 = -3 | **9** | **3** | **10** | **8** | **10** | **6** |
| U6 = -13 | **-1** | **-7** | **0** | **-2** | **0** | **-4** |

Чтобы решение было оптимальным нужно чтобы разность суммы потенциалов Ui+Vi и стоимость доставки Сi,j была равна нулю и меньше.

Xij = Ui + Vj – Cij.

|  |  |  |
| --- | --- | --- |
| Небазисные переменные | Расчет | Результат |
| X11 | 12 – 20 | -8 |
| X12 | 6 – 10 | -4 |
| X13 | 13 – 14 | -1 |
| X14 | 11 – 11 | 0 |
| X15 | 13 – 19 | -6 |
| X16 | 9 – 9 | 0 |
| X21 | 14 – 18 | -4 |
| X22 | 8 - 8 | 0 |
| X23 | 15 – 16 | -1 |
| X24 | 13 - 13 | 0 |
| X25 | 15 - 15 | 0 |
| X26 | 11 - 21 | -10 |
| X31 | 9 – 9 | 0 |
| X32 | 3 – 13 | -10 |
| X33 | 10 – 19 | -9 |
| X34 | 8 – 16 | -8 |
| X35 | 10 – 10 | 0 |
| X36 | 6 – 19 | -13 |
| X41 | 12 - 12 | 0 |
| X42 | 6 – 18 | -12 |
| X43 | 13 - 18 | -5 |
| X44 | 11 - 11 | 0 |
| X45 | 13 – 21 | -8 |
| X46 | 9 - 10 | -1 |
| X51 | 9 - 11 | -2 |
| X52 | 3 – 19 | -16 |
| X53 | 10 – 17 | -7 |
| X54 | 8 – 8 | 0 |
| X55 | 10 – 18 | -8 |
| X56 | 6 – 12 | -6 |
| X61 | -1 - 0 | -1 |
| X62 | -7 – 0 | -7 |
| X63 | 0 – 0 | 0 |
| X64 | -2 – 0 | -2 |
| X65 | 0 – 0 | 0 |
| X66 | -4 – 0 | -4 |

Матрица сумм потенциалов

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | **1** | **2** | **3** | **4** | **5** | **6** |
| **1** | **-8** | **-4** | **-1** | **0** | **-6** | **0** |
| **2** | **-4** | **0** | **-1** | **0** | **0** | **-10** |
| **3** | **0** | **-10** | **-9** | **-8** | **0** | **-13** |
| **4** | **0** | **-12** | **-5** | **0** | **-8** | **-1** |
| **5** | **-2** | **-16** | **-7** | **0** | **-8** | **-6** |
| **6** | **-1** | **-7** | **0** | **-2** | **0** | **-4** |

Опорный план является оптимальным, так все оценки свободных клеток удовлетворяют условию Ui + Vj ≤ Cij.

Минимальные затраты составят: F(x) = (72 \* 9) + (79 \* 12) + (115 \* 8) + (139 \* 0) + (11 \* 5) + (11 \* 88) + (8 \* 108) + (15 \* 6) + (10 \* 86) + (0 \* 11) + (9 \* 171) = 6892.

**Вывод:** в данной лабораторной работе была изучена и решена транспортная задача, одна из классических задач линейного программирования. Главная цель этой задачи заключается в оптимальном распределении ресурсов между источниками и потребителями с учетом минимизации затрат или максимизации прибыли. В результате выполнения лабораторной работы мы получили оптимальное распределение ресурсов между источниками и потребителями, которое достигает минимальных затрат или максимальной прибыли, в зависимости от поставленной задачи.

**Лабораторная работа №6**

Алгоритмы на графах

Выполнил:

Студент 2 курса 4 группы ФИТ

Станчик Максим Андреевич

2024 г.

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

***Задание 1.*** Ориентированный граф **G** взять в соответствии с вариантом. Представить его в отчете в виде матрицы смежности, матрицы инцидентности, списка смежных вершин.

|  |  |
| --- | --- |
| 8 |  |

**Матрица смежности** - это квадратная матрица размера n x n, где n - число вершин в графе. Эта матрица используется для представления графа в виде таблицы, где каждый элемент матрицы a\_ij показывает, существует ли ребро между вершинами i и j.

Всего 7 вершин, следовательно, n = 7.

**Матрица смежности:**

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

**Матрица инцидентности** — одна из форм представления графа, в которой указываются связи между инцидентными элементами графа (ребро(дуга) и вершина). Столбцы матрицы соответствуют ребрам, строки — вершинам. Ненулевое значение в ячейке матрицы указывает связь между вершиной и ребром (их инцидентность).



|  |  |
| --- | --- |
| **Ребра** | **Вершины** |
| **0** | **0-2** |
| **1** | **2-3** |
| **2** | **2-5** |
| **3** | **5-6** |
| **4** | **3-5** |
| **5** | **3-1** |
| **6** | **3-4** |
| **7** | **3-6** |
| **8** | **4-6** |
| **9** | **4-1** |

**Матрица инцидентности:**

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | **0** | **1** | **2** | **3** | **4** | **5** | **6** | **7** | **8** | **9** |
| **0** | **-1** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** |
| **1** | **0** | **0** | **0** | **0** | **0** | **1** | **0** | **0** | **0** | **1** |
| **2** | **1** | **-1** | **-1** | **0** | **0** | **0** | **0** | **0** | **0** | **0** |
| **3** | **0** | **1** | **0** | **0** | **-1** | **-1** | **-1** | **-1** | **0** | **0** |
| **4** | **0** | **0** | **0** | **0** | **0** | **0** | **1** | **0** | **-1** | **-1** |
| **5** | **0** | **0** | **1** | **-1** | **1** | **0** | **0** | **0** | **0** | **0** |
| **6** | **0** | **0** | **0** | **1** | **0** | **0** | **0** | **1** | **1** | **0** |

**Список смежных вершин:**

S0 = {1, 2}

S1 = {3, 4}

S2 = ∅

S3 = {5, 6}

S4 = {3}

S5 = {6}

S6 = {2}

***Задание 2.*** 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

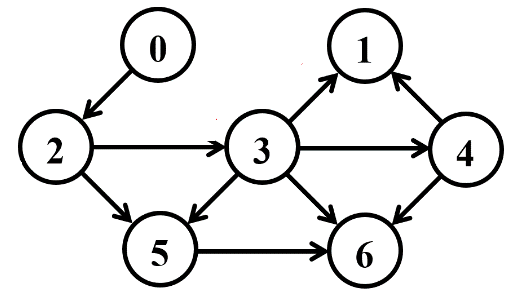
**Поиск в ширину:**

***Черный цвет – посещенные.***

***Белый – непосещённые.***

***Темно серый – текущая вершина.***

***Светло серый – на очереди.***

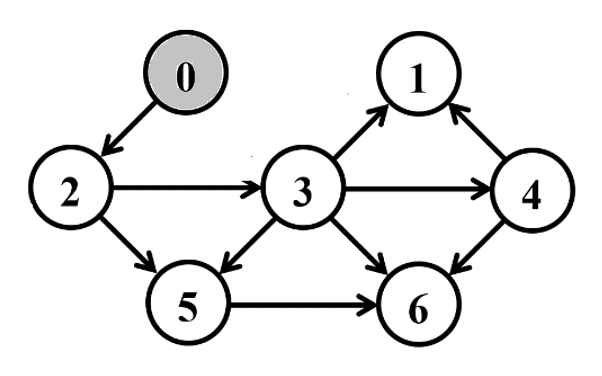


Шаг: 1.

Посещённые вершины: {}

Очередь: {0}

Текущая вершина: {}

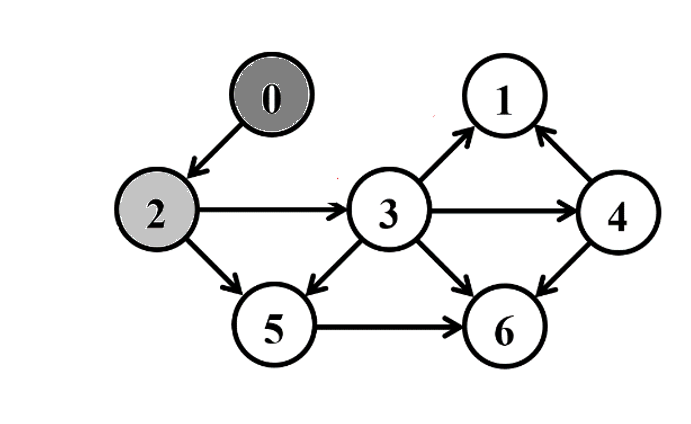


Шаг: 2.

Посещённые вершины: {0}

Очередь: {2}

Текущая вершина: {0}

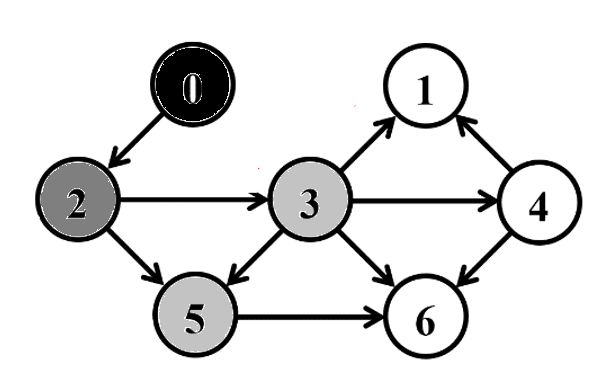


Шаг: 3.

Посещённые вершины: {0}

Очередь: {3, 5}

Текущая вершина: {2}

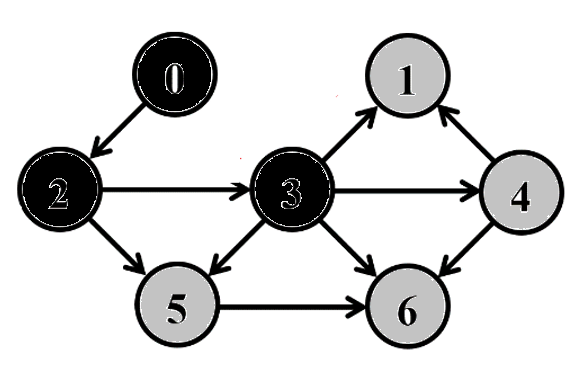


Шаг: 4.

Посещённые вершины: {0, 2}

Очередь: {5, 1, 4, 6}

Текущая вершина: {3}

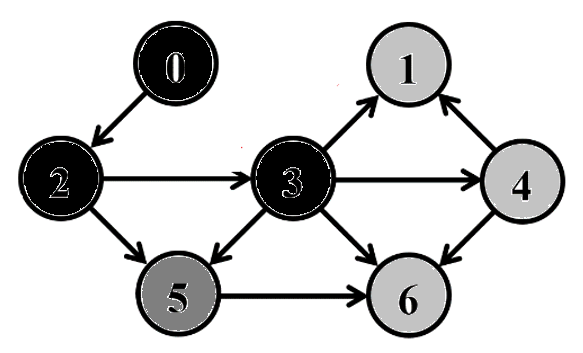


Шаг: 5.

Посещённые вершины: {0, 2, 3}

Очередь: {1, 4, 6}

Текущая вершина: {5}

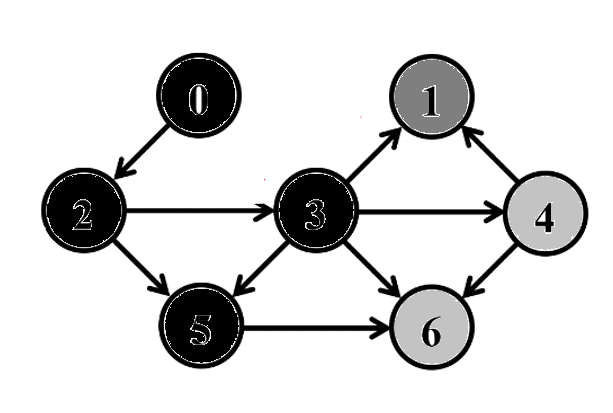


Шаг: 6.

Посещённые вершины: {0, 2, 3, 5}

Очередь: {4, 6}

Текущая вершина: {1}

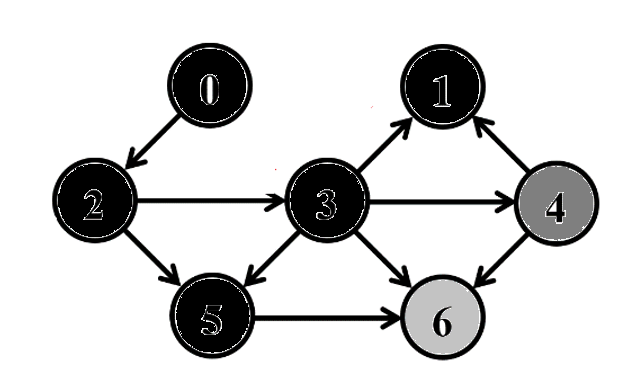


Шаг: 7.

Посещённые вершины: {0, 2, 3, 5, 1}

Очередь: {6}

Текущая вершина: {4}

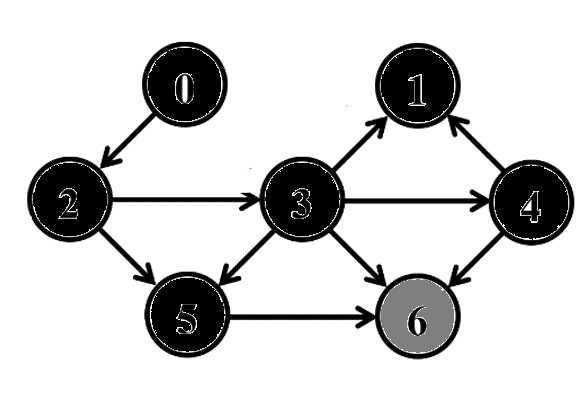


Шаг: 8.

Посещённые вершины: {0, 2, 3, 5, 1, 4}

Очередь: {}

Текущая вершина: {6}

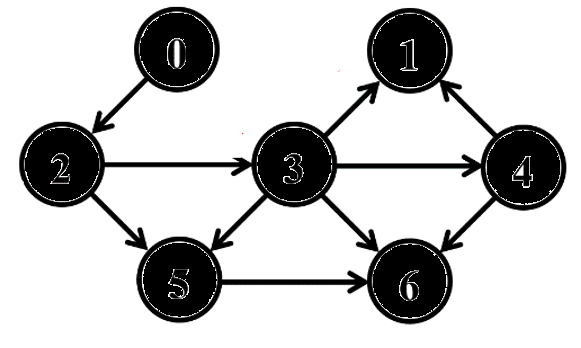


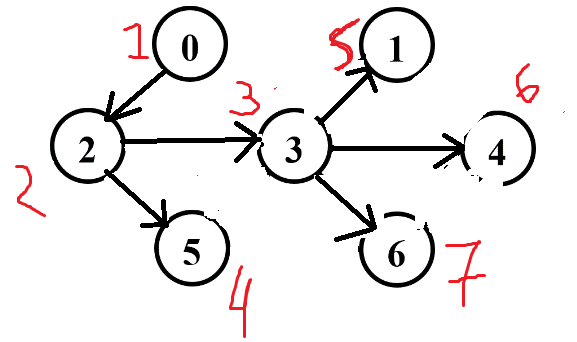
Шаг: 9.

Посещённые вершины: {0, 2, 3, 5, 1, 4, 6}

Очередь: {}

Текущая вершина: {}

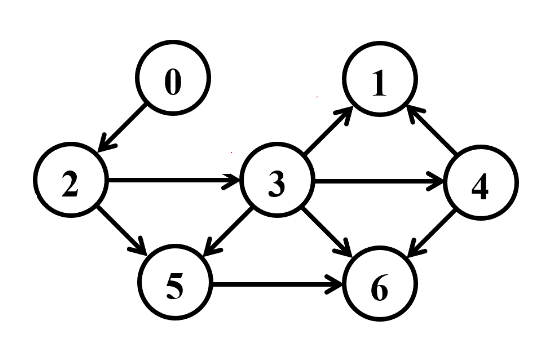




Итоговая последовательность: 0-2-3-5-1-4-6.

**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:**

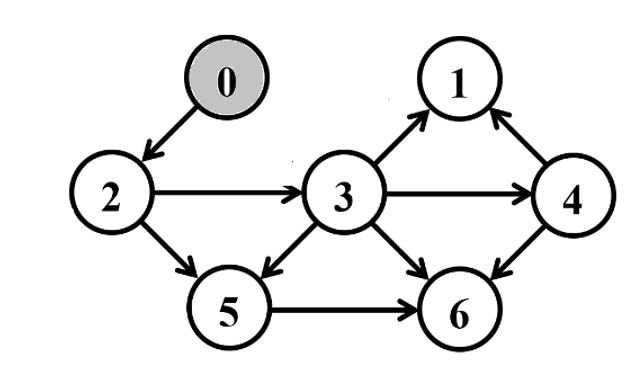


Шаг 1

Посещенные вершины: {}

Стек: {0}

Текущая вершина: {}

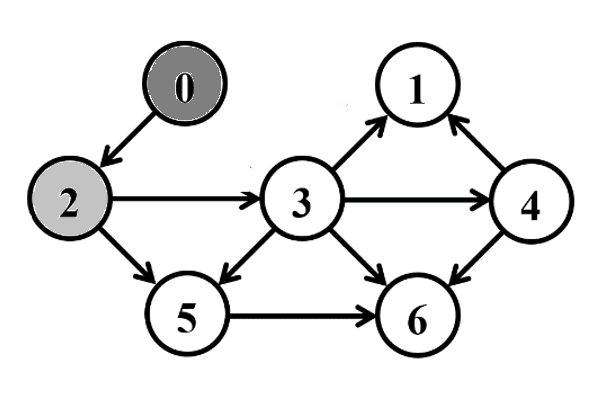


Шаг 2

Посещенные вершины: {}

Стек: {2}

Текущая вершина: {0}

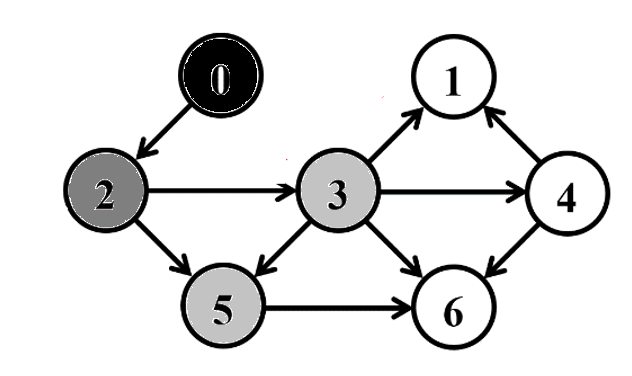


Шаг 3

Посещенные вершины: {0}

Стек: {3, 5}

Текущая вершина: {2}



Шаг 4

Посещенные вершины: {0, 2}

Стек: {1}

Текущая вершина: {3}

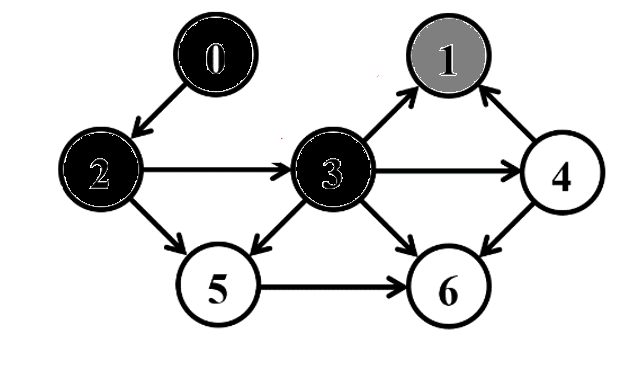


Шаг 5

Посещенные вершины: {0, 2, 3}

Стек: {}

Текущая вершина: {1}

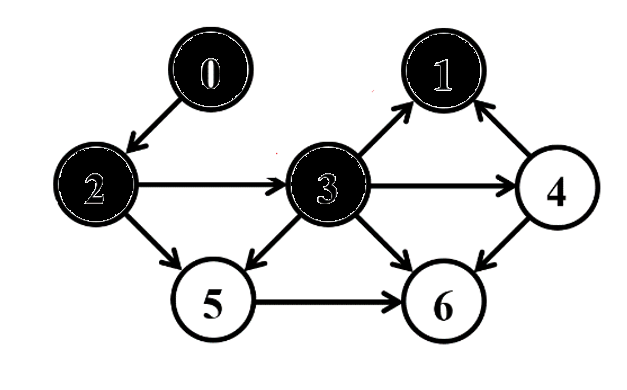


Шаг 6

Посещенные вершины: {0, 2, 3, 1}

Стек: {}

Текущая вершина: {}

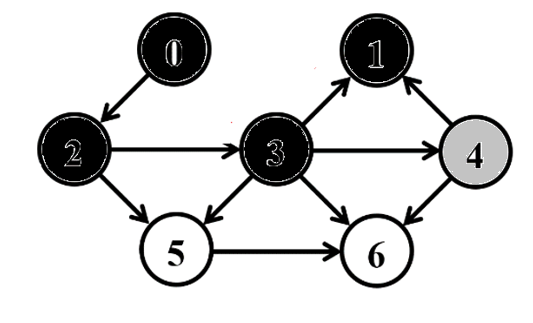


Шаг 7

Посещенные вершины: {0, 2, 3, 1}

Стек: {4}

Текущая вершина: {3}

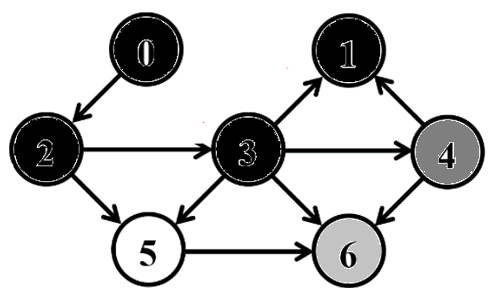


Шаг 8

Посещенные вершины: {0, 2, 3, 1}

Стек: {6}

Текущая вершина: {4}

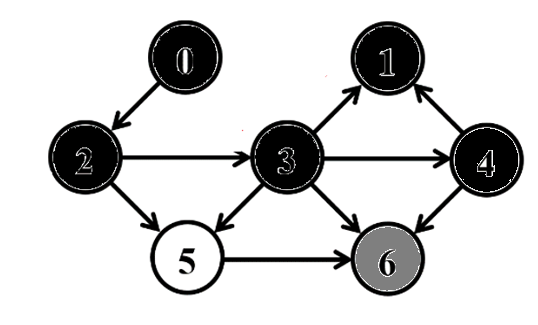


Шаг 8

Посещенные вершины: {0, 2, 3, 1, 4}

Стек: {}

Текущая вершина: {6}

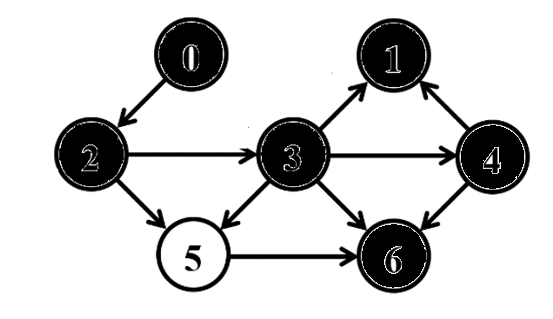


Шаг 9

Посещенные вершины: {0, 2, 3, 1, 4, 6}

Стек: {}

Текущая вершина: {}

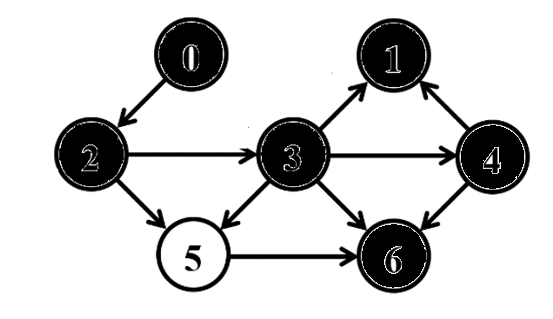


Шаг 10

Посещенные вершины: {0, 2, 3, 1, 4, 6}

Стек: {}

Текущая вершина: {6}

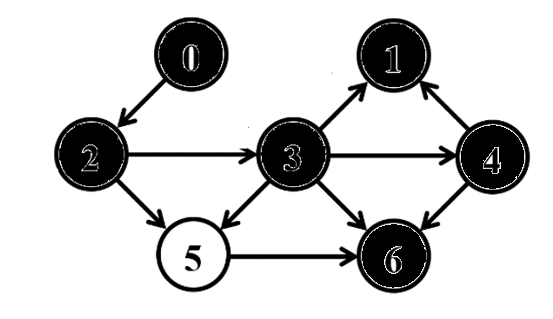


Шаг 11

Посещенные вершины: {0, 2, 3, 1, 4, 6}

Стек: {}

Текущая вершина: {4}

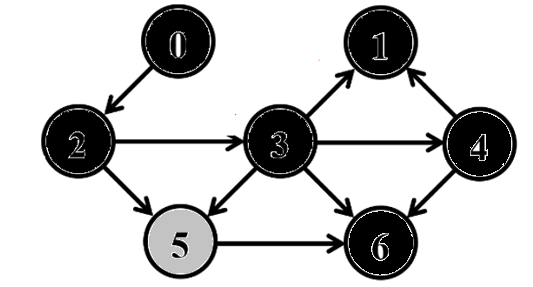


Шаг 12

Посещенные вершины: {0, 2, 3, 1, 4, 6}

Стек: {5}

Текущая вершина: {3}

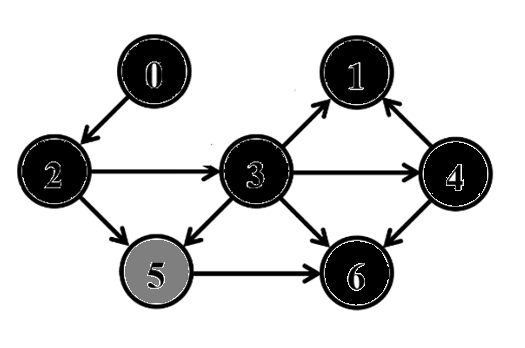


Шаг 13

Посещенные вершины: {0, 2, 3, 1, 4, 6}

Стек: {}

Текущая вершина: {5}

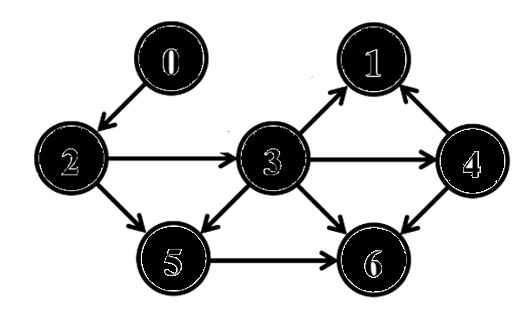


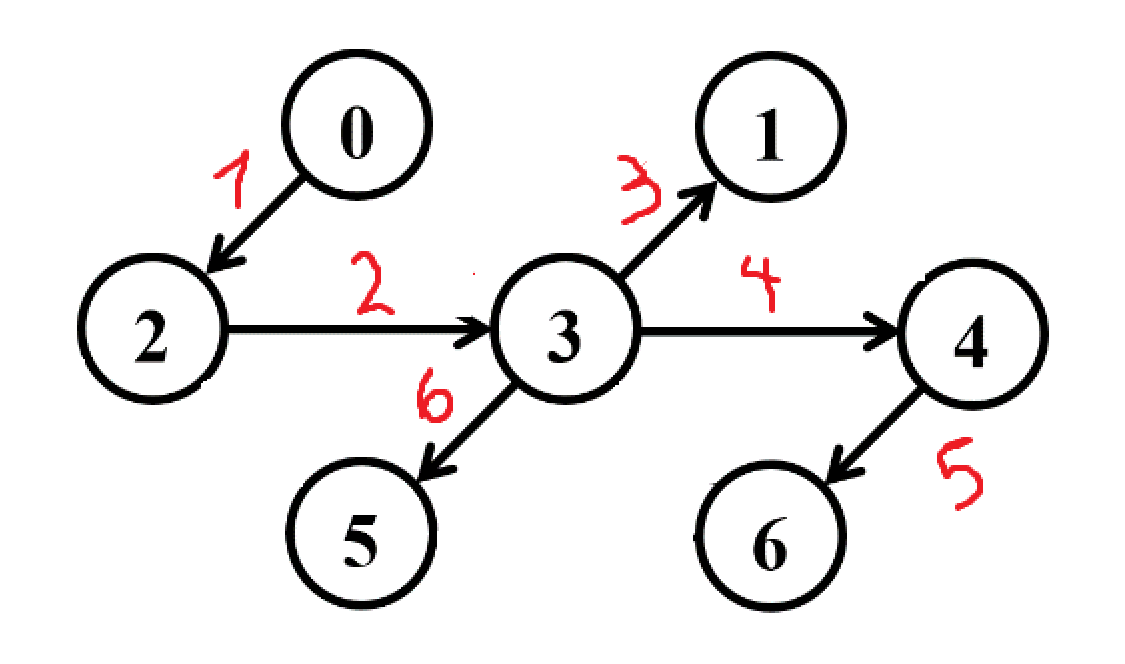
Шаг 14

Посещенные вершины: {0, 2, 3, 1, 4, 6, 5}

Стек: {}

Текущая вершина: {}

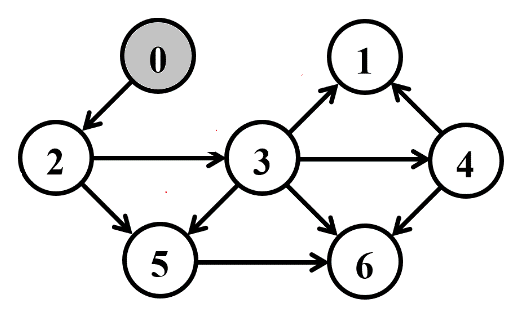


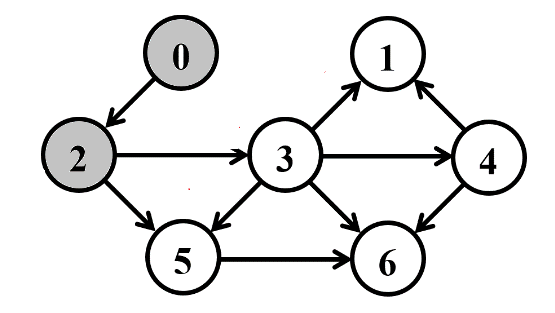


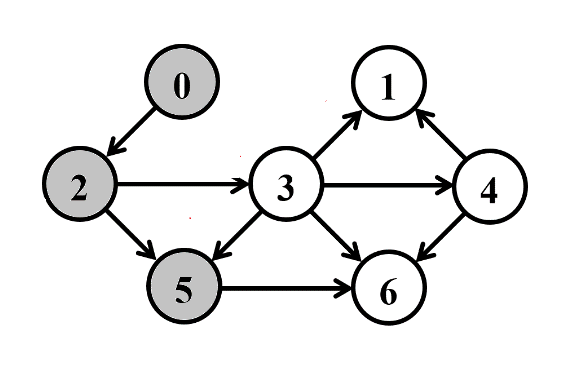
Итоговая последовательность: 0-2-3-1-4-6-5.

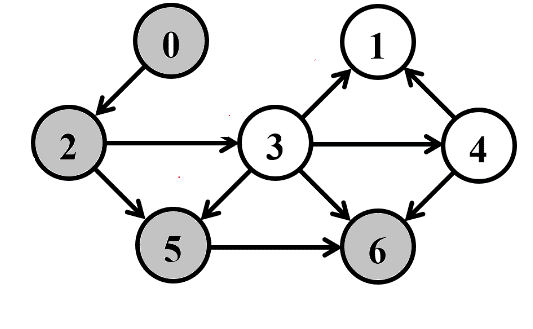
**Топологическая сортировка**

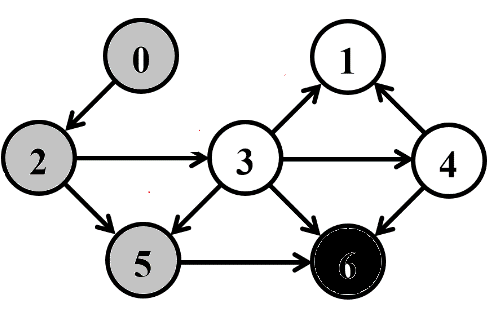
**Топологическая сортировка −** это процедура упорядочивания вершин бесконтурного ориентированного графа, не имеющего циклов (ациклического графа). В результате топологической сортировки для вершин графа определяется такой порядок, что если их расположить на рисунке в соответствии с этим порядком сверху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

****

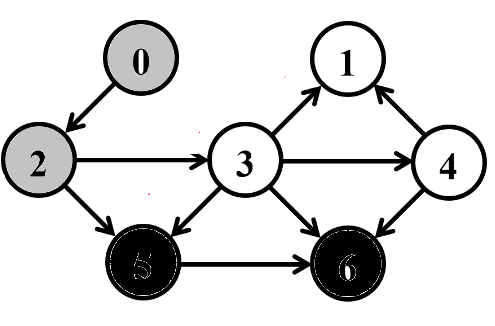
****

****

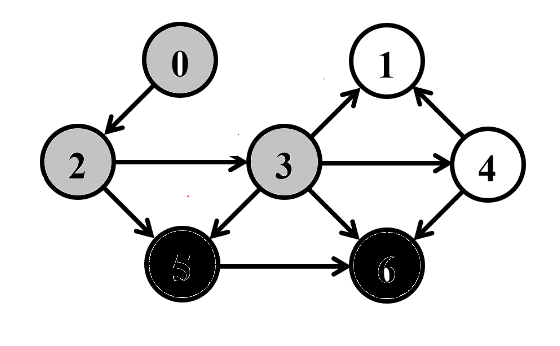
****

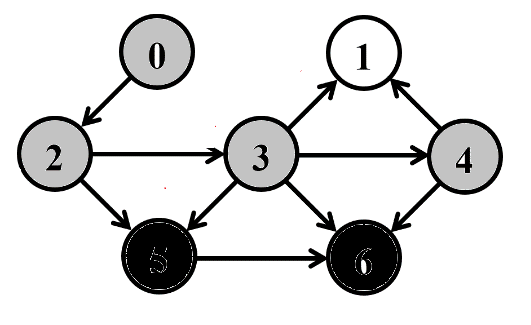
****

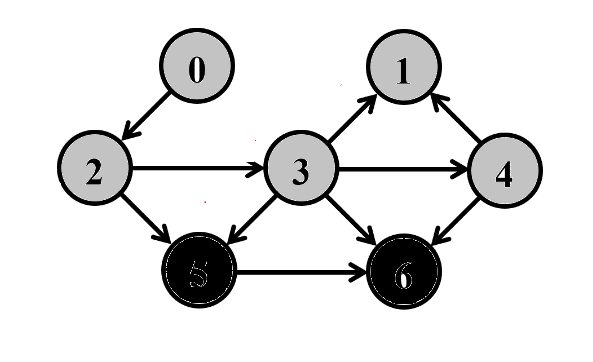
|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **6** |  |  |  |  |  |  |

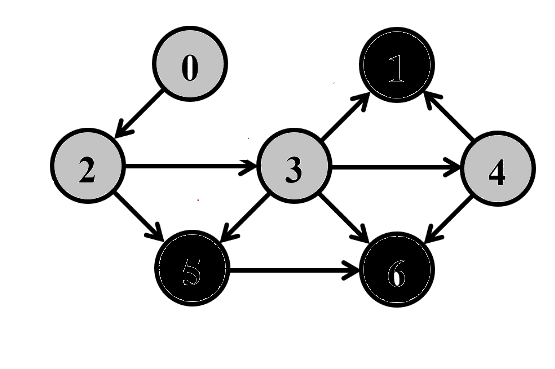
****

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **5** | **6** |  |  |  |  |  |

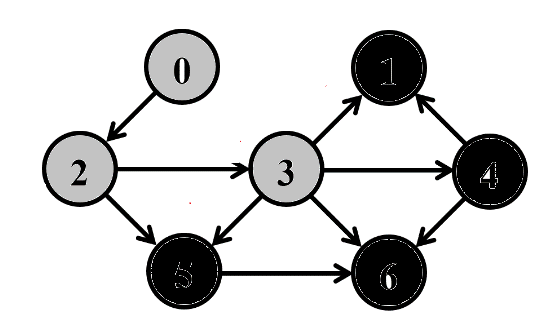




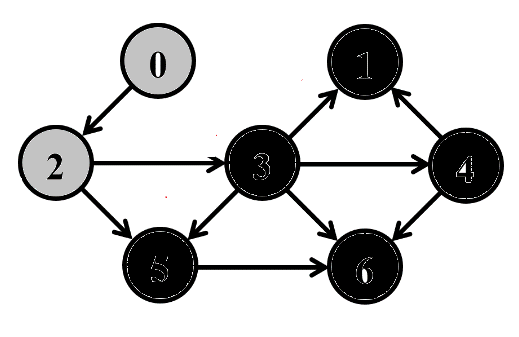




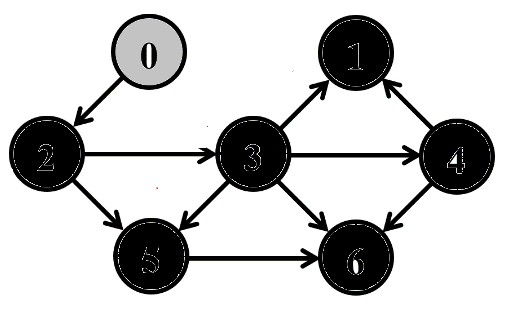
|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **1** | **5** | **6** |  |  |  |  |



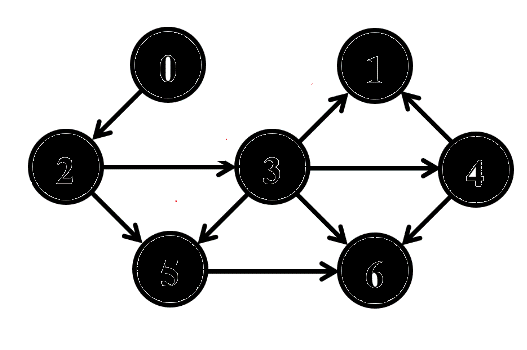
|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **4** | **1** | **5** | **6** |  |  |  |



|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **3** | **4** | **1** | **5** | **6** |  |  |



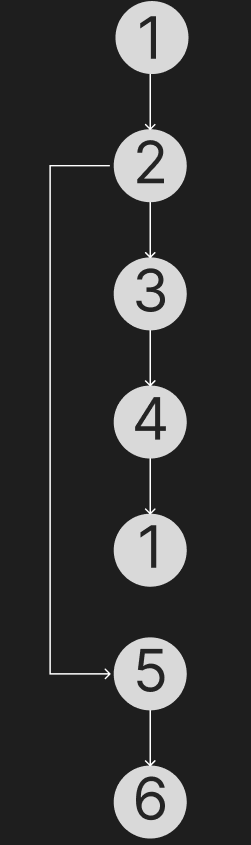
|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **2** | **3** | **4** | **1** | **5** | **6** |  |



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

**Результат**

Топологическая сортировка: 6 5 1 4 3 2 0

****

***Задание 3.*** 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.

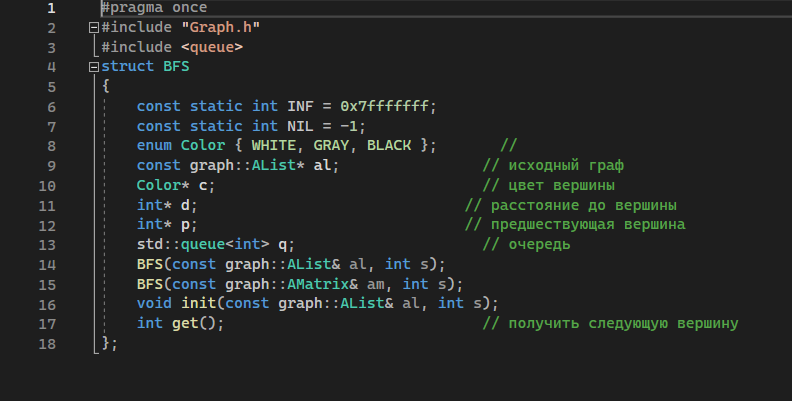


Рисунок 1 -- файл BFS.h

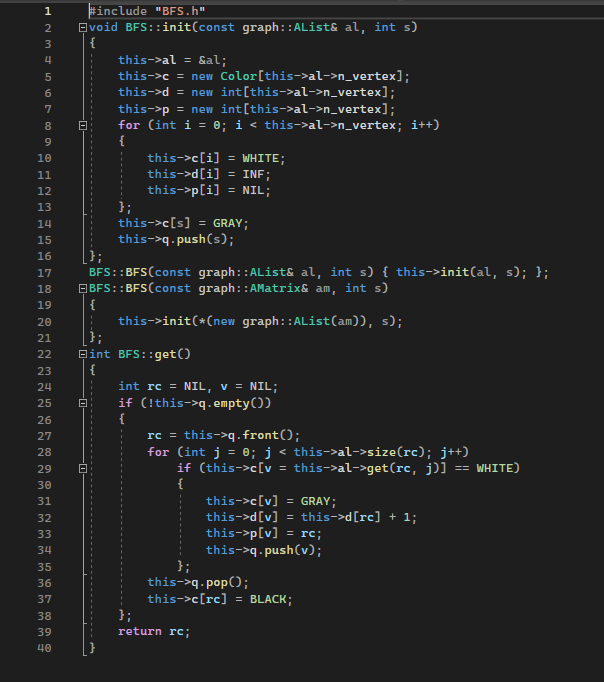


Рисунок 2 -- файл BFS.cpp



Рисунок 3 – демонстрация работы функции BFS

***Задание 4.*** Разработать функцию **DFS**  обхода вершин графа, используя метод поиска глубину. Продемонстрировать работу функции. Копии экрана вставить в отчет.

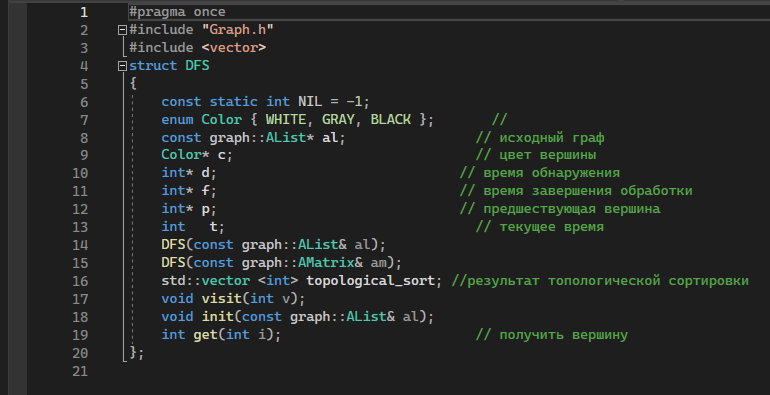


Рисунок 4 -- файл DFS.h



Рисунок 5 -- файл DFS.cpp



Рисунок 6 – результат выполнения программы

***Задание 5.*** Доработайте функцию **DFS**,для выполнения топологической сортировки графа. Продемонстрировать работу функции. Копии экрана вставить в отчет.

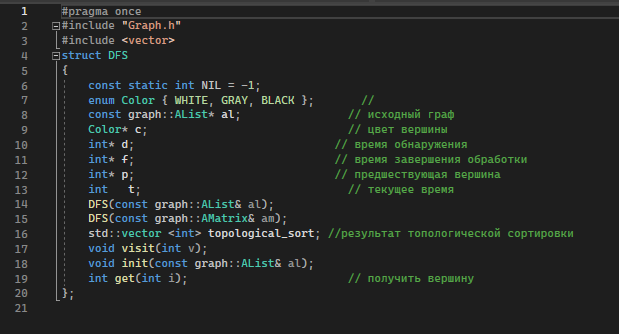


Рисунок 7 -- файл DFS.h

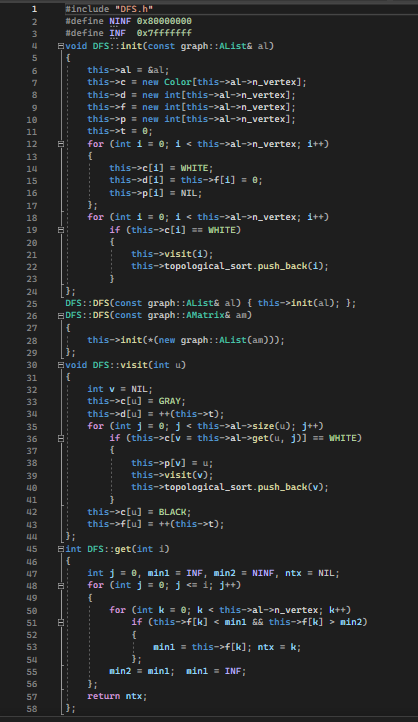


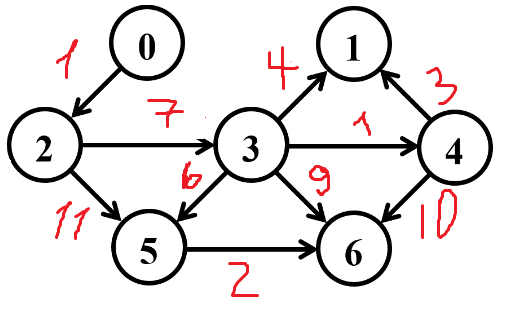
Рисунок 8 -- файл DFS.cpp



Рисунок 9 – результат работы программы

***Задание 6.*** По графу, соответствующему варианту составить минимальное остовное дерево по алгоритму Прима. Шаги построения отразить в отчете.

Веса ребер принять:

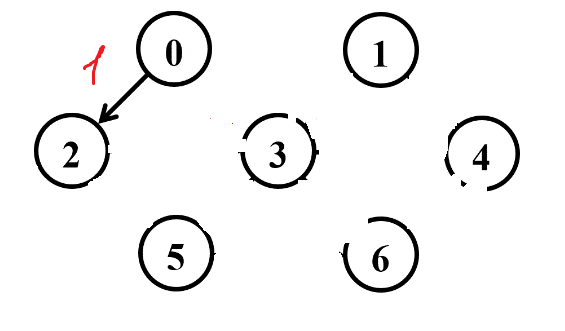


Шаг 1:

Предположим, что в множество U пока только входит вершина 0, нужно найти ребро с минимальной стоимостью, которое соединяет вершину 0 с вершиной, которая в множество U не входит.

U = {0}

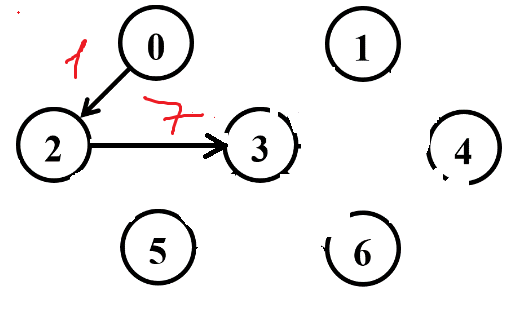
Вершина 0 соединена с вершинами 2. Вес ребра равен 1. Тогда, вершину 2 включаем в множество U, а ребро 0-2 — в множество T.



Шаг 2:

U = {0, 2}

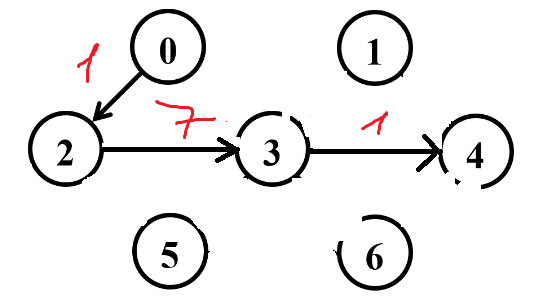
Вершины 0 и 2 соединены с вершинами 3 и 5. Веса их ребер равно 7 и 11 соответственно. Из этих ребер минимальной стоимостью обладает ребро 2-3. Тогда, вершину 3 включаем в множество U, а ребро 2-3 — в множество T.



Шаг 3:

U = {0, 2, 3}

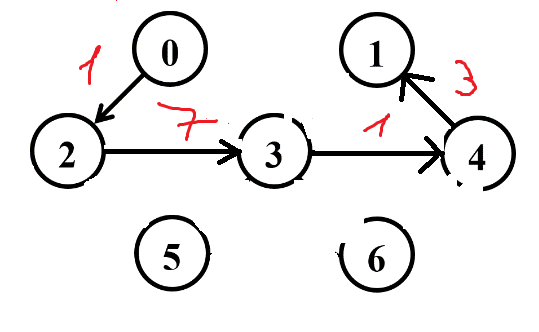
Вершины 0, 2 и 3 соединены с вершинами 1, 5(2-5), 4, 6, 5(3-5). Из этих ребер минимальной стоимостью обладает ребро 3-4 (вес = 1). Тогда, вершину 4 включаем в множество U, а ребро 3-4 — в множество T.



Шаг 4:

U = {0, 2, 3, 4}

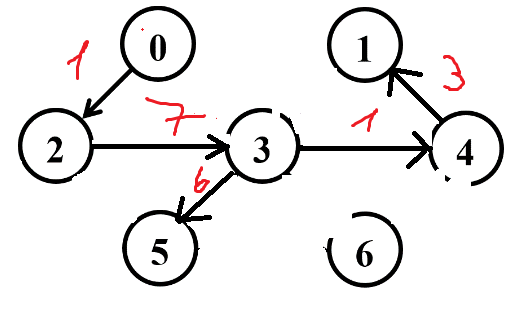
Вершины 0, 2, 3 и 4 соединены с вершинами 1, 6, 5. Из ребер минимальной стоимостью обладает ребро 4-1 (вес = 3). Тогда, вершину 1 включаем в множество U, а ребро 4-1 — в множество T.



Шаг 5:

U = {0, 2, 3, 4, 1}

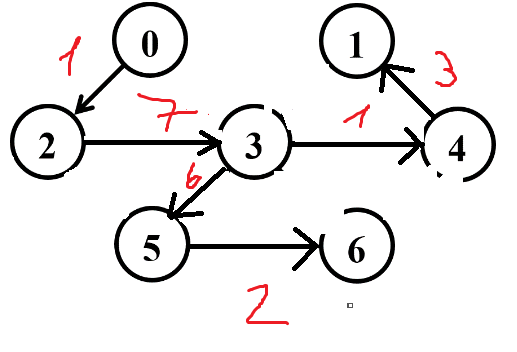
Вершины 0, 2, 3, 4 и 1 соединены с вершинами 5(2-5),5(3-5), 6(3-6), 6(4-6) .Веса их ребер равно 11, 6, 9, 10 соответственно. Из этих ребер минимальной стоимостью обладает ребро 3-5 (вес = 6). Тогда, вершину 5 включаем в множество U, а ребро 3-5 — в множество T.



Шаг 6:

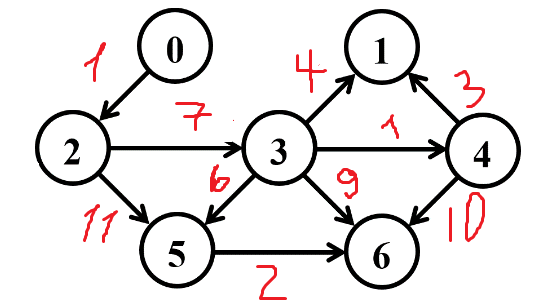
U = {0, 2, 3, 4, 1, 5}

Оставшаяся вершина 6 соединяется с вершиной 5, 3 и 4, их веса равны 2, 9 и 10 соответственно. Тогда, вершину 6 включаем в множество U, а ребро 5-6 — в множество T.



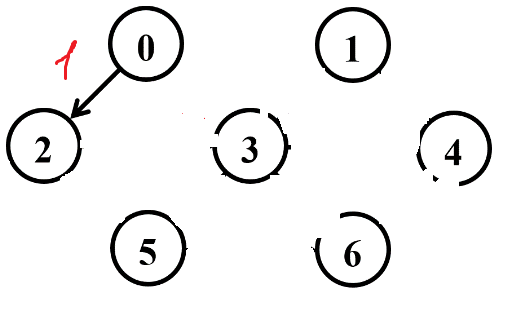
Вес минимального остовного дерева: 20

***Задание 7.*** По графу, соответствующему варианту составить минимальное остовное дерево по алгоритму Крускала. Шаги построения отразить в отчете

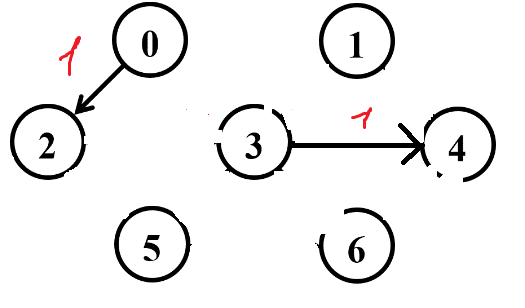


Шаг 1:

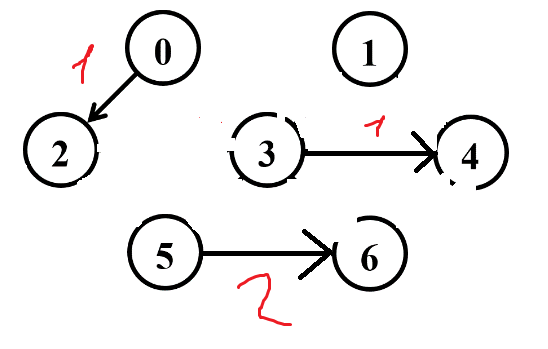
Из всех ребер выбираем ребро с минимальной стоимостью (весом) и включаем это ребро в остовное дерево. Повторяем так далее, однако смотрим на то, чтобы следующее ребро соединяло вершины из разных компонент.



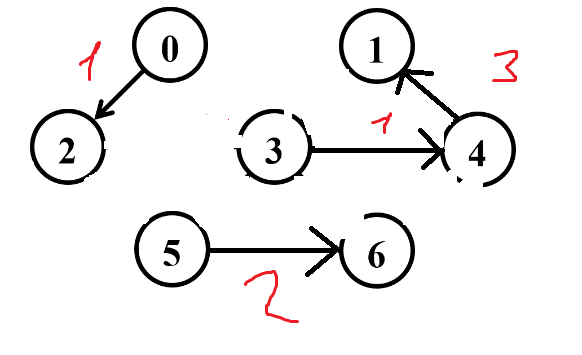
Шаг 2:



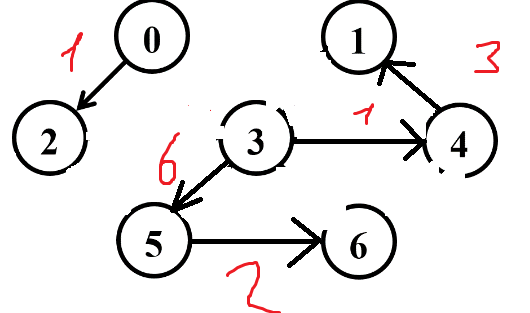
Шаг 3:



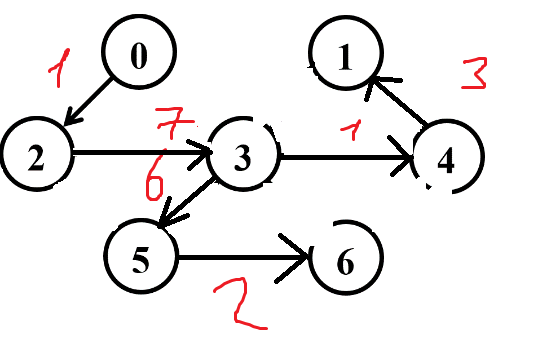
Шаг 4:



Шаг 5:



Шаг 6:



Вес минимального остовного дерева: 20

**Вывод:** в ходе выполнения лабораторной работы были изучены и реализованы различные способы представления графов, такие как матрица смежности и список смежности. Были изучены и реализованы алгоритмы поиска в ширину и глубину. Был изучен и реализован алгоритм топологической сортировки графов. Этот алгоритм применяется для упорядочивания вершин графа таким образом, чтобы все ребра шли от вершин с меньшими индексами к вершинам с большими индексами. Были изучены и разобраны алгоритмы Прима и Крускала для построения минимального остовного дерева во взвешенном связном графе.

Учреждение образования

«Белорусский государственный технологический университет»

**Лабораторная работа №7**

Сетевые модели

Выполнил:

Студент 2 курса 4 группы ФИТ

Станчик Максим Андреевич

2024 г.

**Задание 1,2. Структурное планирование и календарное планирование.**

Подумайте и выделите в проекте, согласно вашему варианту не менее 4 этапов работ. Также разбейте полученные этапы на задачи, их количество в совокупности по этапам должно быть не менее 12. Пример оформления задания смотрите в приложении ниже и в лекционном материале по теме. Распределите время, отпущенное на ваш проект согласно вариантам, на выделенные вами этапы. Скорректируйте сформулированные вами задачи, если это необходимо.

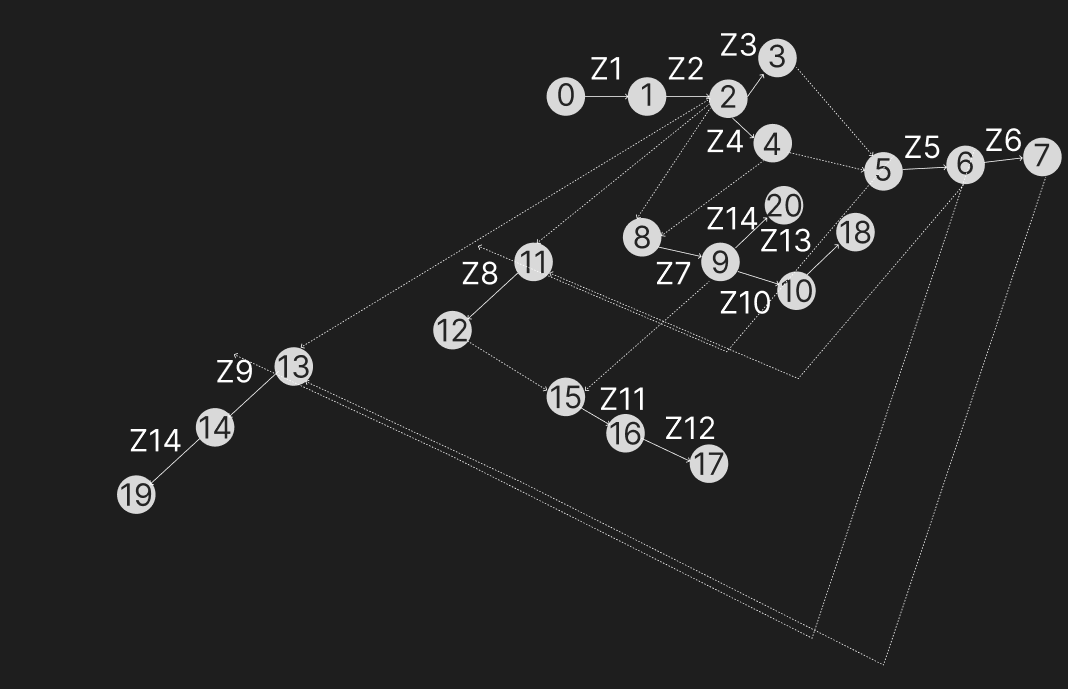
|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Код операции | Наименование операции | Предшествующие операции | t | Конечное событие |
| I. Исследование и планирование | | | |  |
| Z1 | Анализ рынка и требований |  | 2 | 1 |
| Z2 | Создание концепции игры | Z1 | 3 | 3 |
| Z3 | Планирование проекта и распределение ресурсов | Z2 | 3 | 5 |
| II. Проектирование | | | |  |
| Z4 | Проектирование интерфейса и UX | Z2 | 4 | 7 |
| Z5 | Проектирование игровых механик | Z3, Z4 | 4 | 9 |
| Z6 | Проектирование графики и звуков | Z3, Z4 | 4 | 11 |
| III. Разработка и кодирование | | | |  |
| Z7 | Разработка игровых механик | Z4, Z2 | 6 | 13 |
| Z8 | Разработка интерфейса | Z5, Z2 | 5 | 15 |
| Z9 | Создание графики и звуков | Z6, Z2 | 6 | 17 |
| IV. Тестирование и отладка | | | |  |
| Z10 | Тестирование игры | Z7 | 5 | 19 |
| Z11 | Отладка и устранение ошибок | Z7, Z8 | 3 | 21 |
| Z12 | Подготовка к выпуску и оптимизация | Z11 | 2 | 23 |
| Z13 | Устранение ошибок | Z10 | 5 | 25 |
| V. Дополнительные работы | | | |  |
| Z14 | Развертывание сервера | Z7 | 5 | 27 |

**Задание 3. Сетевой график, нахождение критического пути.**

Согласно составленному перечню задач и распределённому времени составьте сетевой график вашего проекта. Помните о правилах составления графика и вводите фиктивные операции и операции ожидания если это необходимо. Найдите критический путь в составленном вами сетевом графике и обоснуйте его нахождение. Критический путь может быть меньше, чем время, отведенное на выполнение всех задач. Выделите, какие операции принадлежат критическому пути.

**Задание №3. Сетевой график, нахождение критического пути**

**Условие:** Согласно составленному перечню задач и распределённому времени составьте сетевой график вашего проекта. Помните о правилах составления графика и вводите фиктивные операции и операции ожидания если это необходимо.



Критический путь в данной сети задач будет следующим:

Z1 -> Z2 -> Z4 -> Z5 -> Z7 -> Z10 -> Z11 -> Z12 -> Z13 -> Z14

Сложив все эти значения, получим общее время выполнения всех задач по критическому пути: 2 + 3 + 4 + 4 + 6 + 5 + 3 + 2 + 5 + 5 = 39 (дней).

Это является минимальным временем, необходимым для завершения всего проекта, при условии, что все задачи по критическому пути выполняются последовательно и без задержек.

**\*Задание №4. Оптимизация**

**Условие:** Предложите варианты оптимизации вашего проекта с привлечением денежных средств или человеческого ресурса.

**Выполнение:**

Для оптимизации проекта "Создание мобильной игры" можно рассмотреть следующие варианты:

1. Аутсорсинг некоторых задач разработки или дизайна для снижения нагрузки на внутренний ресурс.
2. Участие в конкурсах или программе финансирования игровых проектов для получения дополнительных средств.
3. Установка аналитических инструментов для мониторинга и анализа пользовательского поведения и метрик игры, что позволит принимать обоснованные решения по оптимизации и улучшению игрового процесса.
4. Поиск спонсоров или инвесторов, которые могут предоставить дополнительные денежные средства для развития проекта.
5. Партнерство с другими компаниями или брендами для создания совместных акций или продвижения игры.
6. Установка внутриигровых покупок или рекламы для монетизации игры.

**Вывод:** в ходе данной лабораторной работы мы ознакомились с основными принципами и методами сетевого планирования, включая метод критического пути и метод оценки и обзора программ. Мы успешно построили сетевую диаграмму проекта, определили критический путь и его стоимость. Приобретенные знания и использование этих методов позволят нам эффективно управлять проектами, планировать и контролировать время и ресурсы, необходимые для выполнения задач. Эти инструменты являются важными для менеджеров проектов и могут значительно повысить эффективность управления проектами.