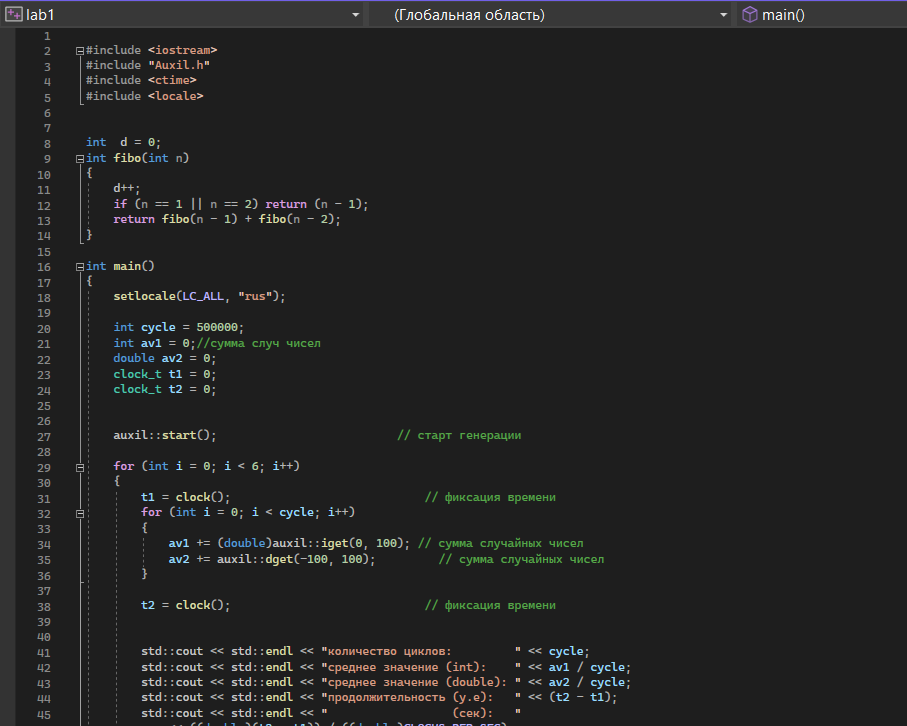
**Лабораторная работа №1. Вспомогательные функции**

**Цель работы:** приобретение навыков составления и отладки программ с использованием пользовательских функций для замера продолжительности процесса вычисления.

**Ход работы**

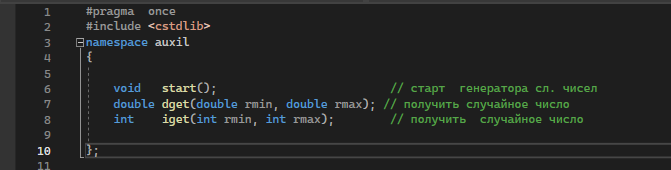
## **1.1. Написание кода программы.**

Код файла main.cpp приведен в листинге 1.1.



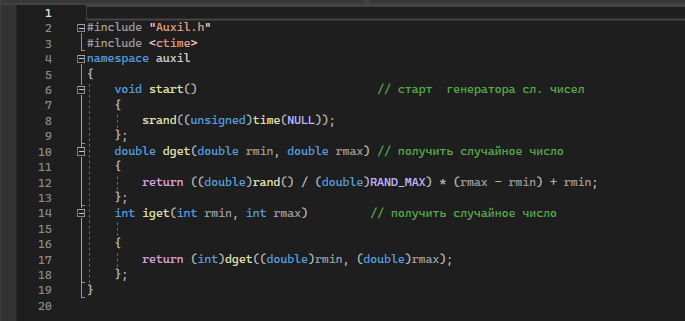
Листинг 1.1. Содержание файла main.cpp

Код файла Auxil.h приведен в листинге 1.2.



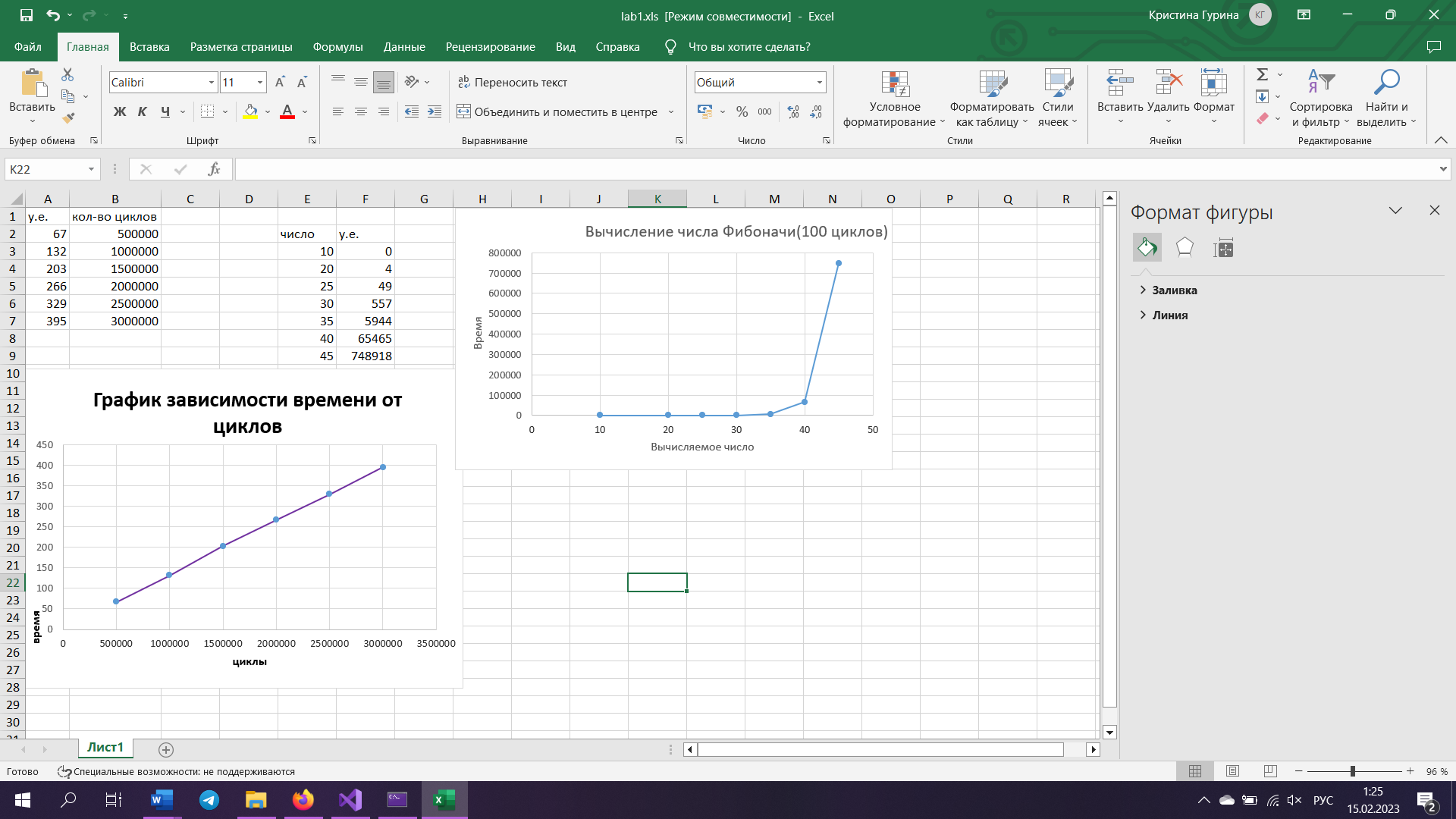
Листинг 1.2. Содержание файла Auxil.h

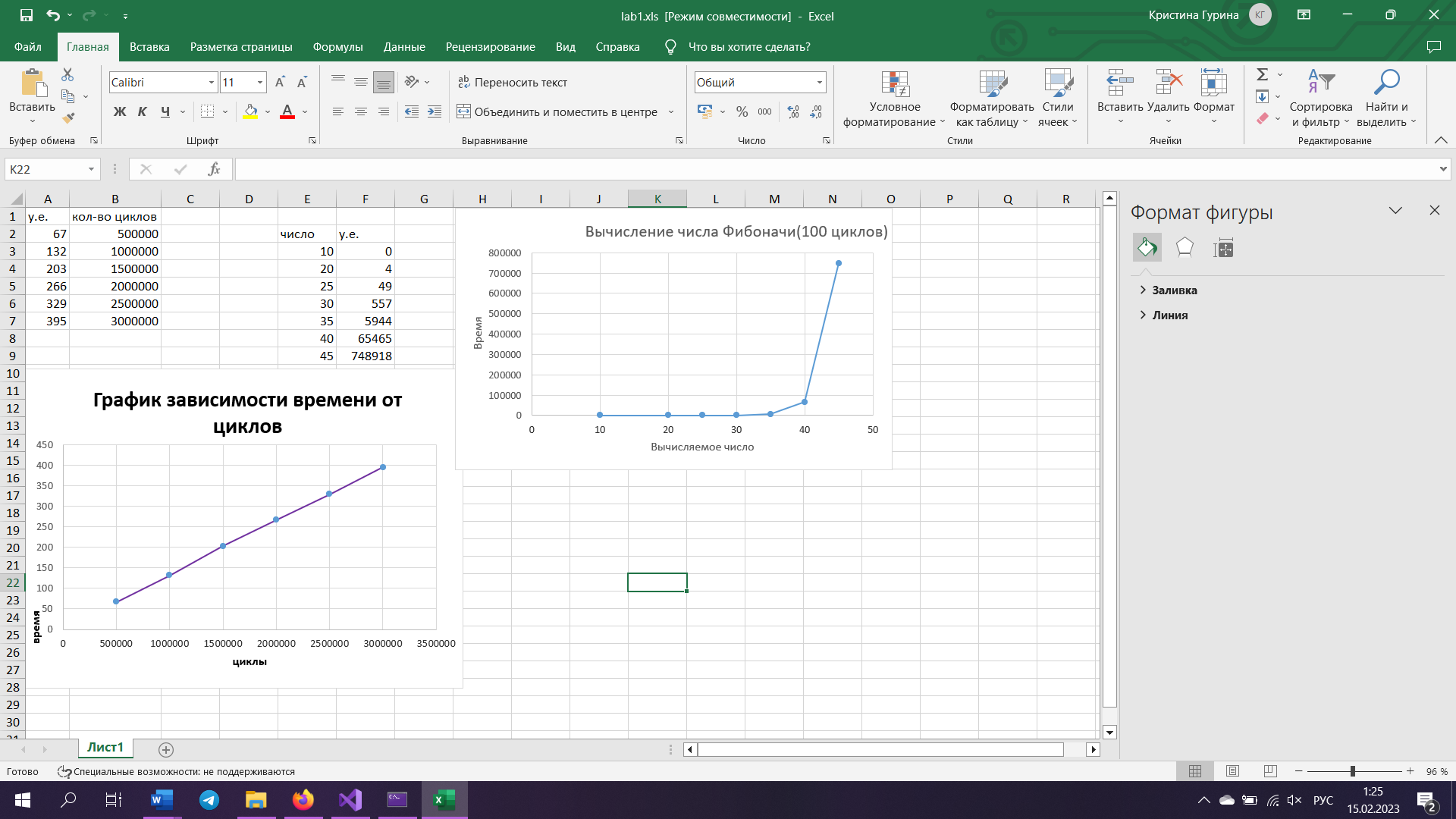
Код файла Auxil.cpp приведет в листинге 1.3.



Листинг 1.3. Содержание файла Auxil.cpp

## **1.2. Измерение скорости выполнения функции генерации случайных чисел.**





Результаты измерений и соответствующий график приведены на рисунке 1.1

Вывод: скорость выполнения программы линейно зависит от количества итераций цикла.

## **1.3. Измерение скорости вычисления n числа Фибоначи за 100 циклов.**

Результаты измерений и соответствующий график приведены на рисунке 1.4

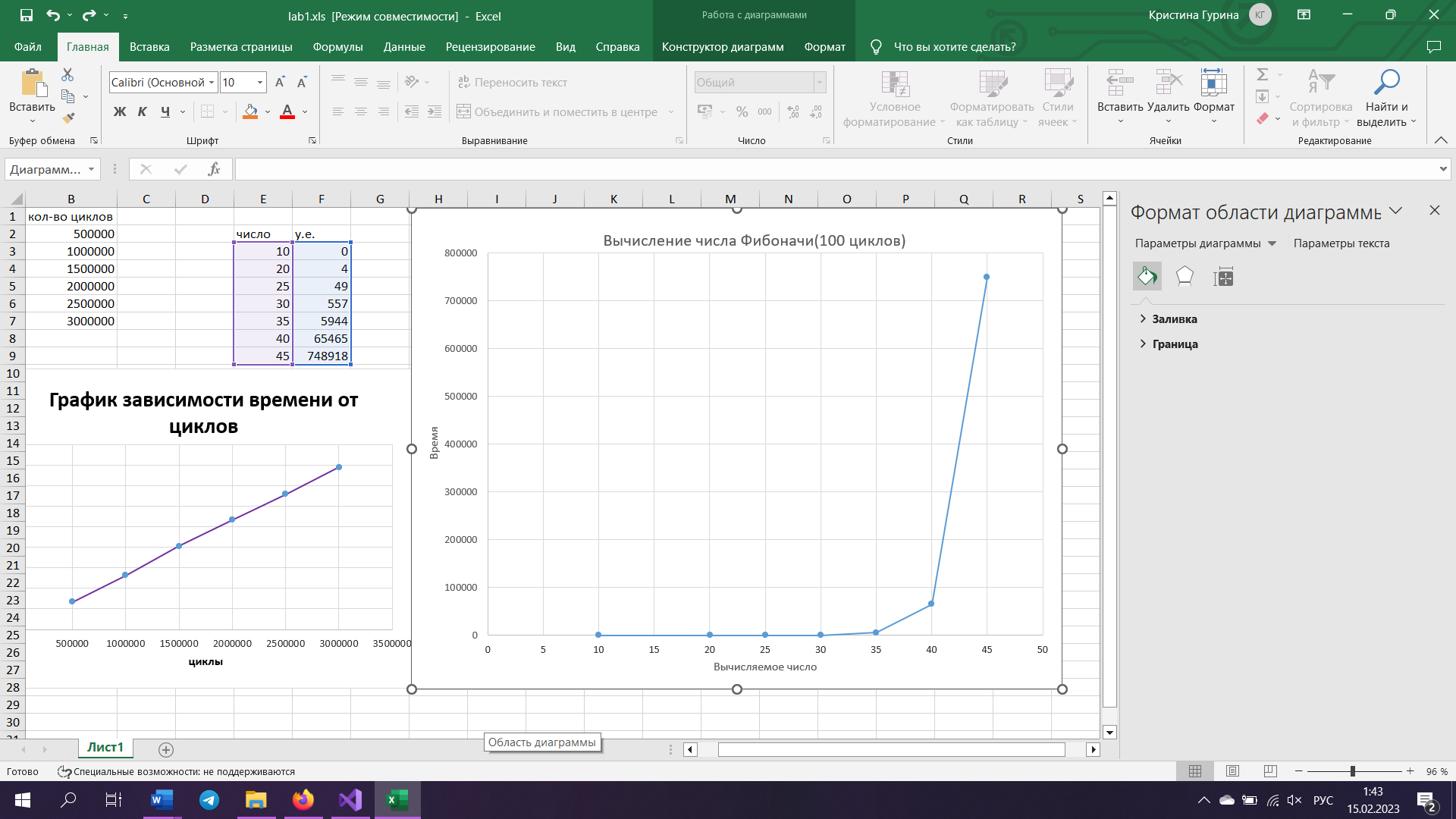
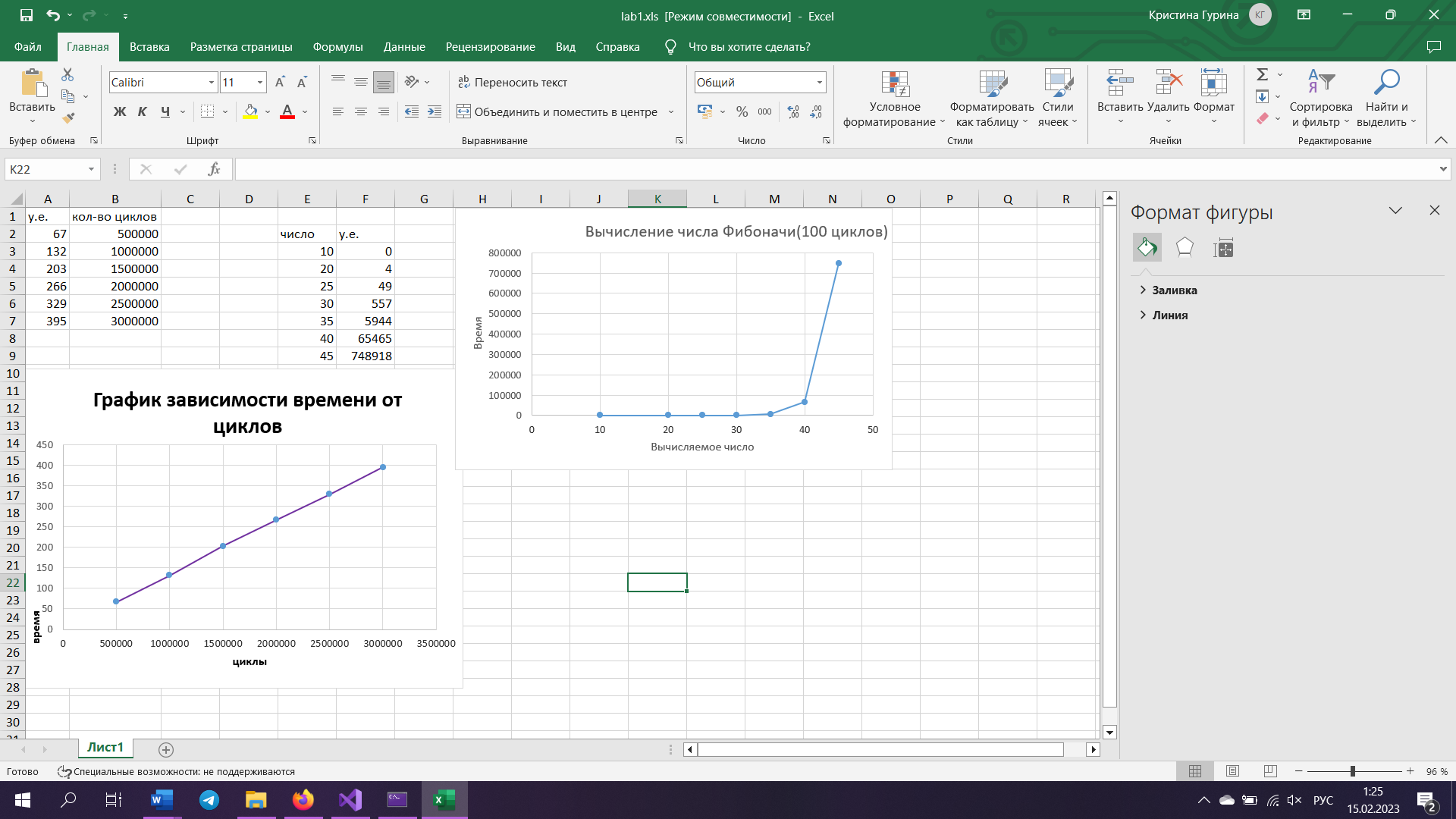


Рисунок 1.2 Результаты измерений и их график

Вывод: сложность данного алгоритма является экспоненциальной. Это подтверждает и график.

**Лабораторная работа №2. Комбинаторные алгоритмы решения оптимизационных задач**

**Цель работы:** приобрести навыки разработки генераторов подмножеств, перестановок, сочетаний и размещений на С++; научиться применять разработанные генераторы для решения задач о рюкзаке (упрощенную, коммивояжера, об оптимальной загрузке судна и об оптимальной загрузке судна с центровкой.

**Ход работы**

## **1.1. Разработка генераторов**

В ходе лабораторной работы были разобраны и разработаны с помощью материалов из лабораторной работы следующие генераторы:

* генератор подмножеств заданного множества
* генератор сочетаний
* генератор перестановок
* генератор размещений

## **1.2. Решение задачи об оптимальной загрузке судна**

**Условие:** (веса контейнеров сгенерировать случайным образом: ограничение по общему весу – 1500 кг., количество мест на судне для контейнеров – 5, количество контейнеров 25, веса контейнеров 100 – 900 кг., доход от перевозки 10 – 150 у.е.);

Для генерации веса контейнеров использовался генератор из лабораторной работы №1.

Результат представлен на рисунке 2.1.

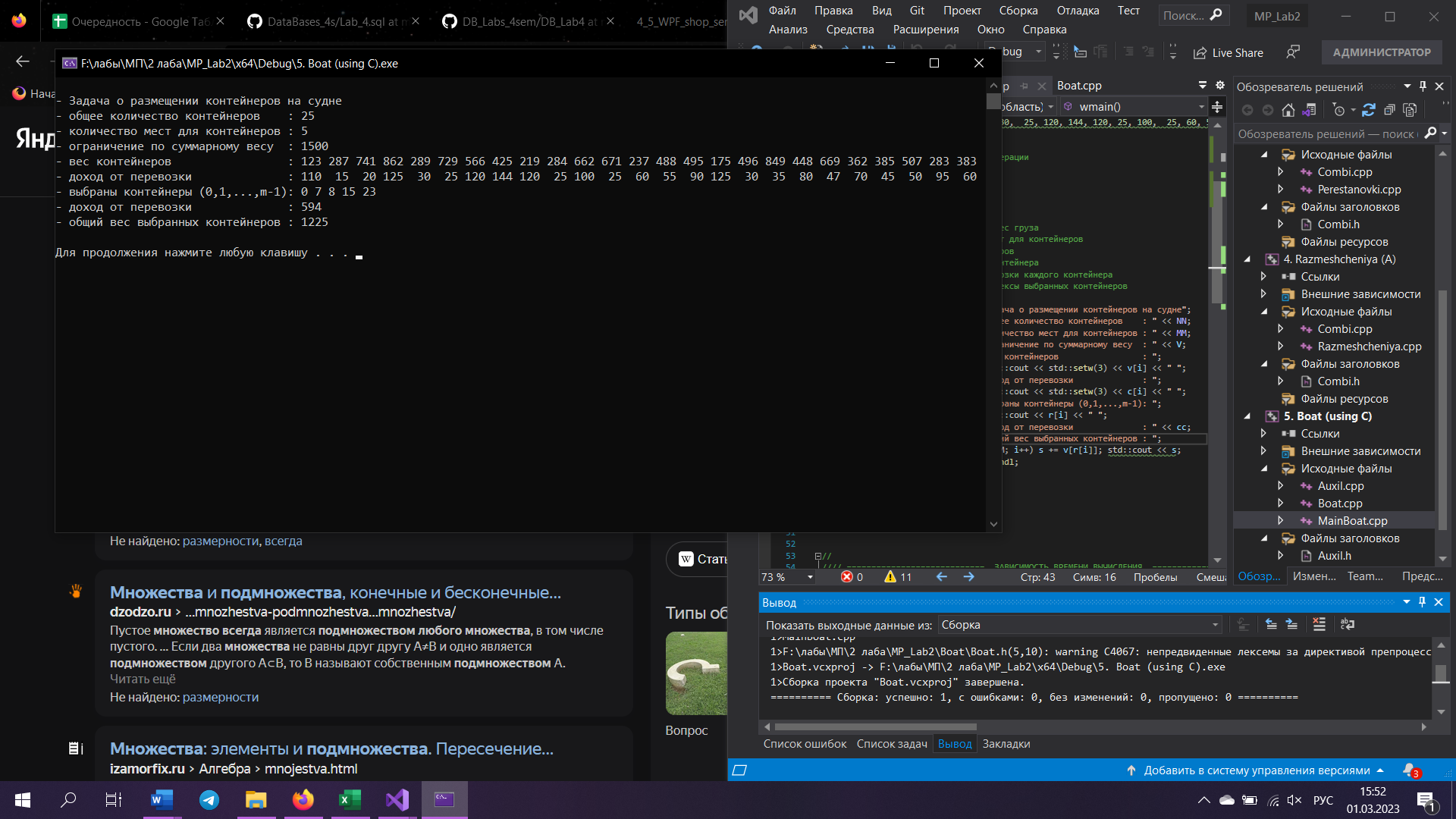


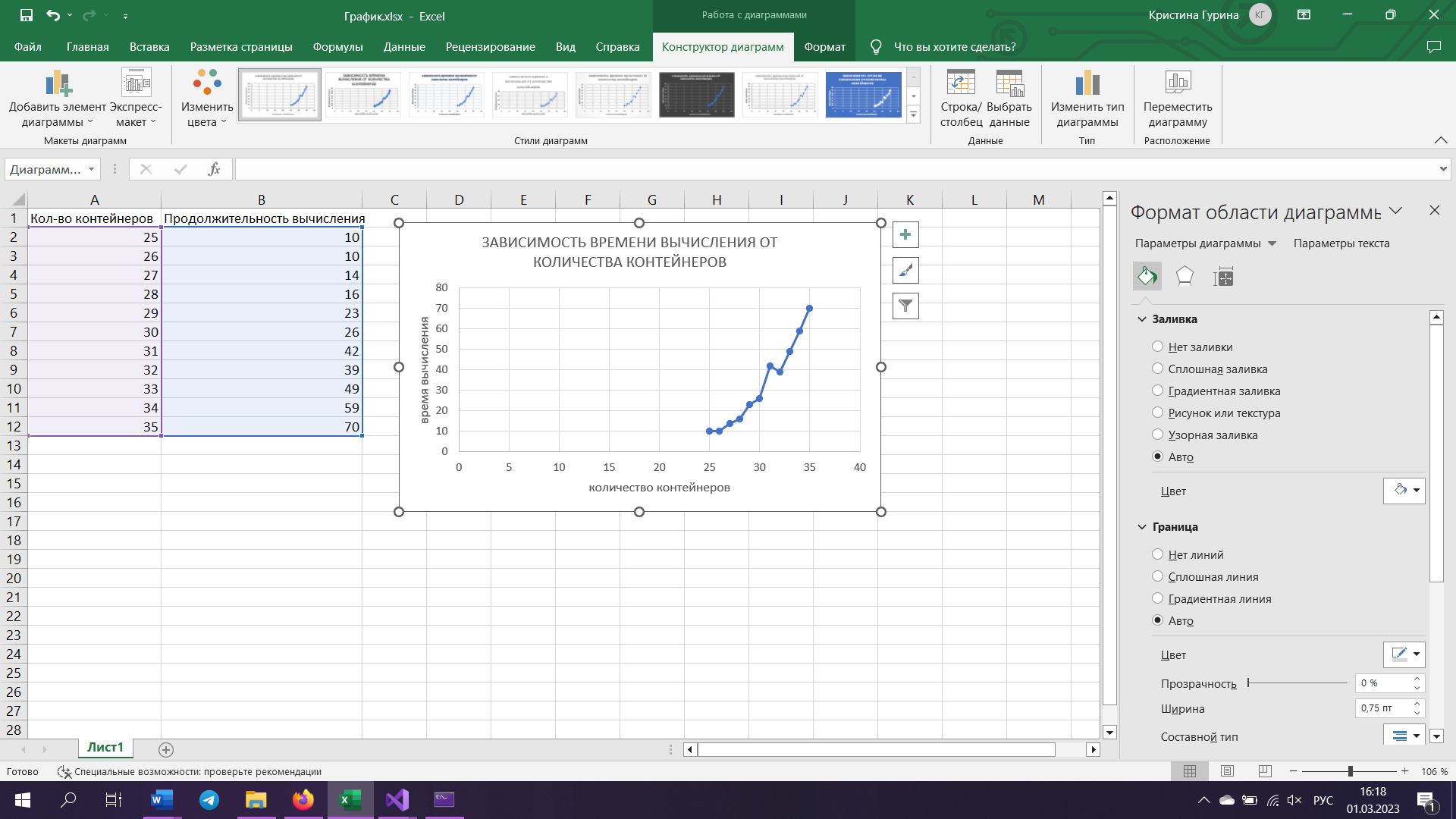
Рисунок 2.1. Результат решения задачи об оптимальной загрузке судна

## **1.3. Исследование зависимости времени вычисления необходимого для решения задачи об оптимальной загрузке судна с использованием генератора сочетаний от количества контейнеров.**

Исходные данные:

* количество мест на судне для контейнеров – 6
* количество контейнеров 25 – 35

Результаты измерений и соответствующий график приведены на рисунке 2.1



Вывод: решение задачи сводится к генерации всех допустимых векторов, а затем выбору из них наиболее оптимального. Вид графика согласуется с оценкой сложности алгоритма генерации сочетаний элементов .

**Лабораторная работа №3. Метод ветвей и границ. Задача коммивояжера и методы её решения.**

**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**

**Условие:** сформулировать условие задачи коммивояжера с параметром. Для этого:

* принять элементы матрицы расстояний равными:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Город** | **1** | **2** | **3** | **4** | **5** |
| **1** |  | 2 \* n | 21 + n |  | n |
| **2** | n |  | 15 + n | 68 - n | 84 - n |
| **3** | 2 + n | 3 \* n |  | 86 | 49 + n |
| **4** | 17 + n | 58 - n | 4 \* n |  | 3 \* n |
| **5** | 93 - n | 66 + n | 52 | 13 + n |  |

где *n* – номер варианта или номер по журналу;

Для выполнения лабораторной работы была составлена матрица расстояний в соответствии и номером варианта 3:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Город** | **1** | **2** | **3** | **4** | **5** |
| **1** |  | 6 | 24 |  | 3 |
| **2** | 3 |  | 18 | 65 | 81 |
| **3** | 5 | 9 |  | 86 | 52 |
| **4** | 20 | 55 | 12 |  | 9 |
| **5** | 90 | 69 | 52 | 16 |  |

**Задание 2.**

**Условие:** решить сформулированную задачу методом ветвей и границ.

**Решение**

Приведённая по строкам матрица:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Город** | **1** | **2** | **3** | **4** | **5** |  |
| **1** |  | 3 | 21 |  | 0 | 3 |
| **2** | 0 |  | 15 | 62 | 78 | 3 |
| **3** | 0 | 4 |  | 81 | 47 | 5 |
| **4** | 11 | 46 | 3 |  | 0 | 9 |
| **5** | 74 | 53 | 36 | 0 |  | 16 |

α =3+3+5+9+16=36

Полностью приведённая матрица:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Город** | **1** | **2** | **3** | **4** | **5** |  |
| **1** |  | 0 | 18 |  | 0 | 3 |
| **2** | 0 |  | 12 | 62 | 78 | 3 |
| **3** | 0 | 1 |  | 81 | 47 | 5 |
| **4** | 11 | 43 | 0 |  | 0 | 9 |
| **5** | 74 | 50 | 33 | 0 |  | 16 |
|  | 0 | 3 | 3 | 0 | 0 | **42** |

β = 3+3=6

Нижняя граница длины кольцевого маршрута:

φ = 36+6=42

01,2 = 1; 02,1 = 12; 03,1 = 12; 04,3 = 12; 05,4 = 95;

01,5 = 0;

Выбираем максимальное значение и получаем путь из города 5 в 4 и соответственно получим граф:

137

42

В случае если мы идём по маршруту (5, 4) то расстояние будет равно нижней границе кольцевого маршрута, то есть 42, а если не пойдём, то расстояние будет равно 42 + 95 = 137.

Так как меньшее расстояние 42, то мы идём из города 5 в город 4. Следовательно, для дальнейших вычислений вычёркиваем 5 строку и 4 столбец из матрицы и делаем обратный путь (4, 5) равным INF:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Город** | **1** | **2** | **3** | **5** |
| **1** |  | 0 | 18 | 0 |
| **2** | 0 |  | 12 | 78 |
| **3** | 0 | 1 |  | 47 |
| **4** | 11 | 43 | 0 |  |

Полученная матрица уже является полностью приведённой, поэтому нижняя граница кольцевого маршрута останется неизменной и равной 42.

01,2 = 1; 02,1 = 12; 03,1 = 12; 04,3 = 23;

01,5 = 47;

Выбираем максимальное значение и получаем путь из города 1 в 5 и соответственно получим граф:

89

42

В случае если мы идём по маршруту (1, 5) то расстояние будет равно нижней границе кольцевого маршрута, то есть 42, а если не пойдём, то расстояние будет равно 42 + 47 = 89.

Так как меньшее расстояние 42, то мы идём из города 1 в город 5. Следовательно, для дальнейших вычислений вычёркиваем 1 строку и 5 столбец из матрицы:

|  |  |  |  |
| --- | --- | --- | --- |
| **Город** | **1** | **2** | **3** |
| **2** | 0 |  | 12 |
| **3** | 0 | 1 |  |
| **4** | 11 | 43 | 0 |

Данная матрица не является полностью приведённой, поэтому её надо привести по столбцам и соответственно она примет вид:

|  |  |  |  |
| --- | --- | --- | --- |
| **Город** | **1** | **2** | **3** |
| **2** | 0 |  | 12 |
| **3** | 0 | 0 |  |
| **4** | 11 | 42 | 0 |

β = 0 + 1 + 0 = 1;

Следовательно, изменится нижняя граница кольцевого маршрута и соответственно:

φ = 42 + 1 = 43

02,1 = 23; 03,1 = 0; 04,3 = 23;

Выбираем максимальное значение и получаем путь из города 4 в 3 и соответственно получим граф:

43

66

В случае если мы идём по маршруту (4, 4) то расстояние будет равно нижней границе кольцевого маршрута, то есть 43, а если не пойдём, то расстояние будет равно 43 + 23 = 66.

Так как меньшее расстояние 43, то мы идём из города 4 в город 3. Следовательно, для дальнейших вычислений вычёркиваем 4 строку и 3 столбец из матрицы:

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

После анализа данной матрицы к нашему графу добавятся пути (3, 2) и (2, 1). Соответственно минимальное расстояние будет равно 43, и граф будет иметь следующий вид:

42

43

42

137

138

89

R3,2

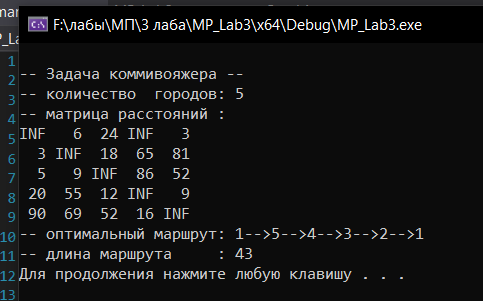
R2,1

Расставим переходы между городами в правильной последовательности и соответственно получим (1, 5), (5, 4), (4, 3), (3, 2), (2, 1).

**Задание 3**

**Условие:** проверить полученное решение при помощи генератора перестановок (см. лаб. 2, задание 5.1.) и включить копию экрана с решением в отчет.

Проверка правильности решения с помощью генератора перестановок представлена на рисунке



**Вывод**: Мы освоили общие принципы решения задач методом ветвей и границ, решили задачу о коммивояжере данным методом, сравнили полученное решение задачи с комбинаторным методом перестановок.

**Лабораторная работа 4**

**ДИНАМИЧЕСКОЕ ПРОГРАММИРОВАНИЕ**

**ЦЕЛЬ РАБОТЫ:** освоить общие принципы решения задач методом динамического программирования, сравнить полученные решения задач с рекурсивным методом.

**Ход выполнения работы**

**Задание 1.** На языке С++ сгенерировать случайным образом строку букв латинского алфавита  длиной  символов и длиной .

**Решение:**

#define \_rand(min, max) ( rand() % ((max) - (min) + 1) + (min) )

int \_tmain(int argc, \_TCHAR\* argv[])

{

setlocale(LC\_ALL, "rus");

srand(time(NULL));

char abc[25]; // наш алфавит

char s1[300];

char s2[250];

// заполняем массив

for (int i = 97, n = 0; i <= 122; ++i, ++n)

{

abc[n] = (char)i;

}

for (int i = 0; i < 300; i++)

{

s1[i] = abc[\_rand(0, 25)];

}

for (int i = 0; i < 250; i++)

{

s2[i] = abc[\_rand(0, 25)];

}

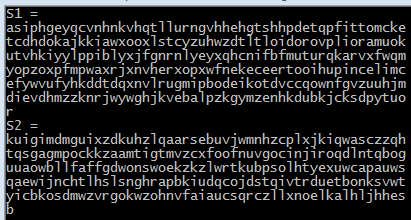


Рис.1 – пример генерации строк

**Задание 2.** Вычислить двумя способами (рекурсивно и с помощью динамического программирования)  – Левенштейна для , где - длина строки ,  - строка, состоящая из первых  символов строки . (копии экрана и код вставить в отчет).

**Решение:**

Ниже приведены варианты реализации нахождения дистанции Левенштейна при помощи динамического программирования и при помощи рекурсивного алгоритма.

Исходный код реализации через динамическое программирование:

int min3(int x1, int x2, int x3)

{ return std::min(std::min(x1,x2),x3); }

int levenshtein(int lx, const char x[],

int ly, const char y[])

{

int \*\*matr;

int w, left, top, left\_top;

matr = new int\*[lx];

for (int i = 0; i < lx; i++)

matr[i] = new int[ly];

matr[0][0] = 0;

for (int i = 1; i < lx; i++)

matr[i][0] = i;

for (int j = 1; j < ly; j++)

matr[0][j] = j;

for (int i = 1; i < lx; i++)

for (int j = 1; j < ly; j++){

w = x[i - 1] == y[j - 1] ? 0 : 1;

top = matr[i - 1][j];

left = matr[i][j - 1];

left\_top = matr[i - 1][j - 1];

matr[i][j] = std::min(left\_top + w, std::min(top + 1, left + 1));

}

return matr[lx-1][ly-1];

}

Пример реализации рекурсивным методом:

int min3(int x1, int x2, int x3)

{ return std::min(std::min(x1,x2),x3); }

int levenshtein\_r(int lx, const char x[],

int ly, const char y[])

{

int rc = 0;

if (lx == 0) rc = ly;

else if (ly == 0) rc = lx;

else if (lx == 1 && ly == 1 && x[0] == y[0]) rc = 0;

else if (lx == 1 && ly == 1 && x[0] != y[0]) rc = 1;

else rc = min3(

levenshtein\_r(lx-1, x, ly, y)+1,

levenshtein\_r(lx, x, ly-1, y)+1,

levenshtein\_r(lx-1, x, ly-1, y)+(x[lx-1] == y[ly-1]?0:1)

);

return rc;

};

На рисунке 2 представлены дистанции Левенштейна вычисленные при помощи метода динамического программирования, а также рекурсивным алгоритмом.

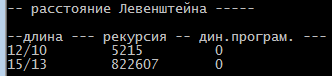


Рис. 2 – проверка работоспособности решений

**Задание 3.** Выполнить сравнительный анализ времени затраченного на вычисление дистанции Левенштейна для двух методов решения. Построить графики зависимости времени вычисления от . (копии экрана и график вставить в отчет).

**Решение:**

На графике, представленном на рисунке 3, можно заметить, что выполненные с помощью динамического алгоритма, вычисления производятся в разы быстрее, чем с помощью рекурсивного алгоритма.

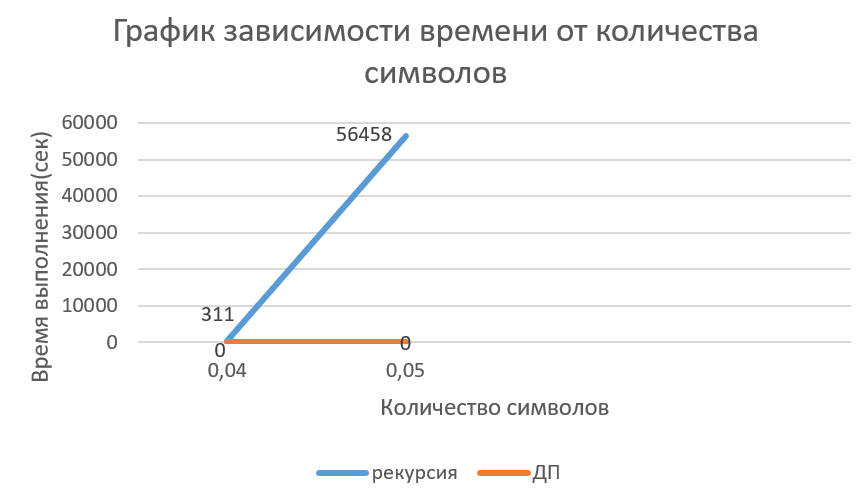


Рис. 3 – график зависимости времени от количества символов

**Задание 4.** Реализовать вручную пример вычисления дистанции Левенштейна при помощи рекурсивного алгоритма (в соответствии с вариантом) (каждый шаг алгоритма по примеру из лекции вставить в отчет).

|  |  |
| --- | --- |
| Вариант 3 | |
| Лом | Гомон |

1. Лом, Гомон = min(Ло, Гомон+1)

( Лом, Гомо+1)

(Ло, Гомо+1)

1. Ло, Гомон = min(Л, Гомон+1)

(Ло, Гомо+1)

(Л, Гомо+1)

1. Лом, Гомо = min(Ло, Гомо+1)

(Лом, Гом+1)

(Ло, Гом+1)

1. Ло, Гомо = min(Л, Гомо+1)

(Ло, Гом+1)

(Л, Гом)

1. Л, Гомон = min(« », Гомон) = 5

(Л, Гомо+1)

(«», Гомо) = 4

1. Л, Гомо = min («», Гомо) = 4

(Л, Гом+1)

(«», Гом) = 3

1. Лом, Гом = min(Ло, Гом+1)

(Лом, Го+1)

(Ло, Го)

1. Ло, Гом = min(Л, Гом+1)

(Ло, Го+1)

(Л, Го+1)

1. Л, Гом = min(«», Гом) = 3

(Л, Го+1)

(«», Го) = 2

1. Лом, Го = min(Ло, Го+1)

(Лом, Г+1)

(Ло, Г+1)

1. Ло, Го = min(Л, Го+1)

(Ло, Г+1)

(Л, Г)

1. Л, Го = min(«», Го) = 2

(Л, Г+1)

(«», Г) = 1

1. Лом, Г = min(Ло, Г+1)

(Лом, «») = 3

(Ло, «») = 2

1. Ло, Г = min(Л, Г+1)

(Ло, «») = 2

(Л, «») = 1

1. Л, Г = min(«», Г) = 1

(Л, «») = 1

(«», «») =0

1. Л, Г = min (2, 2, 1) = 1
2. Ло, Г = min (2, 3, 2) = 2
3. Лом, Г = min (3, 4, 3) = 3
4. Л, Го = min (3, 2, 2) = 2
5. Ло, Го = min (3, 3, 1) = 1
6. Лом, Го = min (2, 4, 3) = 2
7. Л, Гом = min (4, 3, 3) = 3
8. Ло, Гом = min (4, 2, 3) = 2
9. Лом, Гом = min (3, 3, 1) = 1
10. Л, Гомо = min (5, 4, 4) = 4
11. Л, Гомон = min (6, 5, 5) = 5
12. Ло, Гомо = min (5, 3, 4) = 3
13. Лом, Гомо = min (5, 4, 2) = 2
14. Ло, Гомон = min (6, 3, 4) = 3
15. Лом, Гомон = min (4, 3, 4) = 3

**Задание 5.**

Выполнить сравнительный анализ времени, затраченного на решение задачи о наибольшей общей подпоследовательности для двух методов решения (рекурсивное решение, динамическое программирование). Две последовательности взять с соответствии с вариантом. Построить графики зависимости времени вычисления от k. Отобразить ход решения в отчете (по примеру лекции) + код + копии экрана.

|  |  |
| --- | --- |
| Вариант 3 | |
| ABCDFGI | EATUFI |

**Решение:**

// - LCS.h

// -- рекурсивное вычисление длины LCS

int lcs ( int lenx, // длина последовательности X

const char x[], // последовательность X

int leny, // длина последовательности Y

const char y[] // последовательность Y

);

// -- динамическое вычисление LCS

int lcsd( const char x[], // последовательность X

const char y[], // последовательность Y

char z[] // наибольшая общая подпоследовательность

);

// - LCS.cpp

// -- рекурсивное вычисление длины LCS

#include "stdafx.h"

#include <algorithm>

#include "LCS.h"

int lcs (int lenx, const char x[],

int leny, const char y[])

{

int rc = 0;

if (lenx > 0 && leny > 0)

{

if (x[lenx-1] == y[leny-1]) rc = 1 + lcs(lenx-1, x,leny-1, y);

else rc = std::max(lcs(lenx, x,leny-1, y), lcs(lenx-1, x,leny, y));

}

return rc; //длина LCS

}

#define LCS\_C(x1,x2) (C[(x1)\*(leny+1)+(x2)])

#define LCS\_B(x1,x2) (B[(x1)\*(leny+1)+(x2)])

#define LCS\_X(i) (x[(i)-1])

#define LCS\_Y(i) (y[(i)-1])

#define LCS\_Z(i) (z[(i)-1])

enum Dart{TOP,LEFT,LEFTTOP};

void getLCScontent( int lenx, int leny, const char x[],

const Dart\* B,

int n, int i, int j, char z[])

{

if ((i > 0 && j > 0 && n > 0 ))

{

if (LCS\_B(i,j) == LEFTTOP)

{

getLCScontent(lenx, leny,x, B, n-1, i-1, j-1, z);

LCS\_Z(n) = LCS\_X(i);

LCS\_Z(n+1) = 0;

}

else if (LCS\_B(i,j)== TOP)

getLCScontent(lenx, leny,x, B, n, i-1, j, z);

else getLCScontent(lenx, leny,x, B, n, i, j-1, z);

}

};

int lcsd(const char x[], const char y[], char z[])

{

int n;

int lenx = strlen(x), leny = strlen(x),

\*C = new int[(lenx+1)\*(leny+1)];

Dart\* B = new Dart[(lenx+1)\*(leny+1)];

memset(C,0,sizeof(int)\*(lenx+1)\*(leny+1));

for (int i = 1; i <= lenx; i++)

for(int j = 1; j <= leny; j++)

if (LCS\_X(i) == LCS\_Y(j))

{LCS\_C(i,j) = LCS\_C(i-1,j-1)+1;

LCS\_B(i,j) = LEFTTOP;}

else if (LCS\_C(i-1,j) >= LCS\_C(i, j-1))

{

LCS\_C(i,j) = LCS\_C(i-1, j);

LCS\_B(i,j) = TOP;

}

else

{

LCS\_C(i,j) = LCS\_C(i, j-1);

LCS\_B(i,j) = LEFT;

}

getLCScontent(lenx, leny, x, B, LCS\_C(lenx,leny), lenx, leny, z);

return LCS\_C(lenx,leny);

}

#undef LCS\_Z

#undef LCS\_C

#undef LCS\_B

#undef LCS\_X

#undef LCS\_Y

// - main

// -- вычисления длины LCS

#include "stdafx.h"

#include <iostream>

#include "LCS.h"

int \_tmain(int argc, \_TCHAR\* argv[])

{

setlocale(LC\_ALL, "rus");

char z[100]="";

char X[] = "ABCDFGI";

char Y[] = "EATUFI";

std::cout<<std::endl<<"-- вычисление длины LCS для X и Y(рекурсия)";

std::cout<<std::endl<<"-- последовательность X: "<< X;

std::cout<<std::endl<<"-- последовательность Y: "<< Y;

int s = lcs(sizeof(X)-1, " BXWAFRE", sizeof(Y)-1, " XCDUFR" );

std::cout<<std::endl<< "-- длина LCS: "<<s<<std::endl;

// наибольшая общая подпоследовательность

int l = lcsd(x, y, z);

std::cout<<std::endl

<< "-- наибольшая общая подпоследовательость - LCS(динамическое"

<<"программирование)"<< std::endl;

std::cout<<std::endl<<"последовательость X: " << x;

std::cout<<std::endl<<"последовательость Y: " << x;

std::cout<<std::endl<<" LCS: " << z;

std::cout<<std::endl<<" длина LCS: " << l;

std::cout<<std::endl;

system("pause");

return 0;

}

На рисунке 3 представлена наибольшая общая подпоследовательность последовательностей Х и У, вычисленные при помощи метода динамического программирования, а также рекурсивным алгоритмом.

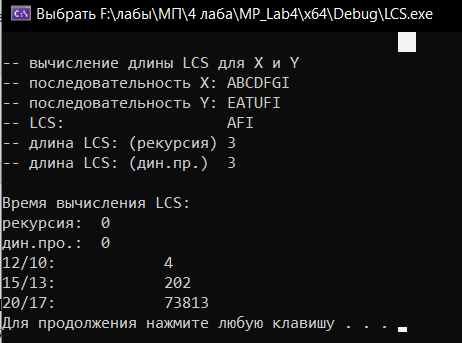


Рис. 3 – проверка работоспособности решений

На графике, представленном на рисунке 4, можно заметить, что выполненные с помощью динамического алгоритма, вычисления производятся в разы быстрее, чем с помощью рекурсивного алгоритма.

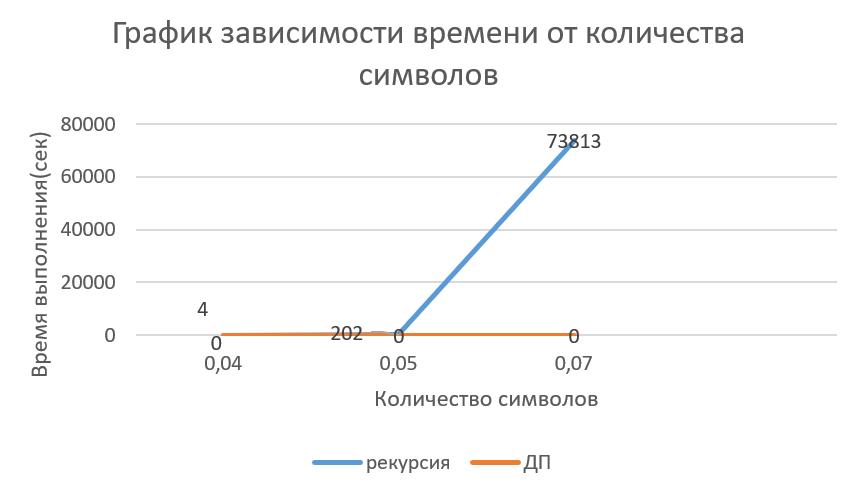


Рис. 4 – сравнительный анализ времени выполнения

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  |  | A -1 | B -2 | C - 3 | D - 4 | F - 5 | G - 6 | I - 7 |
|  | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| E -1 | 0 | ↑0 | ↑0 | ↑0 | ↑0 | ↑0 | ↑0 | ↑0 |
| A - 2 | 0 | ↖1 | ←1 | ←1 | ←1 | ←1 | ←1 | ←1 |
| T - 3 | 0 | ↑1 | ←1 | ←1 | ←1 | ←1 | ←1 | ↑1 |
| U - 4 | 0 | ↑1 | ←1 | ←1 | ←1 | ←1 | ←1 | ↑1 |
| F - 5 | 0 | ↑1 | ←1 | ←1 | ←1 | ↖2 | ←2 | ←2 |
| I - 6 | 0 | ↑1 | ←1 | ←1 | ←1 | ↑2 | ↑2 | ↖3 |

A F I

1. С 7,6 = c 6,5 +1
2. C 6,5 = max (c 5,5, c 6,4)
3. C 5,5 = c 4,4 +1
4. C 6,4 = max (c 6,3, c 5,4)
5. C 4,4 = max (c 3,4, c 4,3)
6. C 6,3 = max (c 6,2, c 5,3)
7. C 5,4 = max (c 5,3, c 4,4)
8. C 3,4 = max (c 3,3, c 2,4)
9. C 4,3 = max (c 4,2, c 3,3)
10. C 6,2 = max (c 6,1, c 5,2)
11. C 5,3 = max (c 5,2, c 4,3)
12. C 3,3 = max (c 3,2, c 2,3)
13. C 2,4 = max (c 2,3, c 1,4)
14. C 4,2 = max (c 4,1, c 3,2)
15. C 6,1 = max (c 6,0, c 5,1) = c 5,1
16. C 5,2 = max (5,1 4,2)
17. C 3,2 = max (3,1 2,2)
18. C 2,3 = max (2,2 1,3)
19. C 1,4 = max (1,3 0,4) = c 1,3
20. C 5,1 = max (5,0 4,1) = c 4,1
21. C 4,1 = max (4,0 3,1) = 3,1
22. C 3,1 = max (3,0 2,1) = c 2,1
23. C 2,2 = max (2,1 1,2)
24. C 1,3 = max (1,2 0,3) = c 1,2
25. C 2,1 = max (2,0 1,1) = c 1,1
26. C 1,2 = c 0,1 +1 = 1
27. C 1,1 = max (0,1 1,0) = 0
28. C 2,1 = 0
29. C 1,3 = 1
30. C 2,2 = max (1,0) = 1
31. C 3,1 = 0
32. C 4,1 = 0
33. C 5,1 = 0
34. C 1,4 = 1
35. C 2,3 = max (1,1) =1
36. C 3,2 = max (1, 0) = 1
37. C 5,2 = max (1, 0) = 1
38. C 6,1 = 0
39. C 4,2 = max (1, 0) = 1
40. C 2,4 = max (1, 1) = 1
41. C 33 = max (1, 1) = 1
42. C 53 = max (1, 1) = 1
43. C 62 = max (1, 0) = 1
44. C 43 = max (1, 1) = 1
45. C 34 = max (1, 1) = 1
46. C 54 = max (1, 1) = 1
47. C 63 = max (1, 1) = 1
48. C 44 = max (1, 1) = 1
49. C 64 = max (1, 1) = 1
50. C 55 = 1+1 = 2
51. C 65 = max (2, 1) = 2
52. C 76 = 2+1 = 3

**Лабораторная работа 5**

**ТРАНСПОРТНАЯ ЗАДАЧА**

**ЦЕЛЬ РАБОТЫ:** Приобретение навыков решения открытой транспортной задачи

**Ход выполнения работы**

**Задание 1.** Решить транспортную задачу. Имеется 5 поставщиков продукции и 6 потребителей. Величина запасов, потребностей и стоимость затрат на перевозку продукции взять в соответствии с вариантом (*N*). Оформить отчет.

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

1. **Проверить, открытая задача или закрытая;**

Задача является открытой, так как суммарное количество потребности груза превышает запасы поставщиков

Запасы: 171+116+153+162+103=705

Потребители: 146+110+134+196+98+116=850

Чтобы получить закрытую модель, введем дополнительную (фиктивную) базу с запасом груза, равным 145 (850-705)

1. **Составить опорный план;**

Этапы:

1. Построение начального базисного решения: метод наименьшей стоимости (минимального элемента)
2. Итеративный процесс поиска оптимального решения (метод потенциалов).

**Решение**

**Вариант 3**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| ПОТРЕБИТЕЛИ  ПОСТАВЩИКИ | 1 | 2 | 3 | 4 | 5 | 6 | ЗАПАСЫ |
| 1 | **15** | **5** | **9** | **6** | **14** | **4** | **171** |
| 2 | **13** | **3** | **11** | **8** | **10** | **16** | **116** |
| 3 | **4** | **8** | **14** | **11** | **5** | **14** | **153** |
| 4 | **7** | **13** | **13** | **6** | **16** | **5** | **162** |
| 5 | **6** | **14** | **12** | **3** | **13** | **7** | **103** |
| 6 | **0** | **0** | **0** | **0** | **0** | **0** | **145** |
| ПОТРЕБНОСТИ | **146** | **110** | **134** | **196** | **98** | **166** |  |

1. Выбор ячейки с наименьшим значением С2,2 = 3

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| ПОТРЕБИТЕЛИ  ПОСТАВЩИКИ | 1 | 2 | 3 | 4 | 5 | 6 | ЗАПАСЫ |
| 1 | **15** | **5** | **9** | **6** | **14** | **4** | **171** |
| 2 | **13** | **3** | **11** | **8** | **10** | **16** | **116** |
| 3 | **4** | **8** | **14** | **11** | **5** | **14** | **153** |
| 4 | **7** | **13** | **13** | **6** | **16** | **5** | **162** |
| 5 | **6** | **14** | **12** | **3** | **13** | **7** | **103** |
|  |  |  |  |  |  |  |  |
| ПОТРЕБНОСТИ | **146** | **110** | **134** | **196** | **98** | **166** |  |

1. Х2,2=min (110,116) = 110

Выбор ячейки с наименьшим значением С5,4 = 3

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| ПОТРЕБИТЕЛИ  ПОСТАВЩИКИ | 1 | 2 | 3 | 4 | 5 | 6 | ЗАПАСЫ |
| 1 | **15** | **5** | **9** | **6** | **14** | **4** | **171** |
| 2 | **13** | **3 | 110** | **11** | **8** | **10** | **16** | **6** |
| 3 | **4** | **8** | **14** | **11** | **5** | **14** | **153** |
| 4 | **7** | **13** | **13** | **6** | **16** | **5** | **162** |
| 5 | **6** | **14** | **12** | **3** | **13** | **7** | **103** |
|  |  |  |  |  |  |  |  |
| ПОТРЕБНОСТИ | **146** | **0** | **134** | **196** | **98** | **166** |  |

1. Х5,4=min (103,196) = 103

Выбор ячейки с наименьшим значением С1,6 = 4

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| ПОТРЕБИТЕЛИ  ПОСТАВЩИКИ | 1 | 2 | 3 | 4 | 5 | 6 | ЗАПАСЫ |
| 1 | **15** | **5** | **9** | **6** | **14** | **4** | **171** |
| 2 | **13** | **3 | 110** | **11** | **8** | **10** | **16** | **6** |
| 3 | **4** | **8** | **14** | **11** | **5** | **14** | **153** |
| 4 | **7** | **13** | **13** | **6** | **16** | **5** | **162** |
| 5 | **6** | **14** | **12** | **3|103** | **13** | **7** | **0** |
|  |  |  |  |  |  |  |  |
| ПОТРЕБНОСТИ | **146** | **0** | **134** | **93** | **98** | **166** |  |

1. Х1,6=min (171,166) = 166

Выбор ячейки с наименьшим значением С3,1 = 4

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| ПОТРЕБИТЕЛИ  ПОСТАВЩИКИ | 1 | 2 | 3 | 4 | 5 | 6 | ЗАПАСЫ |
| 1 | **15** | **5** | **9** | **6** | **14** | **4|166** | **5** |
| 2 | **13** | **3 | 110** | **11** | **8** | **10** | **16** | **6** |
| 3 | **4** | **8** | **14** | **11** | **5** | **14** | **153** |
| 4 | **7** | **13** | **13** | **6** | **16** | **5** | **162** |
| 5 | **6** | **14** | **12** | **3|103** | **13** | **7** | **0** |
|  |  |  |  |  |  |  |  |
| ПОТРЕБНОСТИ | **146** | **0** | **134** | **93** | **98** | **0** |  |

1. Х3,1=min (153,146) = 146

Выбор ячейки с наименьшим значением С3,5 = 5

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| ПОТРЕБИТЕЛИ  ПОСТАВЩИКИ | 1 | 2 | 3 | 4 | 5 | 6 | ЗАПАСЫ |
| 1 | **15** | **5** | **9** | **6** | **14** | **4|166** | **5** |
| 2 | **13** | **3 | 110** | **11** | **8** | **10** | **16** | **6** |
| 3 | **4|146** | **8** | **14** | **11** | **5** | **14** | **7** |
| 4 | **7** | **13** | **13** | **6** | **16** | **5** | **162** |
| 5 | **6** | **14** | **12** | **3|103** | **13** | **7** | **0** |
|  |  |  |  |  |  |  |  |
| ПОТРЕБНОСТИ | **0** | **0** | **134** | **93** | **98** | **0** |  |

1. Х3,5=min (7,98) = 7

Выбор ячейки с наименьшим значением С1,4 = 6

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| ПОТРЕБИТЕЛИ  ПОСТАВЩИКИ | 1 | 2 | 3 | 4 | 5 | 6 | ЗАПАСЫ |
| 1 | **15** | **5** | **9** | **6** | **14** | **4|166** | **5** |
| 2 | **13** | **3 | 110** | **11** | **8** | **10** | **16** | **6** |
| 3 | **4|146** | **8** | **14** | **11** | **5|7** | **14** | **0** |
| 4 | **7** | **13** | **13** | **6** | **16** | **5** | **162** |
| 5 | **6** | **14** | **12** | **3|103** | **13** | **7** | **0** |
|  |  |  |  |  |  |  |  |
| ПОТРЕБНОСТИ | **0** | **0** | **134** | **93** | **91** | **0** |  |

1. Х4,4=min (5,93) = 5

Выбор ячейки с наименьшим значением С4,4 = 6

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| ПОТРЕБИТЕЛИ  ПОСТАВЩИКИ | 1 | 2 | 3 | 4 | 5 | 6 | ЗАПАСЫ |
| 1 | **15** | **5** | **9** | **6|5** | **14** | **4|166** | **0** |
| 2 | **13** | **3 | 110** | **11** | **8** | **10** | **16** | **6** |
| 3 | **4|146** | **8** | **14** | **11** | **5|7** | **14** | **0** |
| 4 | **7** | **13** | **13** | **6** | **16** | **5** | **162** |
| 5 | **6** | **14** | **12** | **3|103** | **13** | **7** | **0** |
|  |  |  |  |  |  |  |  |
| ПОТРЕБНОСТИ | **0** | **0** | **134** | **88** | **91** | **0** |  |

1. Х1,3=min (162,88) = 88

Выбор ячейки с наименьшим значением С2,5 = 10

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| ПОТРЕБИТЕЛИ  ПОСТАВЩИКИ | 1 | 2 | 3 | 4 | 5 | 6 | ЗАПАСЫ |
| 1 | **15** | **5** | **9** | **6|5** | **14** | **4|166** | **0** |
| 2 | **13** | **3 | 110** | **11** | **8** | **10** | **16** | **6** |
| 3 | **4|146** | **8** | **14** | **11** | **5|7** | **14** | **0** |
| 4 | **7** | **13** | **13** | **6|88** | **16** | **5** | **74** |
| 5 | **6** | **14** | **12** | **3|103** | **13** | **7** | **0** |
|  |  |  |  |  |  |  |  |
| ПОТРЕБНОСТИ | **0** | **0** | **129** | **0** | **91** | **0** |  |

1. Х2,5=min (6,91) = 6

Выбор ячейки с наименьшим значением С4,3 = 13

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| ПОТРЕБИТЕЛИ  ПОСТАВЩИКИ | 1 | 2 | 3 | 4 | 5 | 6 | ЗАПАСЫ |
| 1 | **15** | **5** | **9** | **6|5** | **14** | **4|166** | **0** |
| 2 | **13** | **3 | 110** | **11** | **8** | **10|6** | **16** | **0** |
| 3 | **4|146** | **8** | **14** | **11** | **5|7** | **14** | **0** |
| 4 | **7** | **13** | **13** | **6|88** | **16** | **5** | **74** |
| 5 | **6** | **14** | **12** | **3|103** | **13** | **7** | **0** |
|  |  |  |  |  |  |  |  |
| ПОТРЕБНОСТИ | **0** | **0** | **129** | **0** | **85** | **0** |  |

1. Х4,3=min (69,129) = 69

Выбор ячейки с наименьшим значением

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| ПОТРЕБИТЕЛИ  ПОСТАВЩИКИ | 1 | 2 | 3 | 4 | 5 | 6 | ЗАПАСЫ |
| 1 | **15** | **5** | **9** | **6|5** | **14** | **4|166** | **0** |
| 2 | **13** | **3 | 110** | **11** | **8** | **10|6** | **16** | **0** |
| 3 | **4|146** | **8** | **14** | **11** | **5|7** | **14** | **0** |
| 4 | **7** | **13** | **13|74** | **6|88** | **16** | **5** | **0** |
| 5 | **6** | **14** | **12** | **3|103** | **13** | **7** | **0** |
| 6 | **0** | **0** | **0|60** |  | **0|85** | **0** | **145** |
| ПОТРЕБНОСТИ | **0** | **0** | **60** | **0** | **85** | **0** |  |

Должно быть m+n – 1 = 6+6-1=11 переменных

Заполнено: 11

Следовательно, опорный план является *невырожденным*.

Значение функции цели:

4\*146+3\*110+9\*5+13\*74+6\*5+6\*88+3\*103+10\*6+5\*7+4\*116 = 3502

1. **Применить метод потенциалов;**

Определяем потенциалы для всех базисных переменных

u1+v4=6; 0+v4=6; v4=6  
u4+v4=6; 6+u4=6; u4=0  
u4+v3=13; 0+v3=13; v3=13  
u6+v3=0; 13+u6=0; u6=-13  
u6+v5=0; -13+v5=0; v5=13  
u2+v5=10; 13+u2=10; u2=-3  
u2+v2=3; -3+v2=3; v2=6  
u3+v5=5; 13+u3=5; u3=-8  
u3+v1=4; -8+v1=4; v1=12  
u5+v4=3; 6+u5=3; u5=-3  
u1 + v6 = 4; 0 + v6 = 4; v6 = 4

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | v1=12 | v2=6 | v3=13 | v4=6 | v5=13 | v6=4 |
| u1=0 | 15 | 5 | 9 | 6[5] | 14 | 4[166] |
| u2=-3 | 13 | 3[110] | 11 | 8 | 10[6] | 16 |
| u3=-8 | 4[146] | 8 | 14 | 11 | 5[7] | 14 |
| u4=0 | 7 | 13 | 13[74] | 6[88] | 16 | 5 |
| u5=-3 | 6 | 14 | 12 | 3[103] | 13 | 7 |
| u6=-13 | 0 | 0 | 0[60] | 0 | 0[85] | 0 |

Опорный план не является оптимальным, так как существуют оценки свободных клеток, для которых

ui+vj>cij  
(1;2): 0+6 > 5; X12 = 0 + 6 - 5 = 1 > 0  
(1;3): 0 + 13 > 9; X13 = 0 + 13 - 9 = 4 > 0  
(4;1): 0 + 12 > 7; X41 = 0 + 12 - 7 = 5 > 0  
(5;1): -3 + 12 > 6; X51 = -3 + 12 - 6 = 3 > 0  
max(1,4,5,3) = 5  
Выбираем максимальную оценку свободной клетки (4;1): 7  
Для этого в перспективную клетку (4;1) поставим знак «+», а в остальных вершинах многоугольника чередующиеся знаки «-», «+», «-».

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| ПОТРЕБИТЕЛИ  ПОСТАВЩИКИ | 1 | 2 | 3 | 4 | 5 | 6 | Запасы |
| 1 | 15 | 5 | 9 | 6[5] | 14 | 4[166] | 171 |
| 2 | 13 | 3[110] | 11 | 8 | 10[6] | 16 | 116 |
| 3 | 4[146][-] | 8 | 14 | 11 | 5[7][+] | 14 | 153 |
| 4 | 7[+] | 13 | 13[74][-] | 6[88] | 16 | 5 | 162 |
| 5 | 6 | 14 | 12 | 3[103] | 13 | 7 | 103 |
| 6 | 0 | 0 | 0[60][+] | 0 | 0[85]-] | 0 | 145 |
| ПОТРЕБНОСТИ | 146 | 110 | 134 | 196 | 98 | 166 |  |

Цикл приведен в таблице (4,1 → 4,3 → 6,3 → 6,5 → 3,5 → 3,1).  
Из грузов хij стоящих в минусовых клетках, выбираем наименьшее, т.е. у = min (4, 3) = 74. Прибавляем 74 к объемам грузов, стоящих в плюсовых клетках и вычитаем 74 из Хij, стоящих в минусовых клетках. В результате получим новый опорный план.

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| ПОТРЕБИТЕЛИ  ПОСТАВЩИКИ | 1 | 2 | 3 | 4 | 5 | 6 | ЗАПАСЫ |
| 1 | **15** | **5** | **9** | **6|5** | **14** | **4|166** | **171** |
| 2 | **13** | **3 | 110** | **11** | **8** | **10|6** | **16** | **116** |
| 3 | **4|72** | **8** | **14** | **11** | **5|81** | **14** | **153** |
| 4 | **7|74** | **13** | **13** | **6|88** | **16** | **5** | **162** |
| 5 | **6** | **14** | **12** | **3|103** | **13** | **7** | **103** |
| 6 | **0** | **0** | **0|134** | **0** | **0|11** | **0** | **145** |
| ПОТРЕБНОСТИ | **0** | **0** | **60** | **0** | **85** | **0** |  |

Проверим оптимальность опорного плана. Найдем *предварительные потенциалы* ui, vj. по занятым клеткам таблицы, в которых ui + vj = cij, полагая, что u1 = 0.  
u1+v4=6; 0+v4=6; v4=6  
u4+v4=6; 6+u4=6; u4=0  
u4+v1=7; 0+v1=7; v1=7  
u3+v1=4; 7+u3=4; u3=-3  
u3+v5=5; -3+v5=5; v5=8  
u2+v5=10; 8+u2=10; u2=2  
u2+v2=3; 2+v2=3; v2=1  
u6+v5=0; 8+u6=0; u6=-8  
u6+v3=0; -8+v3=0; v3=8  
u5+v4=3; 6+u5=3; u5=-3  
u1 + v6 = 4; 0 + v6 = 4; v6 = 4

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | v1=7 | v2=1 | v3=8 | v4=6 | v5=8 | v6=4 |
| u1=0 | 15 | 5 | 9 | 6[5] | 14 | 4[166] |
| u2=2 | 13 | 3[110] | 11 | 8 | 10[6] | 16 |
| u3=-3 | 4[72] | 8 | 14 | 11 | 5[81] | 14 |
| u4=0 | 7[74] | 13 | 13 | 6[88] | 16 | 5 |
| u5=-3 | 6 | 14 | 12 | 3[103] | 13 | 7 |
| u6=-8 | 0 | 0 | 0[134] | 0 | 0[11] | 0 |

Опорный план является оптимальным, так все оценки свободных клеток удовлетворяют условию ui + vj ≤ cij.  
Минимальные затраты составят:

F(x) = 4\*72+7\*74+3\*110+6\*5+6\*88+3\*103+10\*6+5\*81+4\*166 = 3132

**Лабораторная работа 6. АЛГОРИТМЫ НА ГРАФАХ**

**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.





Матрица инцидентности:

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

Строки = Кол-во вершин

Столбцы = Кол-во рёбер

0 – вершина не инцидентна ребру

1 – дуга выходит из вершины

-1 – дуга входит в вершину

Матрица смежности:

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

Список смежных вершин:

0 → {1,3}

1 → {}

2 → {5}

3 → {1,2,4,6}

4 → {1,6}

5 → {6}

6 → {}

***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.

**Поиск в ширину**



Шаг 1

Посещенные вершины: {}

Очередь: {1, 3}

Текущая вершина: 0

Шаг 2

Посещенные вершины: {0}

Очередь: {3}

Текущая вершина: 1

Шаг 3

Посещенные вершины: {0, 1}

Очередь: {2,4,6}

Текущая вершина: 3

Шаг 4

Посещенные вершины: {0, 1, 3}

Очередь: {4,6,5}

Текущая вершина: 2

Шаг 5

Посещенные вершины: {0, 1, 3, 2}

Очередь: {6, 5}

Текущая вершина: 4

Шаг 6

Посещенные вершины: {0, 1, 3, 2, 4}

Очередь: {5}

Текущая вершина: 6

Шаг 7

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

Очередь: {}

Текущая вершина: 5

Шаг 8

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

Очередь: {}

Текущая вершина: -

**Поиск в глубину**

Шаг 1

Посещенные вершины: {}

Стек: {1,3}

Текущая вершина: 0

Шаг 2

Посещенные вершины: {0}

Стек: {3}

Текущая вершина: 1

Шаг 3

Посещенные вершины: {0,1}

Стек: {2,4,6}

Текущая вершина: 3

Шаг 4

Посещенные вершины: {0,1,3}

Стек: {5,4,6}

Текущая вершина: 2

Шаг 5

Посещенные вершины: {0,1,3,2}

Стек: {4,6}

Текущая вершина: 5

Шаг 5

Посещенные вершины: {0,1,3,2,5}

Стек: {6}

Текущая вершина: 4

Шаг 6

Посещенные вершины: {0,1,3,2,5,4}

Стек: {}

Текущая вершина: 6

Шаг 7

Посещенные вершины: {0,1,3,2,5,4,6}

Стек: {}

Текущая вершина: -

**Топологическая сортировка**

1. Находим какую-либо вершину, в которую не входят дуги, нумеруем ее.
2. Помечаем дуги, выходящие из помеченной вершины, как «не существующие».
3. Повторяем шаги (1) и (2), пока не будут занумерованы все вершины.

Шаг 1



Шаг 2



Шаг 3-7



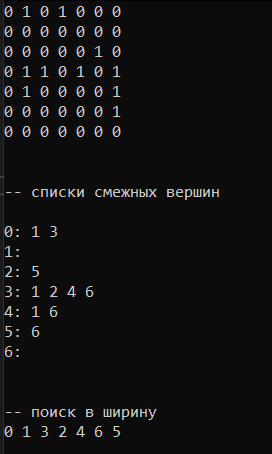
Ещё одна последовательность



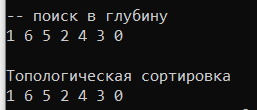
Результирующая последовательность: 6 1 4 5 2 3 0

Или: 1 6 5 2 4 3 0

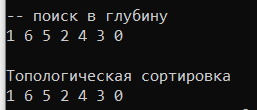
***Задание 3.*** Осуществить программную реализацию алгоритмов на C++. Разработать структуры **AMatrix** и **АList**  для представления ориентированного графа матричным и списковым способом. Разработать функции преобразования из одного способа представления в другой. Разработать функцию **BFS** обхода вершин графа, используя метод поиска в ширину. Продемонстрировать работу функции. Копии экрана вставить в отчет.



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



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



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

Веса ребер принять:

W:

W(e0,1)=8; W(e1,0)=5;

W(e0,2)=1; W(e2,0)=3;

W(e0,3)=2; W(e3,0)=8;

W(e1,3)=11; W(e3,1)=4;

W(e1,4)=5; W(e4,1)=3;

W(e2,3)=7; W(e3,2)=9;

W(e2,5)=11; W(e5,2)=10;

W(e4,3)=4; W(e3,4)=1;

W(e4,6)=10; W(e6,4)=2;

W(e5,6)=2; W(e6,5)=6;

W(e5,3)=3; W(e3,5)=6;

W(e6,3)=7; W(e3,6)=9;

Алгоритм Прима

Шаг 1



Шаг 1. Находим минимальный элемент (неориентированный граф)

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  | 0 | 1 | 2 | 3 | 4 | 5 | 6 |
| 0 | 0 | 8 | 0 | 2 | 0 | 0 | 0 |
| 1 | 8 | 0 | 0 | 4 | 3 | 0 | 0 |
| 2 | 0 | 0 | 0 | 7 | 0 | 11 | 0 |
| 3 | 2 | 4 | 7 | 0 | 1 | 0 | 9 |
| 4 | 0 | 3 | 0 | 1 | 0 | 0 | 10 |
| 5 | 0 | 0 | 11 | 0 | 0 | 0 | 2 |
| 6 | 0 | 0 | 0 | 9 | 10 | 2 | 0 |

Шаг 2. Вычеркнем 3-ю и 4-ю строки таблицы. Столбец 3 и 4 выделим. Найдём минимальный элемент в столбце

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  | 0 | 1 | 2 | 3 | 4 | 5 | 6 |
| 0 | 0 | 8 | 0 | 2 | 0 | 0 | 0 |
| 1 | 8 | 0 | 0 | 4 | 3 | 0 | 0 |
| 2 | 0 | 0 | 0 | 7 | 0 | 11 | 0 |
| 3 |  |  |  |  | 1 |  |  |
| 4 |  |  |  |  |  |  |  |
| 5 | 0 | 0 | 11 | 0 | 0 | 0 | 2 |
| 6 | 0 | 0 | 0 | 9 | 10 | 2 | 0 |

Шаг 3. Вычеркнем 0-ю строки таблицы. Столбец 0 выделим. Найдём минимальный элемент в столбце

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  | 0 | 1 | 2 | 3 | 4 | 5 | 6 |
| 0 |  |  |  | 2 |  |  |  |
| 1 | 8 | 0 | 0 | 4 | 3 | 0 | 0 |
| 2 | 0 | 0 | 0 | 7 | 0 | 11 | 0 |
| 3 |  |  |  |  | 1 |  |  |
| 4 |  |  |  |  |  |  |  |
| 5 | 0 | 0 | 11 | 0 | 0 | 0 | 2 |
| 6 | 0 | 0 | 0 | 9 | 10 | 2 | 0 |

Шаг 4. Вычеркнем 1-ю строки таблицы. Столбец 1 выделим. Найдём минимальный элемент в столбце

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  | 0 | 1 | 2 | 3 | 4 | 5 | 6 |
| 0 |  |  |  | 2 |  |  |  |
| 1 |  |  |  |  | 3 |  |  |
| 2 | 0 | 0 | 0 | 7 | 0 | 11 | 0 |
| 3 |  |  |  |  | 1 |  |  |
| 4 |  |  |  |  |  |  |  |
| 5 | 0 | 0 | 11 | 0 | 0 | 0 | 2 |
| 6 | 0 | 0 | 0 | 9 | 10 | 2 | 0 |

Шаг 4. Вычеркнем 2-ю строки таблицы. Столбец 2 выделим. Найдём минимальный элемент в выделенных столбцах

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  | 0 | 1 | 2 | 3 | 4 | 5 | 6 |
| 0 |  |  |  | 2 |  |  |  |
| 1 |  |  |  |  | 3 |  |  |
| 2 |  |  |  | 7 |  |  |  |
| 3 |  |  |  |  | 1 |  |  |
| 4 |  |  |  |  |  |  |  |
| 5 | 0 | 0 | 11 | 0 | 0 | 0 | 2 |
| 6 | 0 | 0 | 0 | 9 | 10 | 2 | 0 |

Шаг 5. Вычеркнем 6-ю строки таблицы. Столбец 6 выделим. Найдём минимальный элемент в выделенных столбцах

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  | 0 | 1 | 2 | 3 | 4 | 5 | 6 |
| 0 |  |  |  | 2 |  |  |  |
| 1 |  |  |  |  | 3 |  |  |
| 2 |  |  |  | 7 |  |  |  |
| 3 |  |  |  |  | 1 |  |  |
| 4 |  |  |  |  |  |  |  |
| 5 | 0 | 0 | 11 | 0 | 0 | 0 | 2 |
| 6 |  |  |  | 9 |  |  |  |

Шаг 6. Вычеркнем 5-ю строки таблицы. Столбец 5 выделим. Найдём минимальный элемент в выделенных столбцах

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

Минимальное остовное дерево. Вес: 24

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

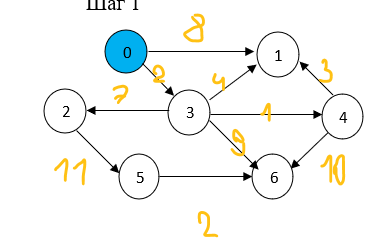
**Алгоритм Краскала**

Шаг 1

Шаг 2

Шаг 3

Шаг 4



Шаг 5

Шаг 5

**Лабораторная работа 7. Сетевые модели**

**Цель работы:** Приобретение навыков сетевого планирования и составления сетевых графиков, приобретение опыта нахождения критического пути.

**Задание для выполнения:**

Лабораторная работа базируется на исследовании различных тематик в проектировании программных продуктов, составлении сетевых графиков для разных тем, нахождении критических путей в составленных графиках. Каждый проект принять условным или обобщенным, но допустимо делать упор на конкретные примеры.

|  |  |  |
| --- | --- | --- |
| Вариант | Проект для исследования | Время выполнения всех задач |
| Вариант 1, 7, 13 | «Создание десктопного приложения» | 65 дней |
| Вариант 2, 8, 14 | «Создание мобильной игры» | 50 дней |
| Вариант 3, 9, 15 | «Создание банковского приложения» | 65 дней |
| Вариант 4, 10, 16 | «Создание облачного хранилища» | 60 дней |
| Вариант 5, 11, 17 | «Создание веб-приложения» | 70 дней |
| Вариант 6, 12, 18 | «Создание компьютерной игры» | 90 дней |

**Задание 1. Структурное 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.

**Задание 2. Календарное планирование.**

Распределите время, отпущенное на ваш проект согласно вариантам, на выделенные вами этапы. Скорректируйте сформулированные вами задачи, если это необходимо.

Вариант 3. «Создание банковского приложения» Время выполнения всех задач – 65 дней

|  |  |  |  |
| --- | --- | --- | --- |
| Код операции | Наименование операции | Предшествующие операции | t |
| I. АНАЛИЗ | | | |
| Z1 | Системный анализ |  | 10 |
| Z2 | Анализ требований | Z1 | 5 |
| II. ПРОЕКТИРОВАНИЕ | | | |
| Z3 | Проектирование базы данных | Z2 | 8 |
| Z4 | Проектирование архитектуры приложения | Z2 | 7 |
| Z5 | Проектирование пользовательского интерфейса | Z2 | 6 |
| III. КОДИРОВАНИЕ | | | |
| Z6 | Разработка серверной части приложения | Z3, Z4 | 12 |
| Z7 | Разработка клиентской части приложения | Z5 | 10 |
| Z8 | Реализация функций работы с банковскими данными | Z6 | 9 |
| IV. ТЕСТИРОВАНИЕ | | | |
| Z9 | Функциональное тестирование | Z6, Z7, Z8, Z15, Z16, Z17 | 8 |
| Z10 | Структурное тестирование | Z6, Z7, Z8, Z15, Z16, Z17 | 8 |
| V. ВНЕДРЕНИЕ | | | |
| Z11 | Разработка документации | Z6, Z7, Z8, Z9 | 6 |
| Z12 | Установка и конфигурирование приложения на сервере | Z11 | 5 |
| Z13 | Тестирование после внедрения | Z11, Z12 | 6 |
| Z14 | Обучение пользователей | Z11, Z12, Z13 | 4 |
| VI. ДОПОЛНИТЕЛЬНЫЕ РАБОТЫ | | | |
| Z15 | Создание дизайна и брендинга | Z5 | 4 |
| Z16 | Обеспечение безопасности приложения | Z6 | 7 |
| Z17 | Резервное копирование и восстановление данных | Z6 | 4 |

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

|  |  |  |  |
| --- | --- | --- | --- |
| Начальное событие | Код операции | Предшествующие операции | Конечное событие |
| 0 | Z1 |  | 1 |
| 2 | Z2 | Z1, | 3 |
| 4 | Z3 | Z2 | 5 |
| 6 | Z4 | Z2 | 7 |
| 8 | Z5 | Z2 | 9 |
| 10 | Z6 | Z3, Z4 | 11 |
| 12 | Z7 | Z5 | 13 |
| 14 | Z8 | Z6 | 15 |
| 16 | Z9 | Z6, Z7, Z8 | 17 |
| 18 | Z10 | Z6, Z7, Z8 | 19 |
| 20 | Z11 | Z6, Z7, Z8, Z9 | 21 |
| 22 | Z12 | Z11 | 23 |
| 24 | Z13 | Z12 | 25 |
| 26 | Z14 | Z11, Z12 | 27 |
| 28 | Z15 | Z5 | 29 |
| 30 | Z16 | Z6 | 31 |
| 32 | Z17 | Z6 | 33 |

<https://www.figma.com/file/Zr3qfyf89Afx2ewDDI9LXp/%D0%A0%D0%B8%D0%90%D0%A2-%D0%BB%D0%B0%D0%B14?type=design&node-id=0%3A1&t=ua4xIiGgyzgqEEVY-1>