МИНИСТЕРСТВО ОБРАЗОВАНИЯ РЕСПУБЛИКИ БЕЛАРУСЬ

Учреждение образования «БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ

ТЕХНОЛОГИЧЕСКИЙ УНИВЕРСИТЕТ»

Факультет Информационных технологий

Кафедра Программной инженерии

Специальность 1-40 01 01 Программное обеспечение информационных технологий

**Отчёт**

К лабораторным работам

Выполнил:  
студент II курса 6 группы   
специальности ПОИТ Ляшонок М.М.

Преподаватель:  
Бурмакова Анастасия Владимировна

Минск 2024

Лабораторная работа №1 “Вспомогательные функции”

Линейная

Полиномиальная в степени 4

Код:

|  |
| --- |
| fibonnachi.h  #pragma once  int fibonacci(int n) {  if (n <= 1)  return n;  else  return fibonacci(n - 1) + fibonacci(n - 2);  }  Auxil.h  #pragma once  #include <cstdlib>  namespace auxil  {  void start(); // старт генератора сл. чисел  double dget(double rmin, double rmax); // получить случайное число  int iget(int rmin, int rmax); // получить случайное число  };  stdafx.h  #pragma once  #include <iostream>  #include <stdio.h>  #include <ctime>  #include <locale>  #include <cstdlib>  main.cpp  #include "stdafx.h"  #include "Auxil.h" // вспомогательные функции  using namespace auxil;  #define CYCLE 1000000 // количество циклов  int main(int argc, char\* argv[])  {  double av1 = 0, av2 = 0;  clock\_t t1 = 0, t2 = 0;  setlocale(LC\_ALL, "rus");  auxil::start(); // старт генерации  t1 = clock(); // фиксация времени  for (int i = 0; i < CYCLE; i++)  {  av1 += (double)auxil::iget(-100, 100); // сумма случайных чисел  av2 += auxil::dget(-100, 100); // сумма случайных чисел  }  t2 = clock(); // фиксация времени  std::cout << std::endl << "количество циклов: " << CYCLE;  std::cout << std::endl << "среднее значение (int): " << av1 / CYCLE;  std::cout << std::endl << "среднее значение (double): " << av2 / CYCLE;  std::cout << std::endl << "продолжительность (у.е): " << (t2 - t1);  std::cout << std::endl << " (сек): "  << ((double)(t2 - t1)) / ((double)CLOCKS\_PER\_SEC);  std::cout << std::endl;  system("pause");  return 0;  }  Auxil.cpp  #include "stdafx.h"  #include "Auxil.h"  namespace auxil  {  void start() // старт генератора сл. чисел  {  srand((unsigned)time(NULL));  };  double dget(double rmin, double rmax) // получить случайное число  {  return ((double)rand() / (double)RAND\_MAX) \* (rmax - rmin) + rmin;  };  int iget(int rmin, int rmax) // получить случайное число  {  return (int)dget((double)rmin, (double)rmax);  };  }  ubung3.cpp  #include "stdafx.h"  #include <vector>  #include "fibonnachi.h"  #include "Auxil.h"  int main(int argc, char\* argv[])  {  setlocale(LC\_ALL, "rus");  int res = 0;  std::vector<int> tests = { 20, 23, 25,28 ,30,35, 40 };  for (auto i : tests)  {  std::cout << "попытка " << i << std::endl;  clock\_t start = clock();  res = fibonacci(i);  std::cout << "результат " << res << std::endl;  clock\_t finish = clock();  std::cout << "Время (у.е.)" << finish - start << std::endl;  std::cout << "------------------------------------------------" << std::endl;  }  system("pause");  return 0;  } |

Вывод

На основании полученных данных можно сделать вывод, что сложность операции суммы чисел является линейной, а подсчёт чисел Фибоначчи имеет полиномиальную зависимость четвёртой степени

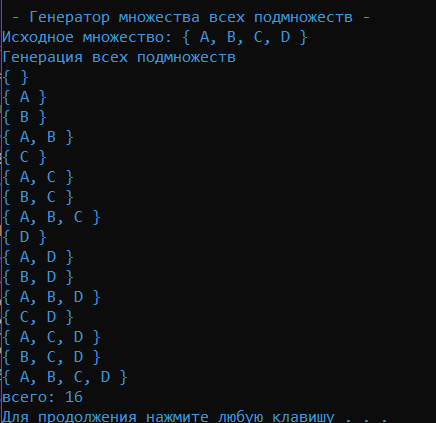
Лабораторная работа №2

Цель работы: изучить генераторы множеств, сочетаний, перестановок, размещений.

Задание 1:

|  |
| --- |
| int main(int argc, char\* argv[])  {  setlocale(LC\_ALL, "rus");  char AA[][2] = { "A", "B", "C", "D" };  std::cout << std::endl << " - Генератор множества всех подмножеств -";  std::cout << std::endl << "Исходное множество: ";  std::cout << "{ ";  for (int i = 0; i < sizeof(AA) / 2; i++)  std::cout << AA[i] << ((i < sizeof(AA) / 2 - 1) ? ", " : " ");  std::cout << "}";  std::cout << std::endl << "Генерация всех подмножеств ";  combi::subset s1(sizeof(AA) / 2); // создание генератора  int n = s1.getfirst(); // первое (пустое) подмножество  while (n >= 0) // пока есть подмножества  {  std::cout << std::endl << "{ ";  for (int i = 0; i < n; i++)  std::cout << AA[s1.ntx(i)] << ((i < n - 1) ? ", " : " ");  std::cout << "}";  n = s1.getnext(); // cледующее подмножество  };  std::cout << std::endl << "всего: " << s1.count() << std::endl;  system("pause");  return 0;  } |

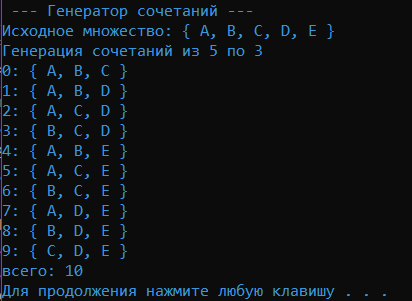
Результат выполнения:



Задание 2:

|  |
| --- |
| #include <iostream>  #include "Combi.h"  int main(int argc, char\* argv[])  {  setlocale(LC\_ALL, "rus");  char AA[][2] = { "A", "B", "C", "D", "E" };  std::cout << std::endl << " --- Генератор сочетаний ---";  std::cout << std::endl << "Исходное множество: ";  std::cout << "{ ";  for (int i = 0; i < sizeof(AA) / 2; i++)  std::cout << AA[i] << ((i < sizeof(AA) / 2 - 1) ? ", " : " ");  std::cout << "}";  std::cout << std::endl << "Генерация сочетаний ";  combi::xcombination xc(sizeof(AA) / 2, 3);  std::cout << "из " << xc.n << " по " << xc.m;  int n = xc.getfirst();  while (n >= 0)  {  std::cout << std::endl << xc.nc << ": { ";  for (int i = 0; i < n; i++)  std::cout << AA[xc.ntx(i)] << ((i < n - 1) ? ", " : " ");  std::cout << "}";  n = xc.getnext();  };  std::cout << std::endl << "всего: " << xc.count() << std::endl;  system("pause");  return 0;  } |

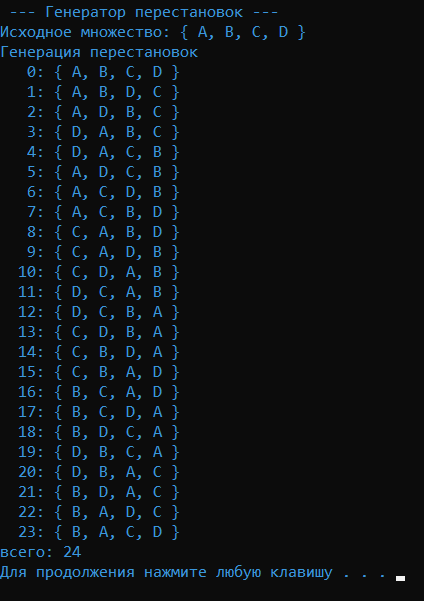
Результат выполнения:



Задание 3:

|  |
| --- |
| #include <iostream>  #include "Combi.h"  #include <iomanip>  int main(int argc, char\* argv[])  {  setlocale(LC\_ALL, "rus");  char AA[][2] = { "A", "B", "C", "D" };  std::cout << std::endl << " --- Генератор перестановок ---";  std::cout << std::endl << "Исходное множество: ";  std::cout << "{ ";  for (int i = 0; i < sizeof(AA) / 2; i++)  std::cout << AA[i] << ((i < sizeof(AA) / 2 - 1) ? ", " : " ");  std::cout << "}";  std::cout << std::endl << "Генерация перестановок ";  combi::permutation p(sizeof(AA) / 2);  \_\_int64 n = p.getfirst();  while (n >= 0)  {  std::cout << std::endl << std::setw(4) << p.np << ": { ";  for (int i = 0; i < p.n; i++)  std::cout << AA[p.ntx(i)] << ((i < p.n - 1) ? ", " : " ");  std::cout << "}";  n = p.getnext();  };  std::cout << std::endl << "всего: " << p.count() << std::endl;  system("pause");  return 0;  } |

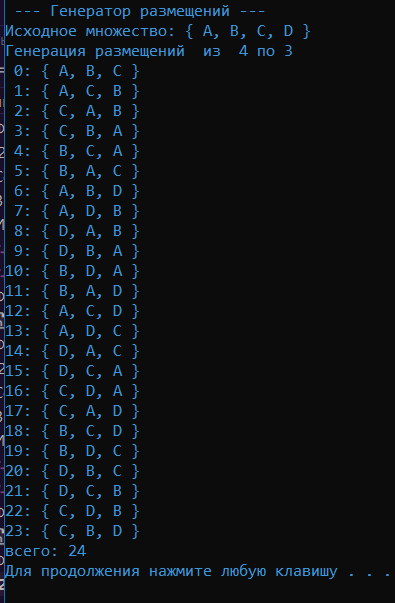
Результат работы:



Задание 4:

|  |
| --- |
| #include <iostream>  #include <iomanip>  #include "Combi.h"  #define N (sizeof(AA)/2)  #define M 3  int main(int argc, char\* argv[])  {  setlocale(LC\_ALL, "rus");  char AA[][2] = { "A", "B", "C", "D" };  std::cout << std::endl << " --- Генератор размещений ---";  std::cout << std::endl << "Исходное множество: ";  std::cout << "{ ";  for (int i = 0; i < N; i++)  std::cout << AA[i] << ((i < N - 1) ? ", " : " ");  std::cout << "}";  std::cout << std::endl << "Генерация размещений из " << N << " по " << M;  combi::accomodation s(N, M);  int n = s.getfirst();  while (n >= 0)  {  std::cout << std::endl << std::setw(2) << s.na << ": { ";  for (int i = 0; i < 3; i++)  std::cout << AA[s.ntx(i)] << ((i < n - 1) ? ", " : " ");  std::cout << "}";  n = s.getnext();  };  std::cout << std::endl << "всего: " << s.count() << std::endl;  system("pause");  return 0;  } |

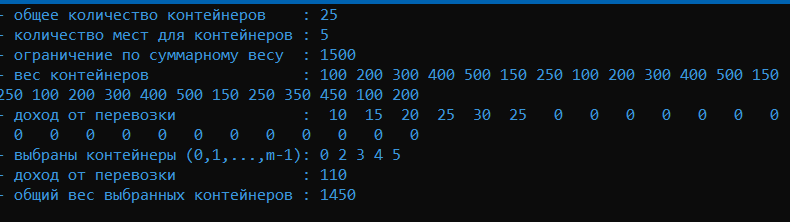
Результат работы:



Задание 5:

|  |
| --- |
| #include <iostream>  #include <iomanip>  #include "Boat.h"  #define NN (sizeof(v)/sizeof(int))  #define MM 5  int wmain()  {  setlocale(LC\_ALL, "rus");  int V = 1500,  v[] = { 100, 200, 300, 400, 500, 150, 250, 100,  200, 300, 400, 500, 150, 250, 100, 200,  300, 400, 500, 150, 250, 350, 450, 100, 200 },  c[NN] = { 10, 15, 20, 25, 30, 25 };  short r[MM];  int cc = boat(  V, // [in] максимальный вес груза  MM, // [in] количество мест для контейнеров  NN, // [in] всего контейнеров  v, // [in] вес каждого контейнера  c, // [in] доход от перевозки каждого контейнера  r // [out] результат: индексы выбранных контейнеров  );  std::cout << std::endl << "- Задача о размещении контейнеров на судне";  std::cout << std::endl << "- общее количество контейнеров : " << NN;  std::cout << std::endl << "- количество мест для контейнеров : " << MM;  std::cout << std::endl << "- ограничение по суммарному весу : " << V;  std::cout << std::endl << "- вес контейнеров : ";  for (int i = 0; i < NN; i++) std::cout << std::setw(3) << v[i] << " ";  std::cout << std::endl << "- доход от перевозки : ";  for (int i = 0; i < NN; i++) std::cout << std::setw(3) << c[i] << " ";  std::cout << std::endl << "- выбраны контейнеры (0,1,...,m-1): ";  for (int i = 0; i < MM; i++) std::cout << r[i] << " ";  std::cout << std::endl << "- доход от перевозки : " << cc;  std::cout << std::endl << "- общий вес выбранных контейнеров : ";  int s = 0; for (int i = 0; i < MM; i++) s += v[r[i]]; std::cout << s;  std::cout << std::endl << std::endl;  system("pause");  return 0;  } |

Результат работы:



Задание 6:

|  |
| --- |
| #include <iostream>  #include <iomanip>  #include "Boat.h"  #include <time.h>  #define NN (sizeof(v)/sizeof(int))  #define MM 6  #define SPACE(n) std::setw(n)<<" "  int main()  {  setlocale(LC\_ALL, "rus");  int V = 1500,  v[] = { 250, 560, 670, 400, 200, 270, 370, 330, 330, 440, 530, 120,  200, 270, 370, 330, 330, 440, 700, 120, 550, 540, 420, 170,  600, 700, 120, 550, 540, 420, 430, 140, 300, 370, 310 };  int c[NN] = { 15, 26, 27, 43, 16, 26, 42, 22, 34, 12, 33,  42, 22, 34, 43, 16, 26, 14, 12, 25, 41, 17, 28,  12, 45, 60, 41, 33, 11, 14, 12, 25, 41, 30, 40 };  short r[MM];  int maxcc = 0;  clock\_t t1, t2;  std::cout << std::endl << "-- Задача об оптимальной загрузке судна -- ";  std::cout << std::endl << "- ограничение по весу : " << V;  std::cout << std::endl << "- количество мест : " << MM;  std::cout << std::endl << "-- количество ------ продолжительность -- ";  std::cout << std::endl << " контейнеров вычисления ";  for (int i = 25; i <= NN; i++)  {  t1 = clock();  int maxcc = boat(V, MM, i, v, c, r);  t2 = clock();  std::cout << std::endl << SPACE(7) << std::setw(2) << i  << SPACE(15) << std::setw(5) << (t2 - t1);  }  std::cout << std::endl << std::endl;  system("pause");  return 0;  } |

Результат работы:

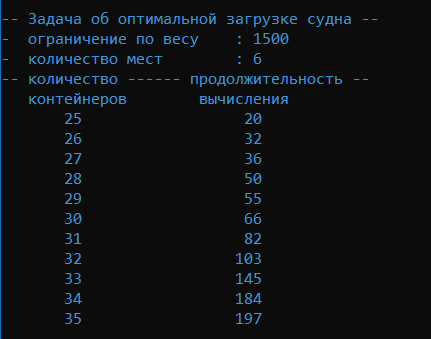
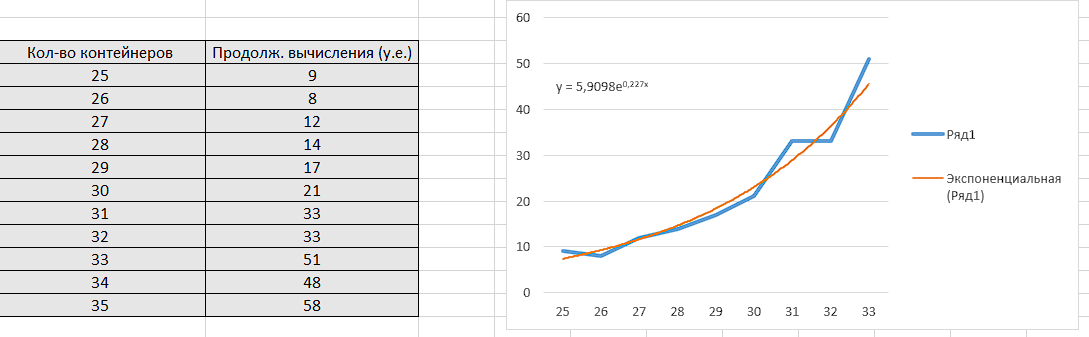


График:



Вывод: для решения этой задачи можно использовать генератор сочетаний- он позволит получить все возможные варианты распределения грузов по трюмам. Далее из всех вариантов можно выбрать оптимальный - тот, который удовлетворяет ограничениям и имеет максимальную суммарную стоимость грузов.

Лабораторная работа №3

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.

**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 10:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Город** | **1** | **2** | **3** | **4** | **5** |
| **1** |  | 20 | 31 |  | 10 |
| **2** | 10 |  | 25 | 58 | 74 |
| **3** | 12 | 30 |  | 86 | 59 |
| **4** | 27 | 48 | 40 |  | 30 |
| **5** | 83 | 76 | 52 | 23 |  |

**Задание 2.**

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

**Решение**

Приведённая по строкам матрица:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Город** | **1** | **2** | **3** | **4** | **5** |  |
| **1** |  | 10 | 21 |  | 0 | 10 |
| **2** | 0 |  | 15 | 48 | 64 | 10 |
| **3** | 0 | 18 |  | 74 | 47 | 12 |
| **4** | 0 | 21 | 13 |  | 3 | 27 |
| **5** | 60 | 53 | 29 | 0 |  | 23 |

α =10+10+12+27+23=82

Полностью приведённая матрица:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Город** | **1** | **2** | **3** | **4** | **5** |  |
| **1** |  | 0 | 11 |  | 0 | 10 |
| **2** | 0 |  | 2 | 58 | 74 | 10 |
| **3** | 0 | 8 |  | 74 | 47 | 12 |
| **4** | 0 | 11 | 0 |  | 3 | 27 |
| **5** | 60 | 43 | 19 | 0 |  | 23 |
|  | 0 | 10 | 13 | 0 | 0 | 10 |

β = 10+13=23

Нижняя граница длины кольцевого маршрута:

φ = 82+20=105

Сумма констант приведения матрицы при замене значения 0 на INF (бесконечность) в соответствующих ячейках таблицы:

01,2 = 8; 02,1 = 2; 03,1 = 8; 04,3 = 2; 05,4 = 77;

01,5 = 3; 04,1 =0;

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

182

105

В случае если мы идём по маршруту (5, 4) то расстояние будет равно нижней границе кольцевого маршрута, то есть 105, а если не пойдём, то расстояние будет равно 77 + 105= 182.

Так как меньшее расстояние 102, 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 | 11 | 0 |
| **2** | 0 |  | 2 | 74 |
| **3** | 0 | 8 |  | 47 |
| **4** | 0 | 11 | 0 |  |

01,2 = 8; 02,1 = 2; 03,1 = 8; 04,1 = 0;

01,5 = 58; 04,3=2;

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

160

105

В случае если мы идём по маршруту (1, 5) то расстояние будет равно нижней границе кольцевого маршрута, то есть 102, а если не пойдём, то расстояние будет равно 102+ 58 = 160.

Так как меньшее расстояние 102, 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 |  | 2 |
| **3** | 0 | 8 |  |
| **4** | 0 | 11 | 0 |

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

|  |  |  |  |
| --- | --- | --- | --- |
| **Город** | **1** | **2** | **3** |
| **2** | 0 |  | 10 |
| **3** | 0 | 0 |  |
| **4** | 0 | 3 | 0 |

Данная матрица не является полностью приведённой, поэтому её надо привести по

β = 0 + 8 + 0 = 8;

Следовательно, изменится нижняя граница кольцевого маршрута и соответственно:

φ =105+8=113

02,1 = 10; 03,1 = 0; 04,3 = 10; 04,1 =0; 03,2 = 3 ;

03,2 = 0;

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

113

123

Так как меньшее расстояние 113, 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 минимальная длина маршрута будет равно 113, и граф будет иметь следующий вид:

105

113

105

182

123

152

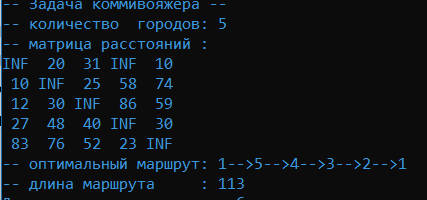
R3,2

R2,1

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

Длина маршрута 113.

Проверка вывода



Использованный код:

Main:

|  |
| --- |
| // Combi.cpp  #include "stdafx.h"  #include "Combi.h"  #include <algorithm>  #define NINF ((short)0x8000)  namespace combi  {  permutation::permutation(short n)  {  this->n = n;  this->sset = new short[n];  this->dart = new bool[n];  this->reset();  };  void permutation::reset()  {  this->getfirst();  };  \_\_int64 permutation::getfirst()  {  this->np = 0;  for (int i = 0; i < this->n; i++)  {  this->sset[i] = i; this->dart[i] = L;  };  return (this->n > 0) ? this->np : -1;  };  \_\_int64 permutation::getnext() //  {  \_\_int64 rc = -1;  short maxm = NINF, idx = -1;  for (int i = 0; i < this->n; i++)  {  if (i > 0 &&  this->dart[i] == L &&  this->sset[i] > this->sset[i - 1] &&  maxm < this->sset[i]) maxm = this->sset[idx = i];  if (i < (this->n - 1) &&  this->dart[i] == R &&  this->sset[i] > this->sset[i + 1] &&  maxm < this->sset[i]) maxm = this->sset[idx = i];  };  if (idx >= 0)  {  std::swap(this->sset[idx],  this->sset[idx + (this->dart[idx] == L ? -1 : 1)]);  std::swap(this->dart[idx],  this->dart[idx + (this->dart[idx] == L ? -1 : 1)]);  for (int i = 0; i < this->n; i++)  if (this->sset[i] > maxm) this->dart[i] = !this->dart[i];  rc = ++this->np;  }  return rc;  };  short permutation::ntx(short i) { return this->sset[i]; };  unsigned \_\_int64 fact(unsigned \_\_int64 x) { return (x == 0) ? 1 : (x \* fact(x - 1)); };  unsigned \_\_int64 permutation::count() const { return fact(this->n); };  } |

Salesman.h

|  |
| --- |
| #pragma once  #define INF 0x7fffffff // бесконечность  #include "Combi.h"  int salesman( // функция возвращает длину оптимального маршрута  int n, // [in] количество городов  const int\* d, // [in] массив [n\*n] расстояний  int\* r // [out] массив [n] маршрут 0 x x x x  ); |

Salesman.cpp

|  |
| --- |
| #include "stdafx.h"  #include "Salesman.h"  int sum(int x1, int x2) // суммирование с учетом бесконечности  {  return (x1 == INF || x2 == INF) ? INF : (x1 + x2);  };  int\* firstpath(int n) // формирование 1го маршрута 0,1,2,..., n-1, 0  {  int\* rc = new int[n + 1];  rc[n] = 0;  for (int i = 0; i < n; i++)  rc[i] = i;  return rc;  };  int\* source(int n) // формирование исходного массива 1,2,..., n-1  {  int\* rc = new int[n - 1];  for (int i = 1; i < n; i++) rc[i - 1] = i;  return rc;  };  void copypath(int n, int\* r1, const int\* r2) // копировать маршрут  {  for (int i = 0; i < n; i++)  r1[i] = r2[i];  };  int distance(int n, int\* r, const int\* d) // длина маршрута  {  int rc = 0;  for (int i = 0; i < n - 1; i++)  rc = sum(rc, d[r[i] \* n + r[i + 1]]);  return sum(rc, d[r[n - 1] \* n + 0]); //+ последн¤¤ дуга (n-1,0)  };  void indx(int n, int\* r, const int\* s, const short\* ntx)  {  for (int i = 1; i < n; i++)  r[i] = s[ntx[i - 1]];  }  int salesman(  int n, // [in] количество городов  const int\* d, // [in] массив [n\*n] рассто¤ний  int\* r // [out] массив [n] маршрут 0 x x x x  )  {  int\* s = source(n), \* b = firstpath(n), rc = INF, dist = 0;//формирует из н городов рц-служебна¤ переменна¤  combi::permutation p(n - 1);  int k = p.getfirst();  while (k >= 0) // цикл генерации перестановок  {  indx(n, b, s, p.sset); // новый маршрут  if ((dist = distance(n, b, d)) < rc)  {  rc = dist; copypath(n, r, b);  }  k = p.getnext();  };  return rc;  } |

Stadfx.h

|  |
| --- |
| #pragma once  #include <stdio.h>  #include <tchar.h> |

Combi.h

|  |
| --- |
| // Combi.h  #pragma once  namespace combi  {  struct permutation // генератор перестановок  {  const static bool L = true; // левая стрелка  const static bool R = false; // правая стрелка  short n, // количество элементов исходного множества  \* sset; // массив индексов текущей перестановки  bool\* dart; // массив стрелок (левых-L и правых-R)  permutation(short n = 1); // конструктор (количество элементов исходного множества)  void reset(); // сбросить генератор, начать сначала  \_\_int64 getfirst(); // сформировать первый массив индексов  \_\_int64 getnext(); // сформировать случайный массив индексов  short ntx(short i); // получить i-й элемент масива индексов  unsigned \_\_int64 np; // номер перествновки 0,... count()-1  unsigned \_\_int64 count() const; // вычислить общее кол. перестановок  };  }; |

Combi.cpp

|  |
| --- |
| // Combi.cpp  #include "stdafx.h"  #include "Combi.h"  #include <algorithm>  #define NINF ((short)0x8000)  namespace combi  {  permutation::permutation(short n)  {  this->n = n;  this->sset = new short[n];  this->dart = new bool[n];  this->reset();  };  void permutation::reset()  {  this->getfirst();  };  \_\_int64 permutation::getfirst()  {  this->np = 0;  for (int i = 0; i < this->n; i++)  {  this->sset[i] = i; this->dart[i] = L;  };  return (this->n > 0) ? this->np : -1;  };  \_\_int64 permutation::getnext() //  {  \_\_int64 rc = -1;  short maxm = NINF, idx = -1;  for (int i = 0; i < this->n; i++)  {  if (i > 0 &&  this->dart[i] == L &&  this->sset[i] > this->sset[i - 1] &&  maxm < this->sset[i]) maxm = this->sset[idx = i];  if (i < (this->n - 1) &&  this->dart[i] == R &&  this->sset[i] > this->sset[i + 1] &&  maxm < this->sset[i]) maxm = this->sset[idx = i];  };  if (idx >= 0)  {  std::swap(this->sset[idx],  this->sset[idx + (this->dart[idx] == L ? -1 : 1)]);  std::swap(this->dart[idx],  this->dart[idx + (this->dart[idx] == L ? -1 : 1)]);  for (int i = 0; i < this->n; i++)  if (this->sset[i] > maxm) this->dart[i] = !this->dart[i];  rc = ++this->np;  }  return rc;  };  short permutation::ntx(short i) { return this->sset[i]; };  unsigned \_\_int64 fact(unsigned \_\_int64 x) { return (x == 0) ? 1 : (x \* fact(x - 1)); };  unsigned \_\_int64 permutation::count() const { return fact(this->n); };  } |

Лабораторная работа №4

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

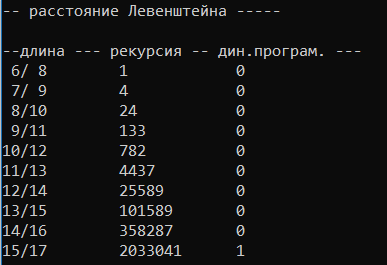
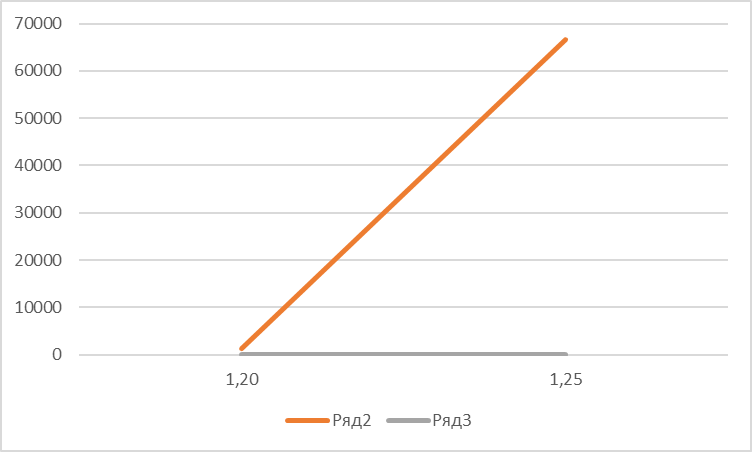
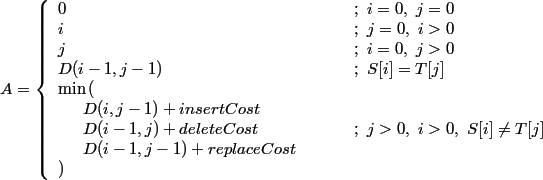


График зависимости расчёта расстояния Левенштейна для рекурсивного метода и динамического программирования.



Задание 4



Решение:

1. L(«Кол», «Столб») = min

2. L(«Ко», «Столб») = min

3. L(«Кол», «Стол») = min

4. L(«Ко», «Стол») = min

5. L(«К», «Столб») = min

L(«», «Столб») = 5,

L(«», «Стол») = 4

6. L(«К», «Стол») = min

L(«», «Сто») = 3

7. L(«Кол», «Сто») = min

8. L(«Ко», «Сто») = min

9. L(«К», «Сто») = min

L(«», «Ст») = 2

10. L(«Кол», «Ст») = min

11. L(«Ко», «Ст») = min

12. L(«К», «Ст») = min

L(«», «С») = 1

13. L(«Кол», «С») = min

L(«Кол», «») = 3

14. L(«Ко», «С») = min

L(«Ко», «») = 2,

L(«К», «») = 1

15. L(«К», «С») = min

16. L(«Ко», «С») = min (2, 3, 2) = 2

17. L(«Кол», «С») = min (4, 4, 2) = 2

18. L(«К», «Ст») = min (3, 2, 2) = 2

19. L(«Ко», «Ст») = min (4, 4, 2) = 2

20. L(«Кол», «Сто») = min (3, 4, 3) = 3

21. L(«К», «Сто») = min (4, 3, 3) = 3

22. L(«Ко», «Сто») = min (4, 3, 3) = 3

24. L(«К», «Стол») = min (5, 4, 4) = 4

25. L(«К», «Столб») = min (6, 5, 5) = 5

26. L(«Ко», «Стол») = min (5, 4, 3) = 3

27. L(«Кол», «Стол») = min (5, 3, 3) = 3

28. L(«Ко», «Столб») = min (6, 4, 5) = 4

29. L(«Кол», «Столб») = min (5, 3, 4) = 3

Задание 5

|  |
| --- |
| Main.cpp  #include <iostream>  #include "LCS.h"  #include <Windows.h>  #include <ctime>  int main()  {  SetConsoleCP(1251);  SetConsoleOutputCP(1251);  clock\_t t1 = 0, t2 = 0, t3 = 0, t4 = 0;  char z[100] = "";  t1 = clock();  char X[] = "QVTWNHO";  char Y[] = "RQTWYK";  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, "QVTWNHO", sizeof(Y) - 1, "RQTWYK");  std::cout << std::endl << "-- длина LCS: " << s << std::endl;  t2 = clock();  // наибольшая общая подпоследовательность  t3 = clock();  char x[] = "QVTWNHO";  char y[] = "RQTWYK";  int l = lcsd(x, y, z);  t4 = clock();  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: " << y;  std::cout << std::endl << " LCS: " << z;  std::cout << std::endl << " длина LCS: " << l;  std::cout << std::endl;  std::cout << std::endl << "Время вычисления LCS";  std::cout << std::endl << "Рекурсия: " << (t2 - t1);  std::cout << std::endl << "Динамическое программирование: " << (t4 - t3) << std::endl;  return 0;  } |

LCS.h

|  |
| --- |
| #pragma once  int lcs(int lenx, // длина последовательности X  const char x[], // последовательность X  int leny, // длина последовательности Y  const char y[] // последовательность Y  );  int lcsd(const char x[], // последовательность X  const char y[], // последовательность Y  char z[] // наибольшая общая подпоследовательность  ); |

LCS.cpp

|  |
| --- |
| // - LCS.cpp  // -- рекурсивное вычисление длины LCS  #include <algorithm>  #include "LCS.h"  #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])  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  }  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);  } |

Результат выполнения:

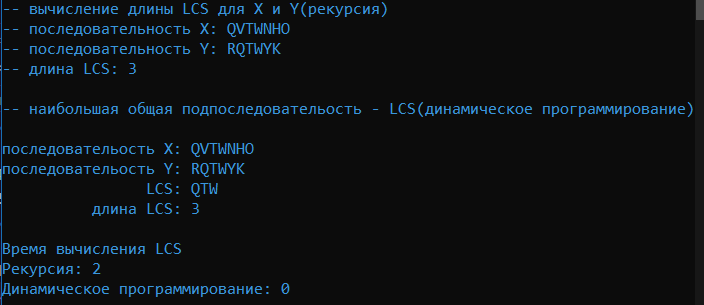
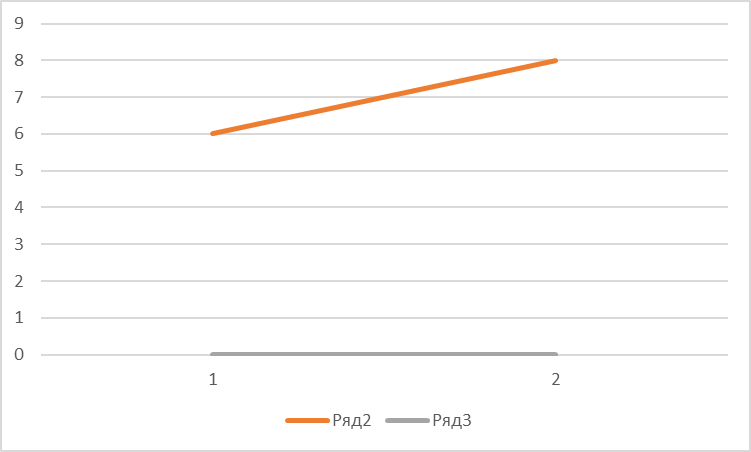


График  


Вывод: Динамическое программирование (DP) - это метод решения сложных задач путем разбиения их на более простые подзадачи и сохранения результатов решения каждой подзадачи для последующего повторного использования

Лабораторная работа №5

Транспортная задача

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| ПОТРЕБИТЕЛИ  ПОСТАВЩИКИ | 1 | 2 | 3 | 4 | 5 | 6 | ЗАПАСЫ |
| 1 | **19** | **9** | **13** | **10** | **18** | **8** | **175** |
| 2 | **17** | **7** | **15** | **12** | **14** | **20** | **120** |
| 3 | **8** | **12** | **18** | **15** | **9** | **18** | **157** |
| 4 | **11** | **17** | **17** | **10** | **20** | **9** | **166** |
| 5 | **10** | **18** | **16** | **7** | **17** | **11** | **107** |
| ПОТРЕБНОСТИ | **150** | **114** | **138** | **200** | **102** | **170** |  |

Для разрешимости транспортной задачи необходимо, чтобы суммарные запасы продукции у поставщиков равнялись суммарной потребности потребителей. Проверим это условие:

∑a = 175 + 120 + 157 + 166 + 107 = 725

∑b = 150 + 114 + 138 + 200 + 102 + 170 = 874

Так как запасы поставщиков меньше потребности потребителей, необходимо ввести фиктивного поставщика 6, с запасом продукции 874 – 725 = 149. Стоимость доставки единицы продукции от фиктивного поставщика ко всем потребителям примем равной нулю.

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| ПОТРЕБИТЕЛИ  ПОСТАВЩИКИ | 1 | 2 | 3 | 4 | 5 | 6 | ЗАПАСЫ |
| 1 | **19** | **9** | **13** | **10** | **18** | **8** | **175** |
| 2 | **17** | **7** | **15** | **12** | **14** | **20** | **120** |
| 3 | **8** | **12** | **18** | **15** | **9** | **18** | **157** |
| 4 | **11** | **17** | **17** | **10** | **20** | **9** | **166** |
| 5 | **10** | **18** | **16** | **7** | **17** | **11** | **107** |
| 6 | **0** | **0** | **0** | **0** | **0** | **0** | **149** |
| ПОТРЕБНОСТИ | **150** | **114** | **138** | **200** | **102** | **170** |  |

**Этап I.** Метод наименьшей стоимости

Суть метода заключается в том, что из всей таблицы стоимостей выбирают клетку с наименьшей стоимостью, для этой ячейки присваиваем 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 строку и столбец, если израсходованы запасы поставщика и удовлетворены потребности потребителя. Повторяем, пока все запасы не будут распределены, а потребности удовлетворены.

Искомый элемент равен c22 = 7, а x22 = min(120, 114) = 114.

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| ПОТРЕБИТЕЛИ  ПОСТАВЩИКИ | 1 | 2 | 3 | 4 | 5 | 6 | ЗАПАСЫ |
| 1 | **19** | **9** | **13** | **10** | **18** | **8** | **175** |
| 2 | **17** | **7** | **15** | **12** | **14** | **20** | **6** |
| 3 | **8** | **12** | **18** | **15** | **9** | **18** | **157** |
| 4 | **11** | **17** | **17** | **10** | **20** | **9** | **166** |
| 5 | **10** | **18** | **16** | **7** | **17** | **11** | **107** |
| 6 | **0** | **0** | **0** | **0** | **0** | **0** | **149** |
| ПОТРЕБНОСТИ | **150** | **0** | **138** | **200** | **102** | **170** |  |

Искомый элемент равен c54 = 7.

x54 = min(107, 200) = 107.

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| ПОТРЕБИТЕЛИ  ПОСТАВЩИКИ | 1 | 2 | 3 | 4 | 5 | 6 | ЗАПАСЫ |
| 1 | **19** | **9** | **13** | **10** | **18** | **8** | **175** |
| 2 | **17** | **7|114** | **15** | **12** | **14** | **20** | **6** |
| 3 | **8** | **12** | **18** | **15** | **9** | **18** | **157** |
| 4 | **11** | **17** | **17** | **10** | **20** | **9** | **166** |
| 5 | **10** | **18** | **16** | **7|107** | **17** | **11** | **0** |
| 6 | **0** | **0** | **0** | **0** | **0** | **0** | **149** |
| ПОТРЕБНОСТИ | **150** | **0** | **138** | **93** | **102** | **170** |  |

Искомый элемент равен c16 = 8.

x16 = min(175, 170) = 170.

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| ПОТРЕБИТЕЛИ  ПОСТАВЩИКИ | 1 | 2 | 3 | 4 | 5 | 6 | ЗАПАСЫ |
| 1 | **19** | **9** | **13** | **10** | **18** | **8|170** | **5** |
| 2 | **17** | **7|114** | **15** | **12** | **14** | **20** | **6** |
| 3 | **8** | **12** | **18** | **15** | **9** | **18** | **157** |
| 4 | **11** | **17** | **17** | **10** | **20** | **9** | **166** |
| 5 | **10** | **18** | **16** | **7|107** | **17** | **11** | **0** |
| 6 | **0** | **0** | **0** | **0** | **0** | **0** | **149** |
| ПОТРЕБНОСТИ | **150** | **0** | **138** | **93** | **102** | **0** |  |

Искомый элемент равен c31 = 8.

x31 = min(157, 150) = 150.

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| ПОТРЕБИТЕЛИ  ПОСТАВЩИКИ | 1 | 2 | 3 | 4 | 5 | 6 | ЗАПАСЫ |
| 1 | **19** | **9** | **13** | **10** | **18** | **8|170** | **5** |
| 2 | **17** | **7|114** | **15** | **12** | **14** | **20** | **6** |
| 3 | **8|150** | **12** | **18** | **15** | **9** | **18** | **7** |
| 4 | **11** | **17** | **17** | **10** | **20** | **9** | **166** |
| 5 | **10** | **18** | **16** | **7|107** | **17** | **11** | **0** |
| 6 | **0** | **0** | **0** | **0** | **0** | **0** | **149** |
| ПОТРЕБНОСТИ | **0** | **0** | **138** | **93** | **102** | **0** |  |

Искомый элемент равен c35 = 9.

x31 = min(7, 102) = 7.

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| ПОТРЕБИТЕЛИ  ПОСТАВЩИКИ | 1 | 2 | 3 | 4 | 5 | 6 | ЗАПАСЫ |
| 1 | **19** | **9** | **13** | **10** | **18** | **8|170** | **5** |
| 2 | **17** | **7|114** | **15** | **12** | **14** | **20** | **6** |
| 3 | **8|150** | **12** | **18** | **15** | **9|7** | **18** | **0** |
| 4 | **11** | **17** | **17** | **10** | **20** | **9** | **166** |
| 5 | **10** | **18** | **16** | **7|107** | **17** | **11** | **0** |
| 6 | **0** | **0** | **0** | **0** | **0** | **0** | **149** |
| ПОТРЕБНОСТИ | **0** | **0** | **138** | **93** | **95** | **0** |  |

Искомый элемент равен c44 = 10.

x14 = min(166, 93) = 93.

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| ПОТРЕБИТЕЛИ  ПОСТАВЩИКИ | 1 | 2 | 3 | 4 | 5 | 6 | ЗАПАСЫ |
| 1 | **19** | **9** | **13** | **10** | **18** | **8|170** | **5** |
| 2 | **17** | **7|114** | **15** | **12** | **14** | **20** | **6** |
| 3 | **8|150** | **12** | **18** | **15** | **9|7** | **18** | **0** |
| 4 | **11** | **17** | **17** | **10|93** | **20** | **9** | **73** |
| 5 | **10** | **18** | **16** | **7|107** | **17** | **11** | **0** |
| 6 | **0** | **0** | **0** | **0** | **0** | **0** | **149** |
| ПОТРЕБНОСТИ | **0** | **0** | **138** | **0** | **95** | **0** |  |

Искомый элемент равен c13 = 13.

x14 = min(138, 5) = 5.

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| ПОТРЕБИТЕЛИ  ПОСТАВЩИКИ | 1 | 2 | 3 | 4 | 5 | 6 | ЗАПАСЫ |
| 1 | **19** | **9** | **13|5** | **10** | **18** | **8|170** | **0** |
| 2 | **17** | **7|114** | **15** | **12** | **14** | **20** | **6** |
| 3 | **8|150** | **12** | **18** | **15** | **9|7** | **18** | **0** |
| 4 | **11** | **17** | **17** | **10|93** | **20** | **9** | **73** |
| 5 | **10** | **18** | **16** | **7|107** | **17** | **11** | **0** |
| 6 | **0** | **0** | **0** | **0** | **0** | **0** | **149** |
| ПОТРЕБНОСТИ | **0** | **0** | **133** | **0** | **95** | **0** |  |

Искомый элемент равен c25 = 14.

x14 = min(95, 6) = 6.

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| ПОТРЕБИТЕЛИ  ПОСТАВЩИКИ | 1 | 2 | 3 | 4 | 5 | 6 | ЗАПАСЫ |
| 1 | **19** | **9** | **13|5** | **10** | **18** | **8|170** | **0** |
| 2 | **17** | **7|114** | **15** | **12** | **14|6** | **20** | **0** |
| 3 | **8|150** | **12** | **18** | **15** | **9|7** | **18** | **0** |
| 4 | **11** | **17** | **17** | **10|93** | **20** | **9** | **73** |
| 5 | **10** | **18** | **16** | **7|107** | **17** | **11** | **0** |
| 6 | **0** | **0** | **0** | **0** | **0** | **0** | **149** |
| ПОТРЕБНОСТИ | **0** | **0** | **133** | **0** | **89** | **0** |  |

Искомый элемент равен c43 = 17.

x14 = min(133, 73) = 73.

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| ПОТРЕБИТЕЛИ  ПОСТАВЩИКИ | 1 | 2 | 3 | 4 | 5 | 6 | ЗАПАСЫ |
| 1 | **19** | **9** | **13|5** | **10** | **18** | **8|170** | **0** |
| 2 | **17** | **7|114** | **15** | **12** | **14|6** | **20** | **0** |
| 3 | **8|150** | **12** | **18** | **15** | **9|7** | **18** | **0** |
| 4 | **11** | **17** | **17** | **10|93** | **20** | **9** | **0** |
| 5 | **10** | **18** | **16** | **7|107** | **17** | **11** | **0** |
| 6 | **0** | **0** | **0** | **0** | **0** | **0** | **149** |
| ПОТРЕБНОСТИ | **0** | **0** | **60** | **0** | **89** | **0** |  |

Искомый элемент равен c63 = 0.

x14 = min(149, 60) = 60.

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| ПОТРЕБИТЕЛИ  ПОСТАВЩИКИ | 1 | 2 | 3 | 4 | 5 | 6 | ЗАПАСЫ |
| 1 | **19** | **9** | **13|5** | **10** | **18** | **8|170** | **0** |
| 2 | **17** | **7|114** | **15** | **12** | **14|6** | **20** | **0** |
| 3 | **8|150** | **12** | **18** | **15** | **9|7** | **18** | **0** |
| 4 | **11** | **17** | **17** | **10|93** | **20** | **9** | **0** |
| 5 | **10** | **18** | **16** | **7|107** | **17** | **11** | **0** |
| 6 | **0** | **0** | **0|60** | **0** | **0** | **0** | **89** |
| ПОТРЕБНОСТИ | **0** | **0** | **0** | **0** | **89** | **0** |  |

Искомый элемент равен c66 = 0.

x14 = min(89, 89) = 89.

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| ПОТРЕБИТЕЛИ  ПОСТАВЩИКИ | 1 | 2 | 3 | 4 | 5 | 6 | ЗАПАСЫ |
| 1 | **19** | **9** | **13|5** | **10** | **18** | **8|170** | **0** |
| 2 | **17** | **7|114** | **15** | **12** | **14|6** | **20** | **0** |
| 3 | **8|150** | **12** | **18** | **15** | **9|7** | **18** | **0** |
| 4 | **11** | **17** | **17** | **10|93** | **20** | **9** | **0** |
| 5 | **10** | **18** | **16** | **7|107** | **17** | **11** | **0** |
| 6 | **0** | **0** | **0|60** | **0** | **0|89** | **0** | **0** |
| ПОТРЕБНОСТИ | **0** | **0** | **0** | **0** | **0** | **0** |  |

В результате получен первый опорный план, который является допустимым, так как все грузы из баз вывезены, потребность потребителей удовлетворена, а план соответствует системе ограничений транспортной задачи.

5\*13 + 170\*8 + 114\*7 + 6\*14 + 150\*8 + 7\*9 + 73\*17 + 93\*10 + 107\*7 + 60\*0 + 89\*0 = 6490

**Этап II.** Метод потенциалов

Каждому поставщику ai ставим в соответствие некоторое число - ui, называемое потенциалом поставщика. Каждому потребителю bj ставим в соответствие некоторое число - vj, называемое потенциалом потребителя. Для базисной ячейки (задействованного маршрута), сумма потенциалов поставщика и потребителя должна быть равна тарифу данного маршрута.

ui + vj = cij

Найдем предварительные потенциалы ui, vj. по занятым клеткам таблицы, полагая, что u1 = 0.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| A1B3 : | v3 + u1 = 13 | | v3 = 13 - 0 = 13 | | |
| A1B6 : | v6 + u1 = 8 | | v6 = 8 - 0 = 8 | | |
| A4B3 : | v3 + u4 = 17 | | u4 = 17 - 13 = 4 | | |
| A4B4 : | v4 + u4 = 10 | | v4 = 10 - 4 = 6 | | |
| A5B4 : | v4 + u5 = 7 | | u5 = 7 - 6 = 1 | | |
| A6B3 : | v3 + u6 = 0 | | u6 = 0 - 13 = -13 | | |
| A6B5 : | v5 + u6 = 0 | | v5 = 0 - (-13) = 13 | | |
| A2B5 : | v5 + u2 = 14 | | u2 = 14 - 13 = 1 | | |
| A3B5 : | v5 + u3 = 9 | | u3 = 9 - 13 = -4 | | |
| A2B2 : | v2 + u2 = 7 | | v2 = 7 - 1 = 6 | | |
| A3B1 : | v1 + u3 = 8 | | v1 = 8 - (-4) = 12 | | |
|  |  | |  | | |
|  |  | |  | | |
| ПОТРЕБИТЕЛИ  ПОСТАВЩИКИ | | 1 | | 2 | 3 | | 4 | 5 | 6 | U |
| 1 | | **19** | | **9** | **13|5** | | **10** | **18** | **8|170** | **0** |
| 2 | | **17** | | **7|114** | **15** | | **12** | **14|6** | **20** | **1** |
| 3 | | **8|150** | | **12** | **18** | | **15** | **9|7** | **18** | **-4** |
| 4 | | **11** | | **17** | **17** | | **10|93** | **20** | **9** | **4** |
| 5 | | **10** | | **18** | **16** | | **7|107** | **17** | **11** | **1** |
| 6 | | **0** | | **0** | **0|60** | | **0** | **0|89** | **0** | **-13** |
| V | | **12** | | **6** | **13** | | **6** | **15** | **8** |  |

Опорный план не является оптимальным, так как существуют оценки свободных клеток, для которых ui + vj > cij

(4;1): ∆12 = 0 + 12 - 11 = 1 > 0

(5;1): ∆13 = 0 + 12 - 10 = 2 > 0

max(1, 2) = 2

Выбираем максимальную оценку свободной клетки (5;1): 10. Для этого в перспективную клетку (5;1) поставим знак «+», а в остальных вершинах многоугольника чередующиеся знаки «-», «+», «-».

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| ПОТРЕБИТЕЛИ  ПОСТАВЩИКИ | 1 | 2 | 3 | 4 | 5 | 6 | U |
| 1 | **19** | **9** | **13|5** | **10** | **18** | **8|170** | **0** |
| 2 | **17** | **7|114** | **15** | **12** | **14|6** | **20** | **1** |
| 3 | **8|150-** | **12** | **18** | **15** | **9|7+** | **18** | **-4** |
| 4 | **11** | **17** | **17** | **10|93+** | **20** | **9** | **4** |
| 5 | **10+** | **18** | **16** | **7|107-** | **17** | **11** | **1** |
| 6 | **0** | **0** | **0|60+** | **0** | **0|89-** | **0** | **-13** |
| V | **12** | **6** | **13** | **6** | **15** | **8** |  |

Из грузов хij стоящих в минусовых клетках, выбираем наименьшее, т.е. Прибавляем 89 к объемам грузов, стоящих в плюсовых и вычитаем 89 из xij, стоящих в минусовых клетках. В результате получим новый опорный план.

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| ПОТРЕБИТЕЛИ  ПОСТАВЩИКИ | 1 | 2 | 3 | 4 | 5 | 6 | U |
| 1 | **19** | **9** | **13|5** | **10** | **18** | **8|170** | **0** |
| 2 | **17** | **7|114** | **15** | **12** | **14|6** | **20** | **0** |
| 3 | **8|61-** | **12** | **18** | **15** | **9|96+** | **18** | **0** |
| 4 | **11** | **17** | **17** | **10|93+** | **20** | **9** | **0** |
| 5 | **10|89+** | **18** | **16** | **7|107-** | **17** | **11** | **0** |
| 6 | **0** | **0** | **0|149+** | **0** | **0** | **0** | **0** |
| V | **0** | **0** | **0** | **0** | **0** | **0** |  |

Проверим оптимальность опорного плана. Найдем предварительные потенциалы ui, vj. по занятым клеткам таблицы, в которых ui + vj = cij, полагая, что u1 = 0.

u1 + v3 = 13; 0 + v3 = 13; v3 = 13

u1 + v6 = 8; 0 + v6 = 8; v6 = 8

u6 + v3 = 0; 0 + v3 = 0; u6 = -13

v5 + u6 = 0 v5 = 0 - (-13) = 13

v5 + u2 = 14 u2 = 14 - 13 = 1

v5 + u3 = 9 u3 = 9 - 13 = -4

v2 + u2 = 7 v2 = 7 - 1 = 6

v1 + u3 = 8 v1 = 8 - (-4) = 12

v1 + u4 = 11 u4 = 11 - 12 = -1

v4 + u4 = 10 v4 = 10 - (-1) = 11

v4 + u5 = 7 u5 = 7 - 11 = -4

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| ПОТРЕБИТЕЛИ  ПОСТАВЩИКИ | 1 | 2 | 3 | 4 | 5 | 6 | U |
| 1 | **19** | **9** | **13|5** | **10** | **18** | **8|170** | **13** |
| 2 | **17** | **7|114** | **15** | **12** | **14|6** | **20** | **8** |
| 3 | **8|61-** | **12** | **18** | **15** | **9|96+** | **18** | **-13** |
| 4 | **11** | **17** | **17** | **10|93+** | **20** | **9** | **1** |
| 5 | **10|89+** | **18** | **16** | **7|107-** | **17** | **11** | **-4** |
| 6 | **0** | **0** | **0|149+** | **0** | **0** | **0** | **6** |
| V | **-7** | **0** | **-4** | **11** | **-1** | **12** |  |

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

Минимальные затраты составят: F(x) = 13\*5 + 8\*172 + 7\*114 + 14\*6 + 10\*72 + 11\*87 + 13\*80 + 12\*88 + 7\*107 + 0\*149 = 6320

Лабораторная работа №6

Вариант 7

Задание 1



Матрица инцидентности

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

Матрица смежности

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

Список смежности

0:3

1:

2:

3:1 4 5 6

4:1 6

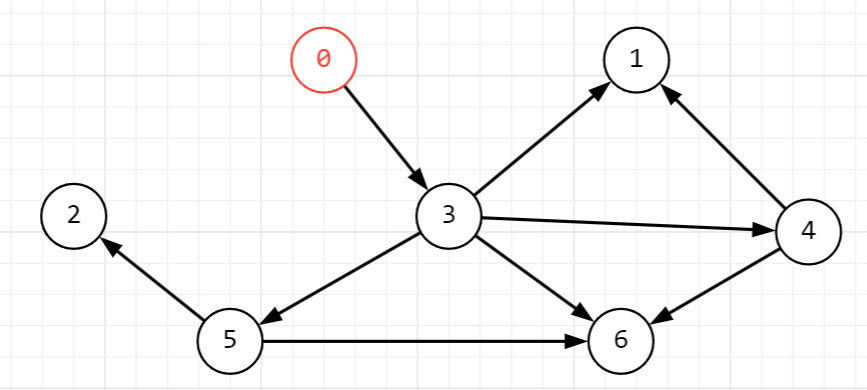
5: 2 6

6:

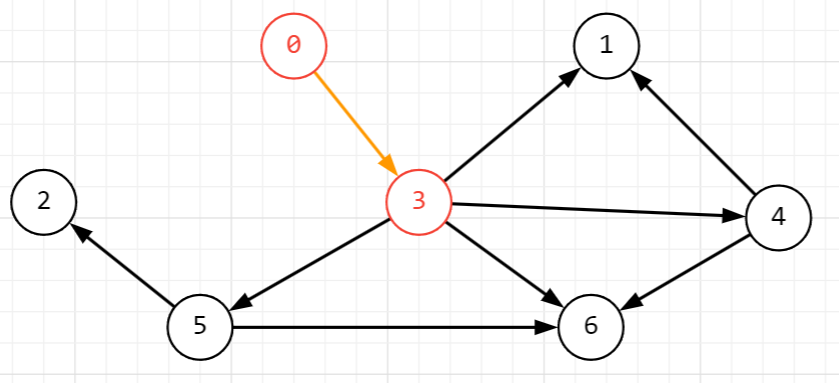
Задание 2

Стартовая вершина :0

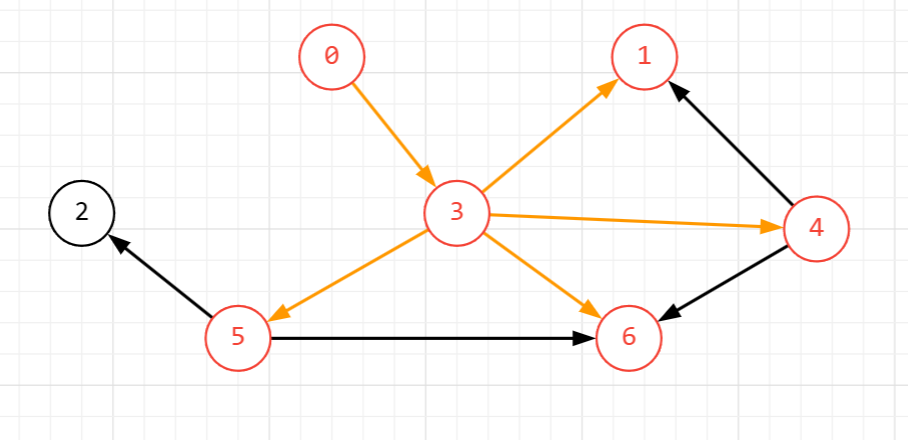
BFS



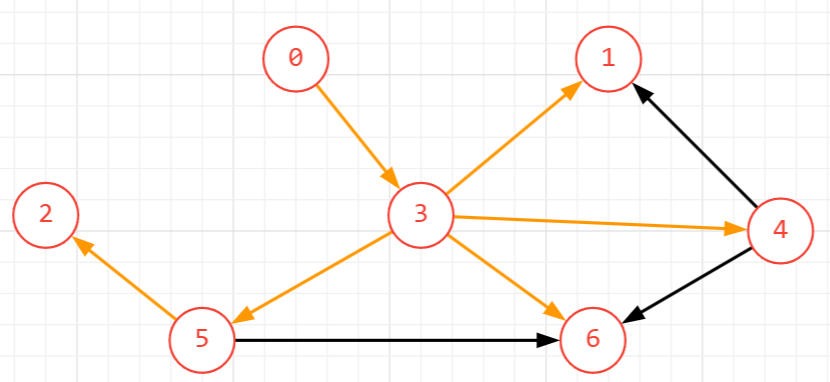
|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Q | 0 |  |  |  |  |
| C | G | W | W | W | W |
| D | 0 | I | I | I | I |
| P | N | N | N | N | N |



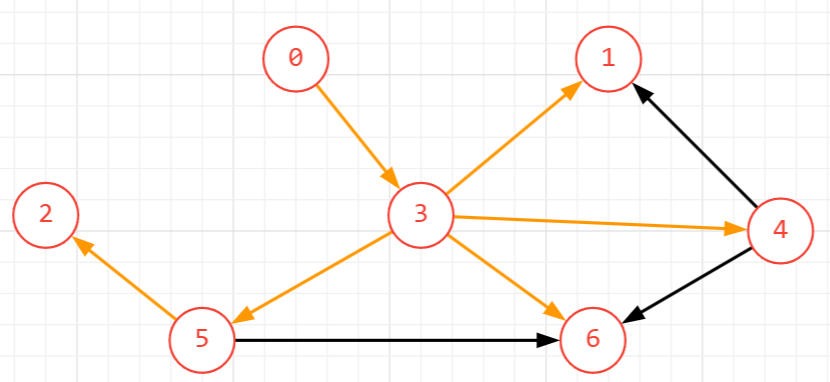
|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Q | 3 |  |  |  |  |  |  |
| C | B | W | W | G | W | W | W |
| D | 0 | I | I | 1 | I | I | I |
| P | N | N | N | 0 | N | N | N |



|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Q | 1 | 4 | 5 | 6 |  |  |  |
| C | B | G | B | B | G | G | G |
| D | 0 | 2 | 2 | 1 | 2 | 2 | 2 |
| P | N | 3 | 3 | 0 | 3 | 3 | 3 |

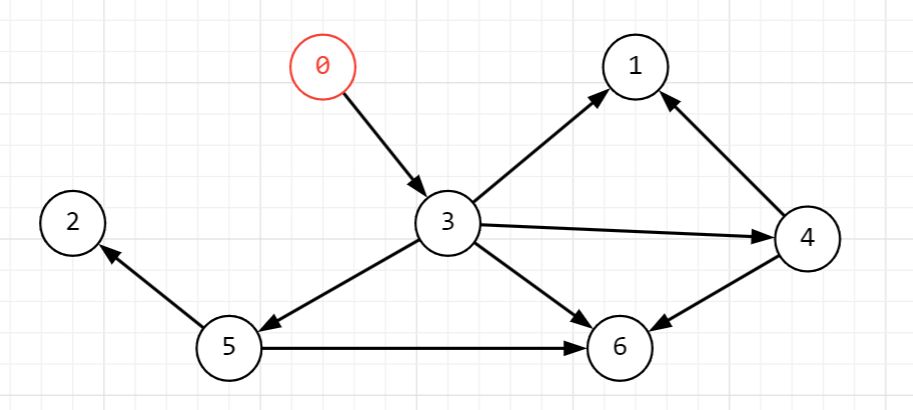


|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Q | 6 | 2 |  |  |  |  |  |
| C | B | G | B | B | B | B | G |
| D | 0 | I | 2 | 1 | 2 | 2 | 2 |
| P | N | 3 | 5 | 0 | 3 | 3 | 3 |

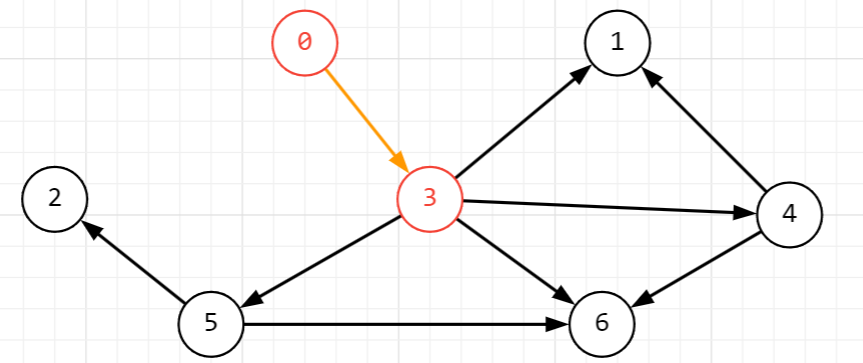


|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Q |  |  |  |  |  |  |  |
| C | B | B | B | B | B | B | B |
| D | 0 | 3 | 2 | 1 | 2 | 2 | 2 |
| P | N | 3 | 5 | 0 | 3 | 3 | 3 |

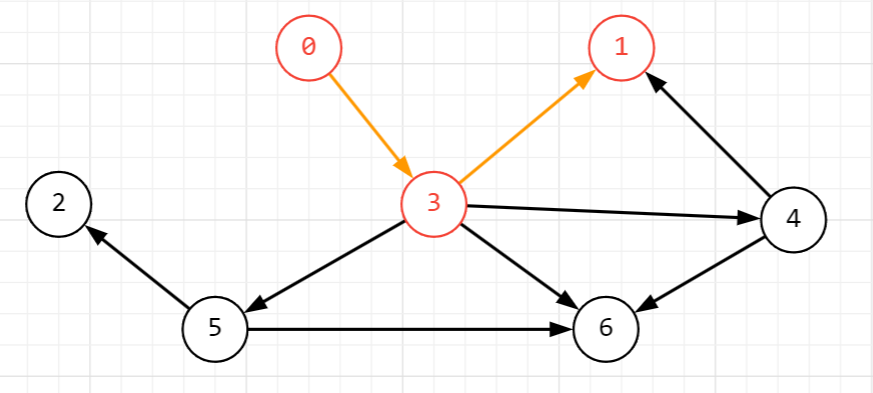
DFS



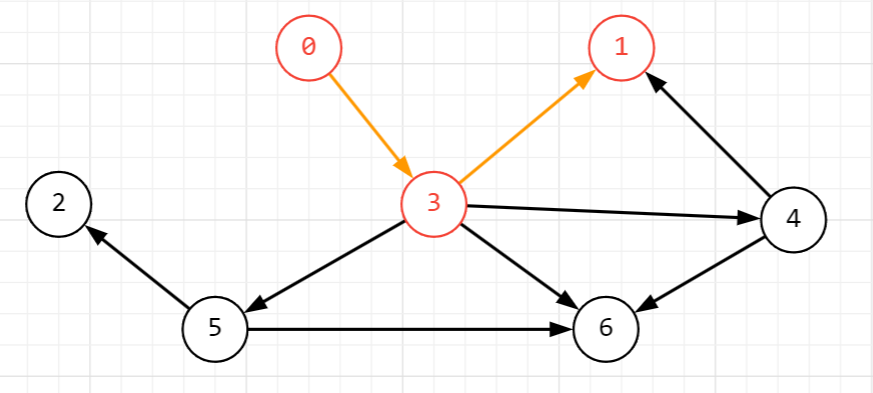
|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| T | 1 |  |  |  |  |  |  |
| C | G | W | W | W | W | W | W |
| D | 0 | I | I | I | I | I | I |
| P | N | N | N | N | N | N | N |
| F | 0 | 0 | 0 | 0 | 0 | 0 | 0 |



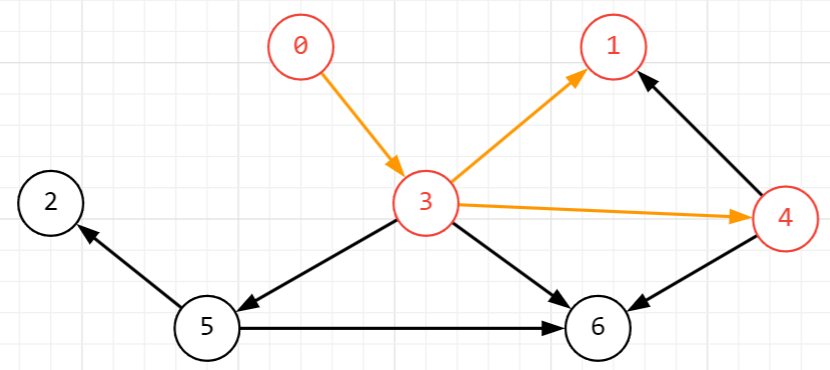
|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| T | 2 |  |  |  |  |  |  |
| C | G | W | W | G | W | W | W |
| D | 0 | I | I | 1 | I | I | I |
| P | N | N | N | 0 | N | N | N |
| F | 0 | 0 | 0 | 0 | 0 | 0 | 0 |



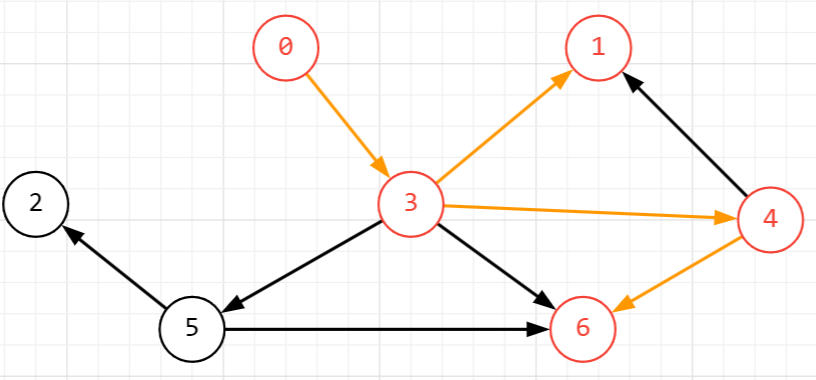
|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| T | 3 |  |  |  |  |  |  |
| C | G | G | W | G | W | W | W |
| D | 0 | 2 | I | 1 | I | I | I |
| P | N | 3 | N | 0 | N | N | N |
| F | 0 | 0 | 0 | 0 | 0 | 0 | 0 |



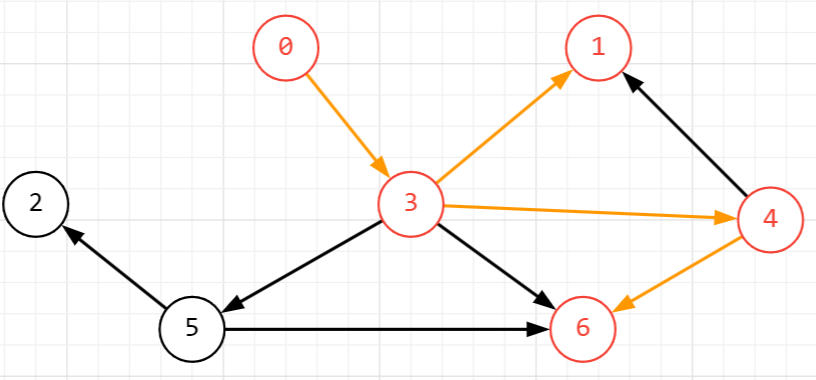
|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| T | 4 |  |  |  |  |  |  |
| C | G | B | W | G | W | W | W |
| D | 0 | 2 | I | 1 | I | I | I |
| P | N | 3 | N | 0 | N | N | N |
| F | 0 | 4 | 0 | 0 | 0 | 0 | 0 |



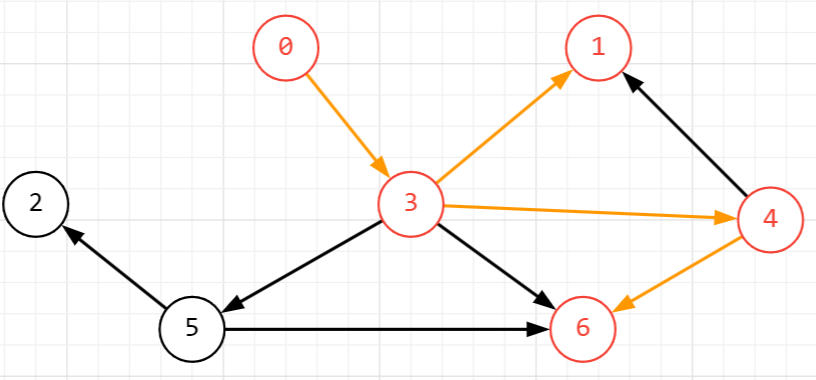
|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| T | 5 |  |  |  |  |  |  |
| C | G | G | W | G | G | W | W |
| D | 0 | 2 | I | 1 | 2 | I | I |
| P | N | 3 | N | 0 | 3 | N | N |
| F | 0 | 0 | 0 | 0 | 0 | 0 | 0 |



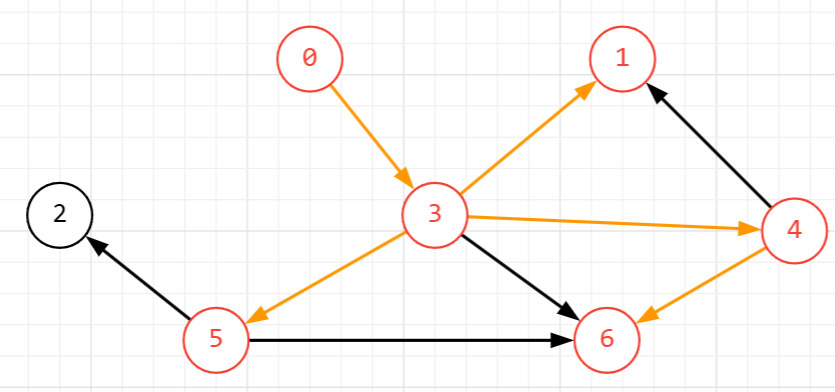
|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| T | 6 |  |  |  |  |  |  |
| C | G | G | W | G | G | W | G |
| D | 0 | 2 | I | 1 | 2 | I | 3 |
| P | N | 3 | N | 0 | 3 | N | 4 |
| F | 0 | 0 | 0 | 0 | 0 | 0 | 0 |



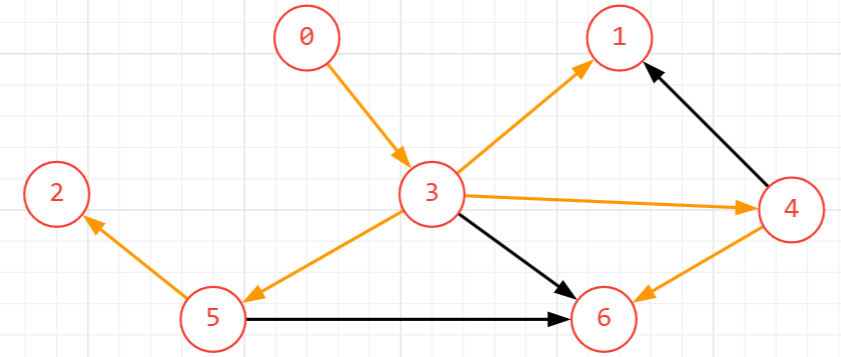
|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| T | 7 |  |  |  |  |  |  |
| C | G | G | W | G | G | W | B |
| D | 0 | 2 | I | 1 | 2 | I | 3 |
| P | N | 3 | N | 0 | 3 | N | 4 |
| F | 0 | 0 | 0 | 0 | 0 | 0 | 6 |



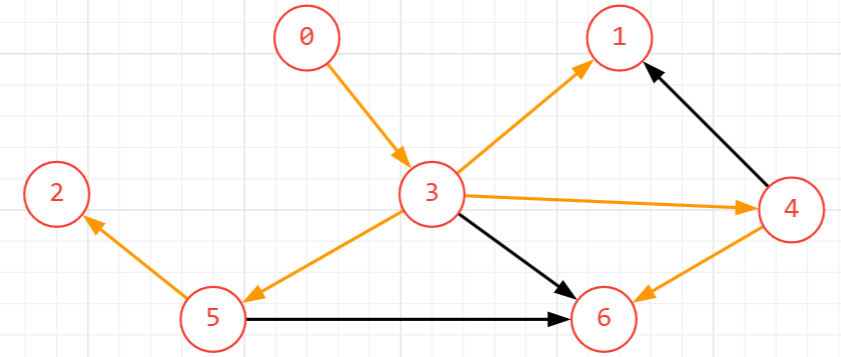
|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| T | 8 |  |  |  |  |  |  |
| C | G | G | W | G | G | W | B |
| D | 0 | 2 | I | 1 | 2 | I | 3 |
| P | N | 3 | N | 0 | 3 | N | 4 |
| F | 0 | 0 | 0 | 0 | 0 | 0 | 6 |



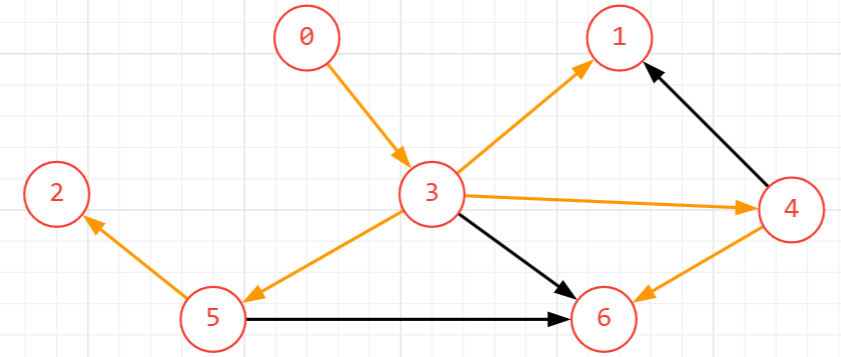
|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| T | 9 |  |  |  |  |  |  |
| C | G | G | W | G | G | G | B |
| D | 0 | 2 | I | 1 | 2 | 2 | 3 |
| P | N | 3 | N | 0 | 3 | 3 | 4 |
| F | 0 | 0 | 0 | 0 | 0 | 0 | 6 |



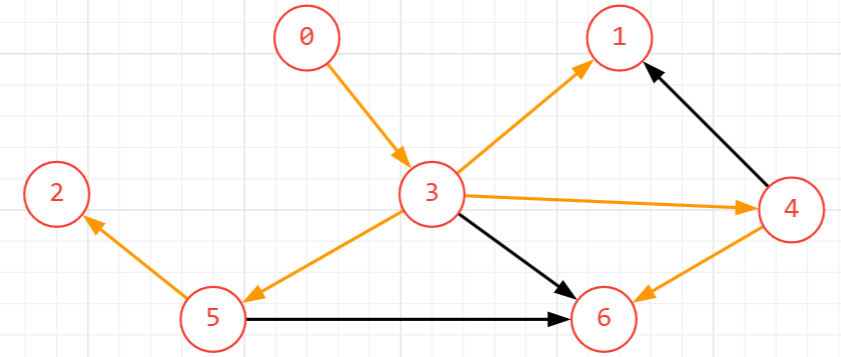
|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| T | 10 |  |  |  |  |  |  |
| C | G | G | G | G | G | G | B |
| D | 0 | 2 | 3 | 1 | 2 | 2 | 3 |
| P | N | 3 | 5 | 0 | 3 | 3 | 4 |
| F | 0 | 0 | 0 | 0 | 0 | 0 | 6 |



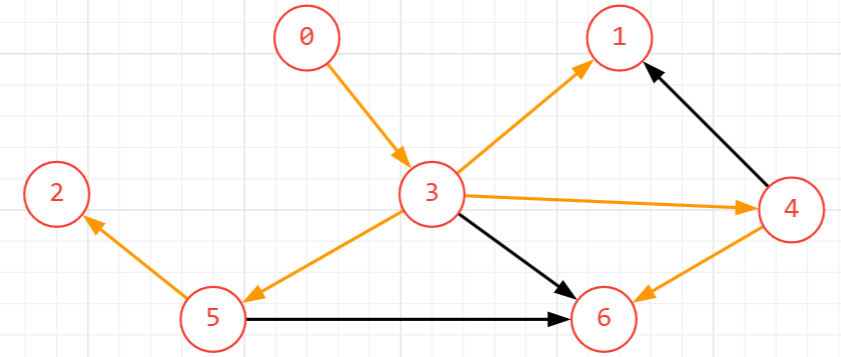
|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| T | 11 |  |  |  |  |  |  |
| C | G | G | G | G | G | G | B |
| D | 0 | 2 | 3 | 1 | 2 | 2 | 3 |
| P | N | 3 | 5 | 0 | 3 | 3 | 4 |
| F | 0 | 0 | 0 | 0 | 0 | 0 | 6 |



|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| T | 12 |  |  |  |  |  |  |
| C | G | G | G | G | G | G | B |
| D | 0 | 2 | 3 | 1 | 2 | 2 | 3 |
| P | N | 3 | 5 | 0 | 3 | 3 | 4 |
| F | 0 | 0 | 0 | 0 | 0 | 0 | 6 |

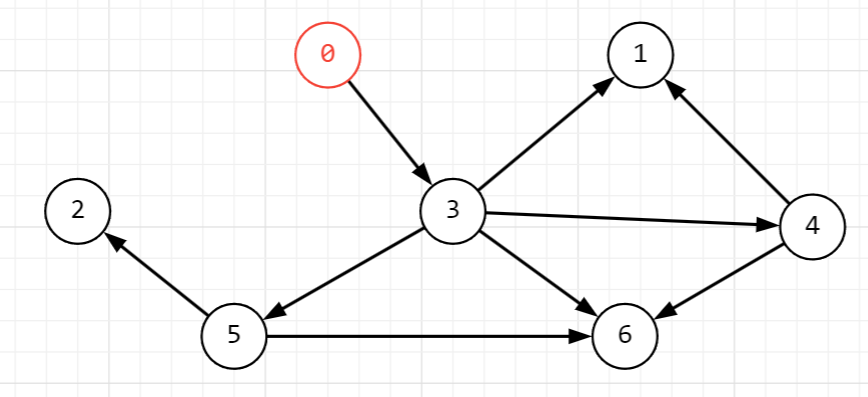


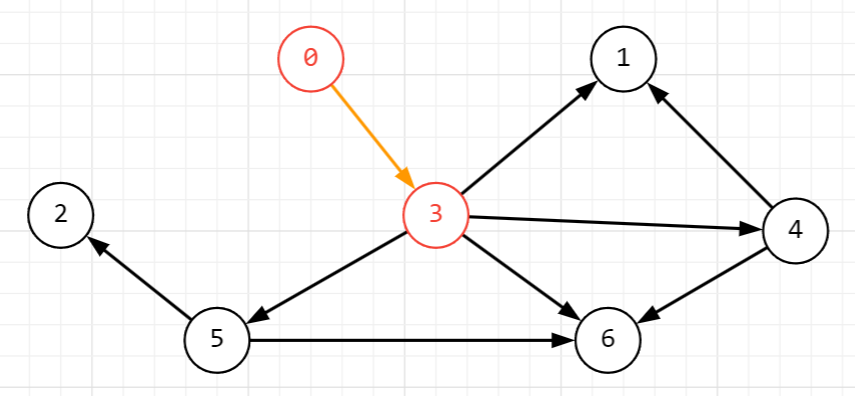
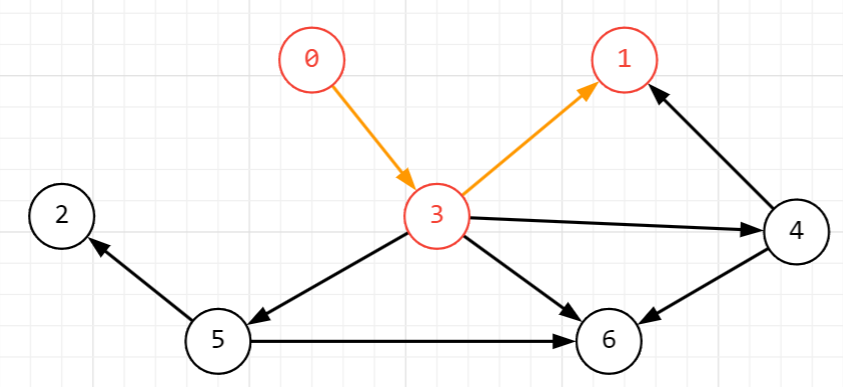
|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| T | 13 |  |  |  |  |  |  |
| C | G | G | G | G | G | G | B |
| D | 0 | 2 | 3 | 1 | 2 | 2 | 3 |
| P | N | 3 | 5 | 0 | 3 | 3 | 4 |
| F | 0 | 0 | 0 | 0 | 0 | 0 | 6 |

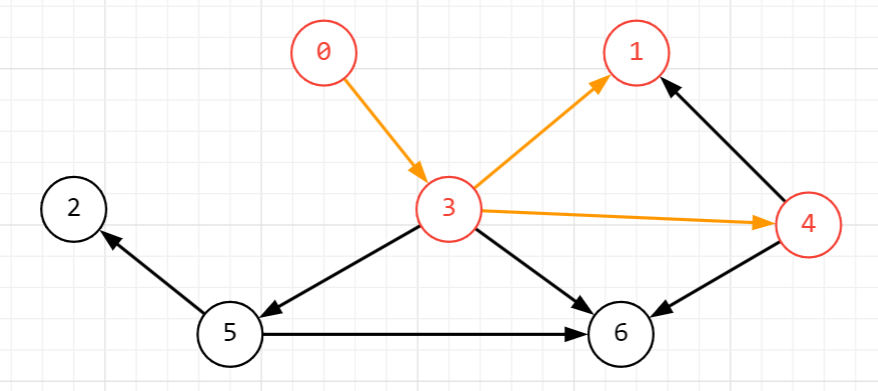


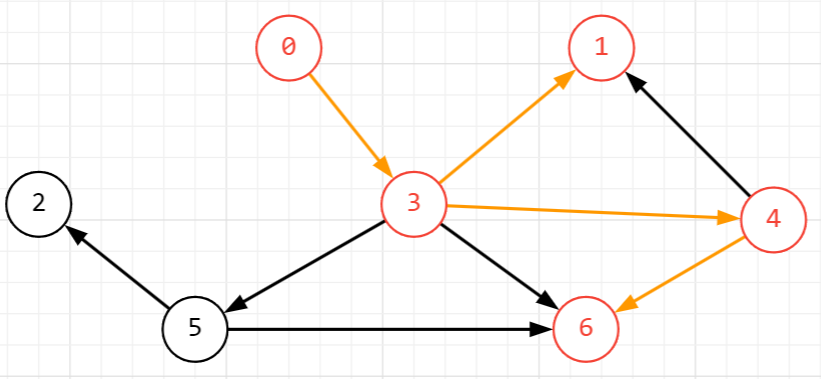
|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| T | 14 |  |  |  |  |  |  |
| C | B | B | B | B | B | B | B |
| D | 0 | 2 | 3 | 1 | 2 | 2 | 3 |
| P | N | 3 | 5 | 0 | 3 | 3 | 4 |
| F | 4 | 14 | 8 | 6 | 9 | 11 | 6 |

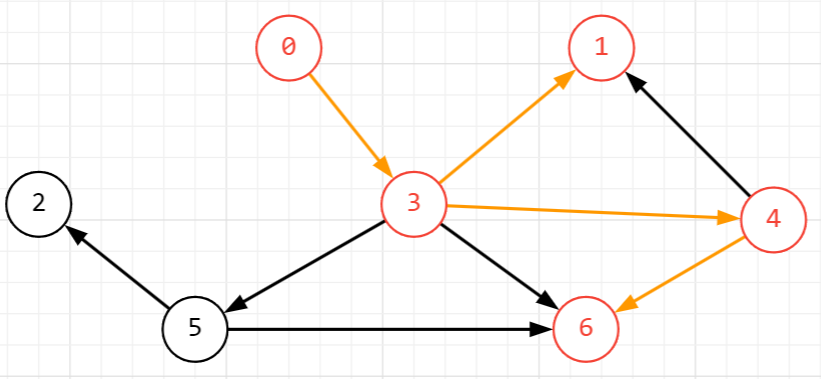
Топологическая сортировка

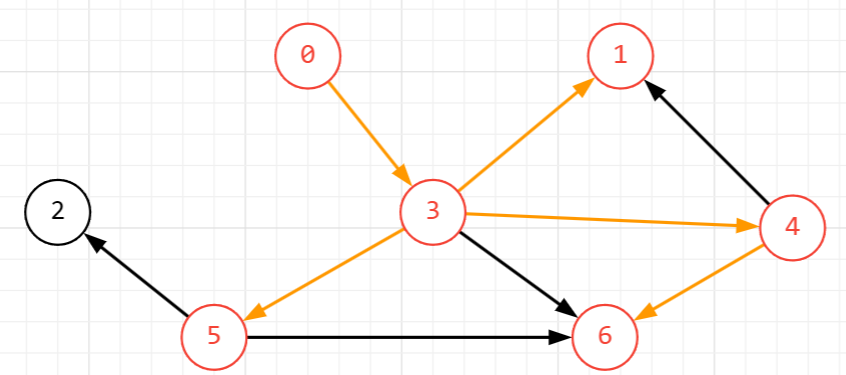


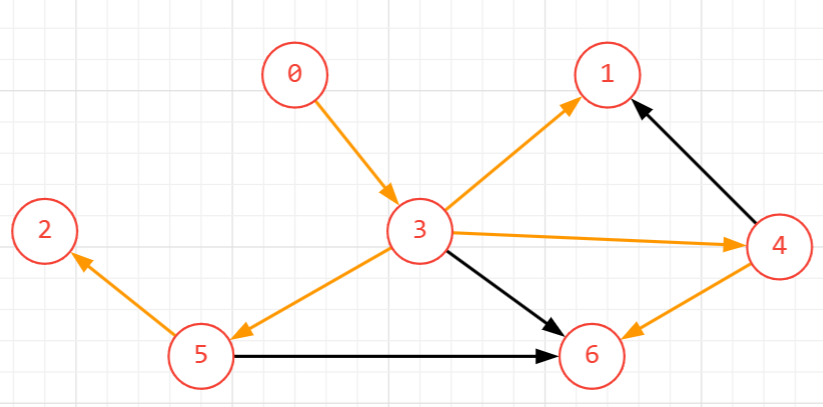
  


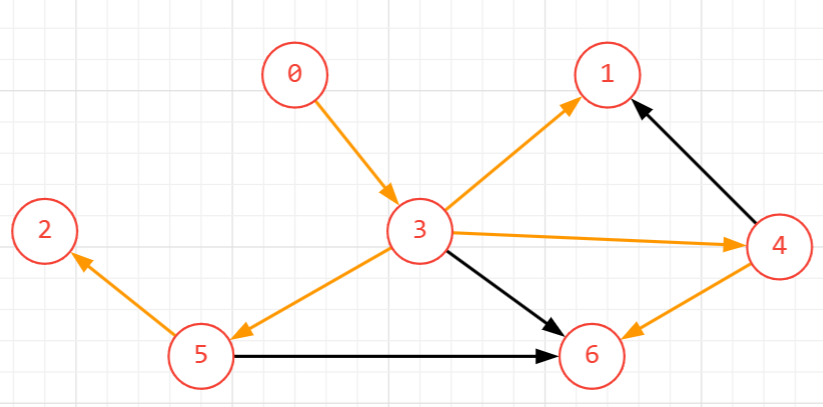


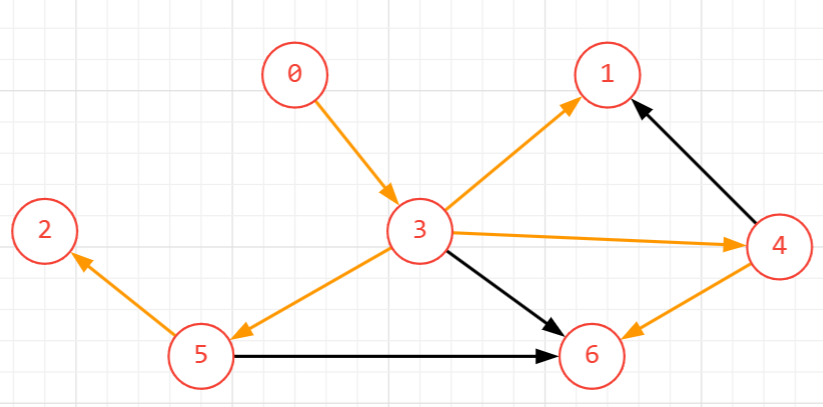


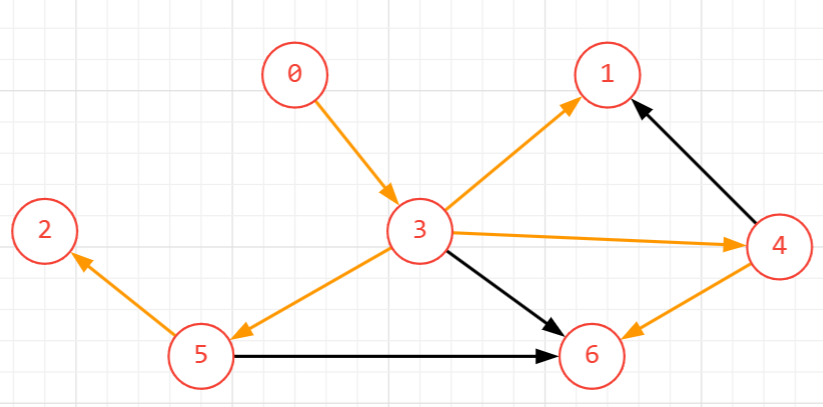


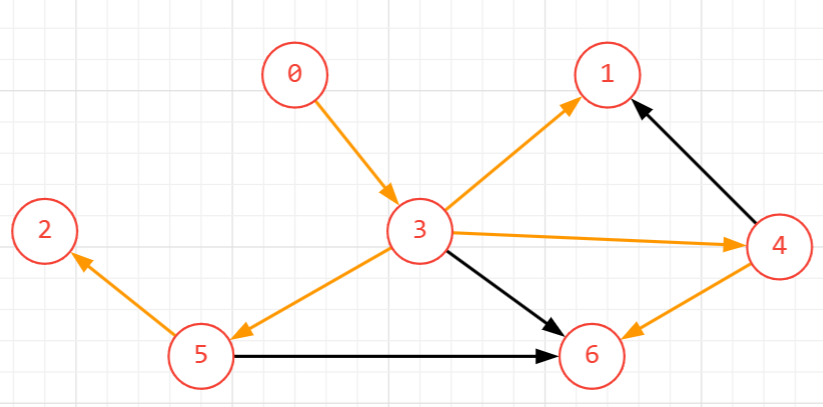








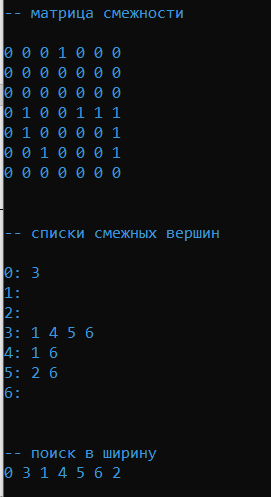




Задание 3

|  |
| --- |
| Graph.h  #pragma once  #include <list>  #include "Graph.h"  namespace graph  {  struct AList;  struct AMatrix // матрица смежности  {  int n\_vertex; // количество вершин  int\* mr; // матрица  AMatrix(int n); // создать нулевую матрицу n\*n  AMatrix(int n, int mr[]); // создать матрицу n\*n и  AMatrix(const AMatrix& am); // создать подобную матрицу  AMatrix(const AList& al); // создать матрицу из спискового  void set(int i, int j, int r); // записать mr[i,j] = r  int get(int i, int j)const; // элемент mr[i,j]  };  struct AList // списки смежности  {  int n\_vertex; // количество вершин  std::list<int>\* mr; // массив списков  void create(int n); // создать массив пустых списков  AList(int n); // создать массив пустых списков  AList(int n, int mr[]); // создать списковое представление  AList(const AMatrix& am); // создать списковое представление  AList(const AList& al); // создать подобную структуру  void add(int i, int j); // добавить в i-ый список  int size(int i) const; // размер i-го списка  int get(int i, int j)const; // j-ый элемент i-го списка  };  };  #pragma once  // ---BFS.h  //  #pragma once  #include "Graph.h"  #include <queue>  struct BFS // breadth-first search поиск в ширину (связный граф)  {  const static int INF = 0x7fffffff;  const static int NIL = -1;  enum Color { WHITE, GRAY, BLACK }; //  const graph::AList\* al; // исходный граф  Color\* c; // цвет вершины  int\* d; // расстояние до вершины  int\* p; // предшествующая вершина  std::queue<int> q; // очередь  BFS(const graph::AList& al, int s);  BFS(const graph::AMatrix& am, int s);  void init(const graph::AList& al, int s);  int get(); // получить следующую вершину  };  // --- Graph.cpp  //  #include "Graph.h"  namespace graph  {  AMatrix::AMatrix(int n)  {  this->n\_vertex = n;  this->mr = new int[this->n\_vertex \* this->n\_vertex];  for (int i = 0; i < n \* n; i++)mr[i] = 0;  };  AMatrix::AMatrix(int n, int mr[])  {  this->n\_vertex = n;  this->mr = mr;  };  AMatrix::AMatrix(const AMatrix& am)  {  this->n\_vertex = am.n\_vertex;  this->mr = new int[this->n\_vertex \* this->n\_vertex];  for (int i = 0; i < this->n\_vertex; i++)  for (int j = 0; j < this->n\_vertex; j++)  this->set(i, j, am.get(i, j));  };  AMatrix::AMatrix(const AList& al)  {  this->n\_vertex = al.n\_vertex;  this->mr = new int[this->n\_vertex \* this->n\_vertex];  for (int k = 0; k < this->n\_vertex \* this->n\_vertex; k++)mr[k] = 0;  for (int i = 0; i < this->n\_vertex; i++)  for (int j = 0; j < al.size(i); j++) this->set(i, al.get(i, j), 1);  };  void AMatrix::set(int i, int j, int r) { this->mr[i \* this->n\_vertex + j] = r; };  int AMatrix::get(int i, int j)const  {  return this->mr[i \* this->n\_vertex + j];  };  void AList::create(int n)  {  this->mr = new std::list<int>[this->n\_vertex = n];  };  AList::AList(int n) { create(n); }  AList::AList(const AMatrix& am)  {  this->create(am.n\_vertex);  for (int i = 0; i < this->n\_vertex; i++)  for (int j = 0; j < this->n\_vertex; j++)  if (am.get(i, j) != 0) this->add(i, j);  };  AList::AList(const AList& al)  {  this->create(al.n\_vertex);  for (int i = 0; i < this->n\_vertex; i++)  for (int j = 0; j < al.size(i); j++) this->add(i, al.get(i, j));  };  AList::AList(int n, int mr[])  {  this->create(n);  for (int i = 0; i < this->n\_vertex; i++)  for (int j = 0; j < this->n\_vertex; j++)  if (mr[i \* this->n\_vertex + j] != 0) this->add(i, j);  };  void AList::add(int i, int j) { this->mr[i].push\_back(j); };  int AList::size(int i) const { return (int)this->mr[i].size(); };  int AList::get(int i, int j)const  {  std::list<int>::iterator rc = this->mr[i].begin();  for (int k = 0; k < j; k++) rc++;  return (int)\*rc;  };  };// --- Graph.cpp  //  #include "Graph.h"  namespace graph  {  AMatrix::AMatrix(int n)  {  this->n\_vertex = n;  this->mr = new int[this->n\_vertex \* this->n\_vertex];  for (int i = 0; i < n \* n; i++)mr[i] = 0;  };  AMatrix::AMatrix(int n, int mr[])  {  this->n\_vertex = n;  this->mr = mr;  };  AMatrix::AMatrix(const AMatrix& am)  {  this->n\_vertex = am.n\_vertex;  this->mr = new int[this->n\_vertex \* this->n\_vertex];  for (int i = 0; i < this->n\_vertex; i++)  for (int j = 0; j < this->n\_vertex; j++)  this->set(i, j, am.get(i, j));  };  AMatrix::AMatrix(const AList& al)  {  this->n\_vertex = al.n\_vertex;  this->mr = new int[this->n\_vertex \* this->n\_vertex];  for (int k = 0; k < this->n\_vertex \* this->n\_vertex; k++)mr[k] = 0;  for (int i = 0; i < this->n\_vertex; i++)  for (int j = 0; j < al.size(i); j++) this->set(i, al.get(i, j), 1);  };  void AMatrix::set(int i, int j, int r) { this->mr[i \* this->n\_vertex + j] = r; };  int AMatrix::get(int i, int j)const  {  return this->mr[i \* this->n\_vertex + j];  };  void AList::create(int n)  {  this->mr = new std::list<int>[this->n\_vertex = n];  };  AList::AList(int n) { create(n); }  AList::AList(const AMatrix& am)  {  this->create(am.n\_vertex);  for (int i = 0; i < this->n\_vertex; i++)  for (int j = 0; j < this->n\_vertex; j++)  if (am.get(i, j) != 0) this->add(i, j);  };  AList::AList(const AList& al)  {  this->create(al.n\_vertex);  for (int i = 0; i < this->n\_vertex; i++)  for (int j = 0; j < al.size(i); j++) this->add(i, al.get(i, j));  };  AList::AList(int n, int mr[])  {  this->create(n);  for (int i = 0; i < this->n\_vertex; i++)  for (int j = 0; j < this->n\_vertex; j++)  if (mr[i \* this->n\_vertex + j] != 0) this->add(i, j);  };  void AList::add(int i, int j) { this->mr[i].push\_back(j); };  int AList::size(int i) const { return (int)this->mr[i].size(); };  int AList::get(int i, int j)const  {  std::list<int>::iterator rc = this->mr[i].begin();  for (int k = 0; k < j; k++) rc++;  return (int)\*rc;  };  };  #include "BFS.h"  void BFS::init(const graph::AList& al, int s)  {  this->al = &al;  this->c = new Color[this->al->n\_vertex];  this->d = new int[this->al->n\_vertex];  this->p = new int[this->al->n\_vertex];  for (int i = 0; i < this->al->n\_vertex; i++)  {  this->c[i] = WHITE;  this->d[i] = INF;  this->p[i] = NIL;  };  this->c[s] = GRAY;  this->q.push(s);  };  BFS::BFS(const graph::AList& al, int s) { this->init(al, s); };  BFS::BFS(const graph::AMatrix& am, int s)  {  this->init(\*(new graph::AList(am)), s);  };  int BFS::get()  {  int rc = NIL, v = NIL;  if (!this->q.empty())  {  rc = this->q.front();  for (int j = 0; j < this->al->size(rc); j++)  if (this->c[v = this->al->get(rc, j)] == WHITE)  {  this->c[v] = GRAY;  this->d[v] = this->d[rc] + 1;  this->p[v] = rc;  this->q.push(v);  };  this->q.pop();  this->c[rc] = BLACK;  };  return rc;  } |

Результат работы



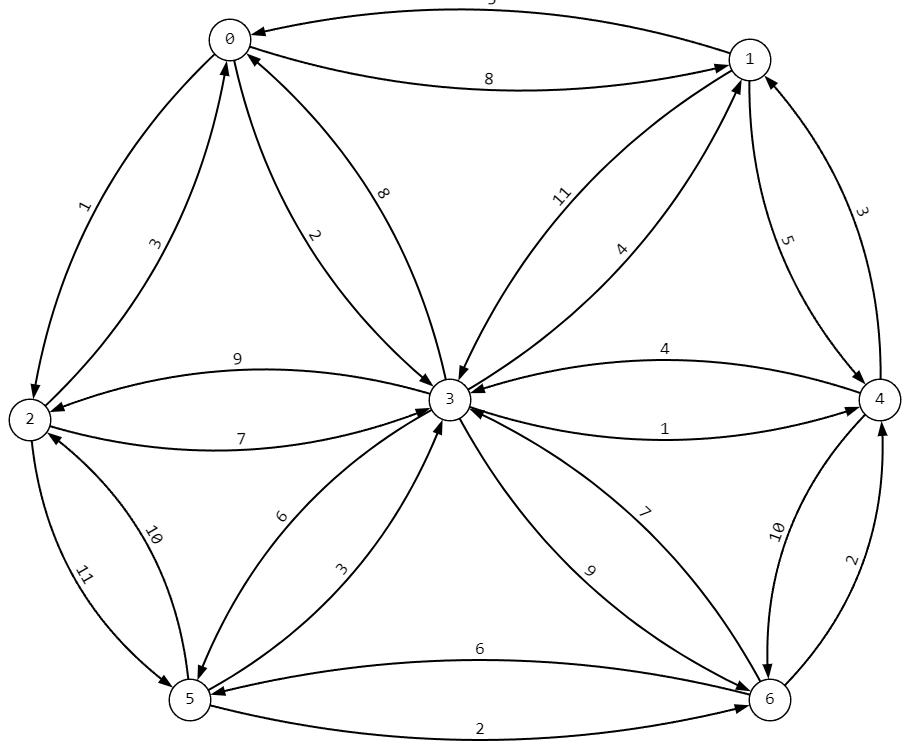
Задание 4

|  |
| --- |
| #pragma once  // ---DFS.h  //  #pragma once  #include "Graph.h"  #include <vector>  struct DFS // depth-first search поиск в глубину  {  const static int NIL = -1;  enum Color { WHITE, GRAY, BLACK }; //  const graph::AList\* al; // исходный граф  Color\* c; // цвет вершины  int\* d; // время обнаружения  int\* f; // время завершения обработки  int\* p; // предшествующая вершина  int t; // текущее время  DFS(const graph::AList& al);  DFS(const graph::AMatrix& am);  std::vector <int> topological\_sort; //результат топологической сортировки  void visit(int v);  void init(const graph::AList& al);  int get(int i); // получить вершину  };  // ---DFS.cpp  //  #include "DFS.h"  #define NINF 0x80000000  #define INF 0x7fffffff  void DFS::init(const graph::AList& al)  {  this->al = &al;  this->c = new Color[this->al->n\_vertex];  this->d = new int[this->al->n\_vertex];  this->f = new int[this->al->n\_vertex];  this->p = new int[this->al->n\_vertex];  this->t = 0;  for (int i = 0; i < this->al->n\_vertex; i++)  {  this->c[i] = WHITE;  this->d[i] = this->f[i] = 0;  this->p[i] = NIL;  };  for (int i = 0; i < this->al->n\_vertex; i++)  if (this->c[i] == WHITE)  {  this->visit(i);  this->topological\_sort.push\_back(i);  }  };  DFS::DFS(const graph::AList& al) { this->init(al); };  DFS::DFS(const graph::AMatrix& am)  {  this->init(\*(new graph::AList(am)));  };  void DFS::visit(int u)  {  int v = NIL;  this->c[u] = GRAY;  this->d[u] = ++(this->t);  for (int j = 0; j < this->al->size(u); j++)  if (this->c[v = this->al->get(u, j)] == WHITE)  {  this->p[v] = u;  this->visit(v);  this->topological\_sort.push\_back(v);  }  this->c[u] = BLACK;  this->f[u] = ++(this->t);  };  int DFS::get(int i)  {  int j = 0, min1 = INF, min2 = NINF, ntx = NIL;  for (int j = 0; j <= i; j++) // iая статистика  {  for (int k = 0; k < this->al->n\_vertex; k++)  if (this->f[k] < min1 && this->f[k] > min2)  {  min1 = this->f[k]; ntx = k;  };  min2 = min1; min1 = INF;  };  return ntx;  }; |

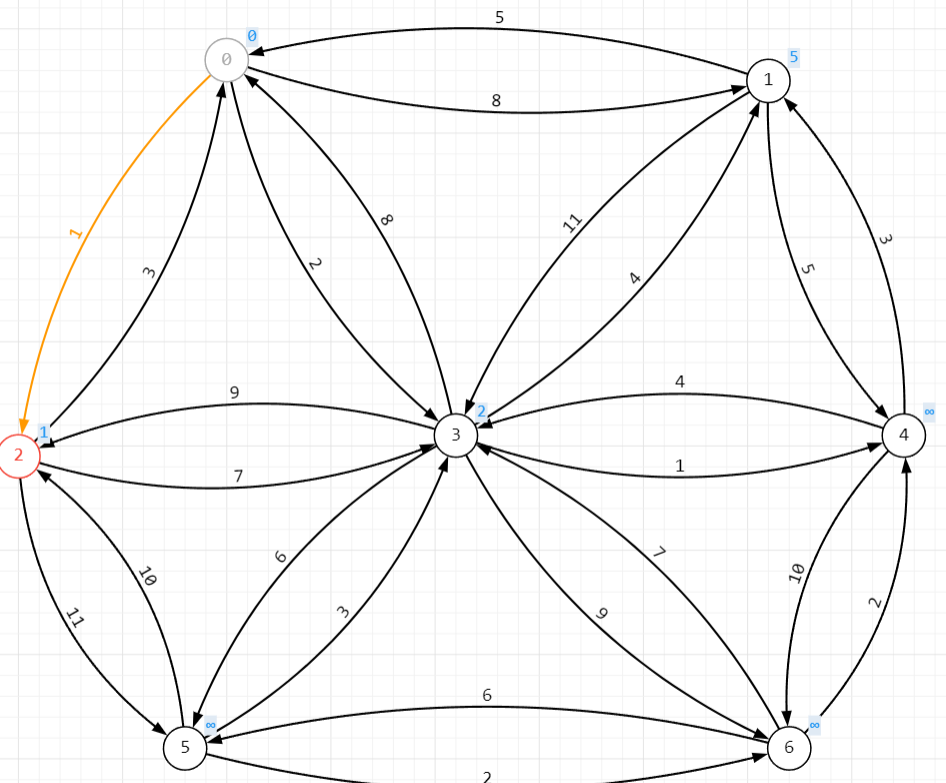
Задание 5

|  |
| --- |
| void topologicalSort(int v, bool visited[], std::stack<int>& Stack, graph::AList& g)  {  // Помечаем текущую вершину как посещенную  visited[v] = true;  // Рекурсивно вызываем функцию для всех смежных вершин  // Если смежная вершина не была посещена, то рекурсивно вызываем функцию  for (int i = 0; i < g.size(v); i++)  if (!visited[g.get(v, i)])  topologicalSort(g.get(v, i), visited, Stack, g);  // Помещаем текущую вершину в стек  Stack.push(v);  } |

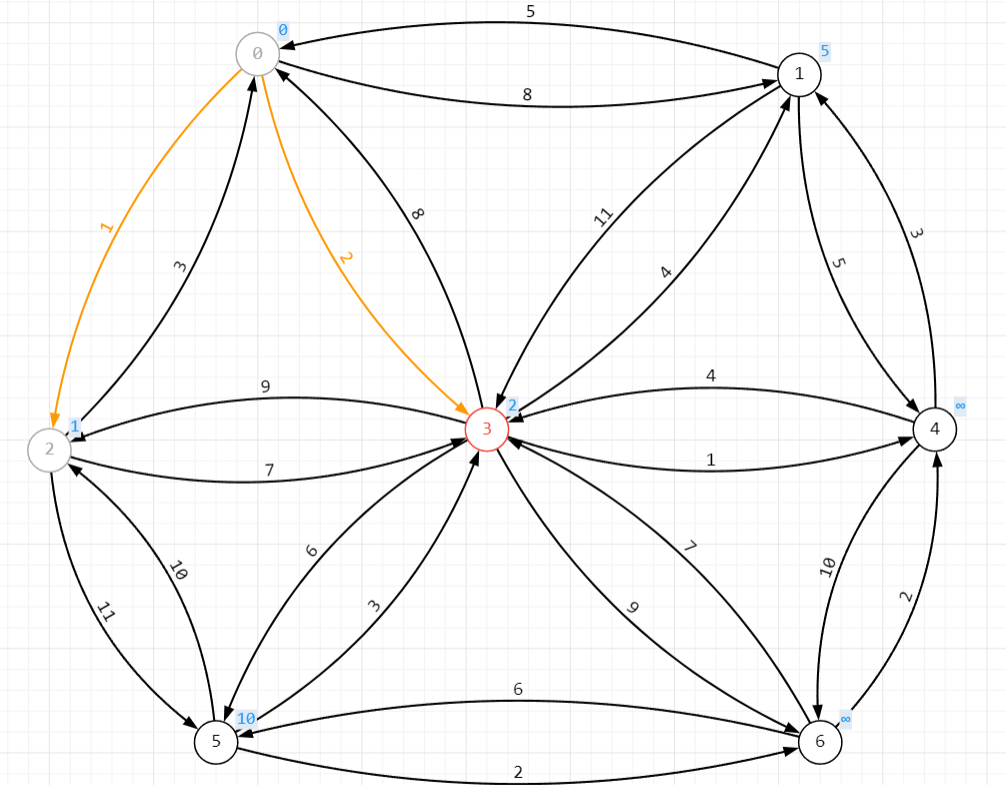
Задание 6



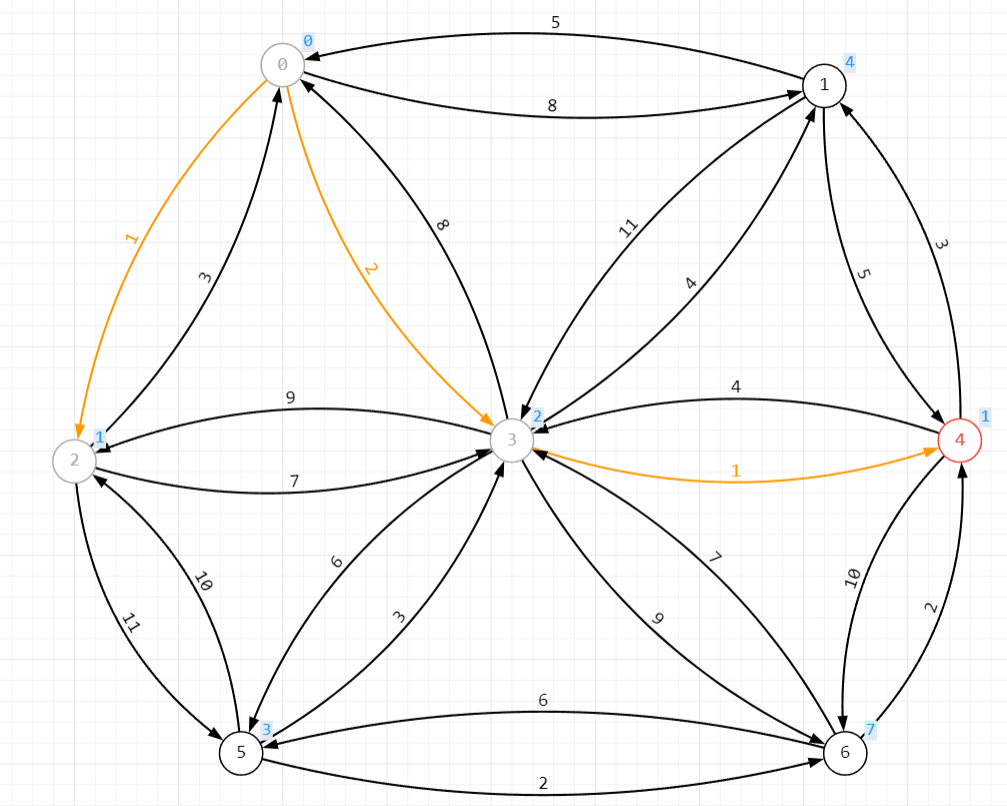
Начинаем с 0, проверяем все соседние вершины, находим минимальный путь



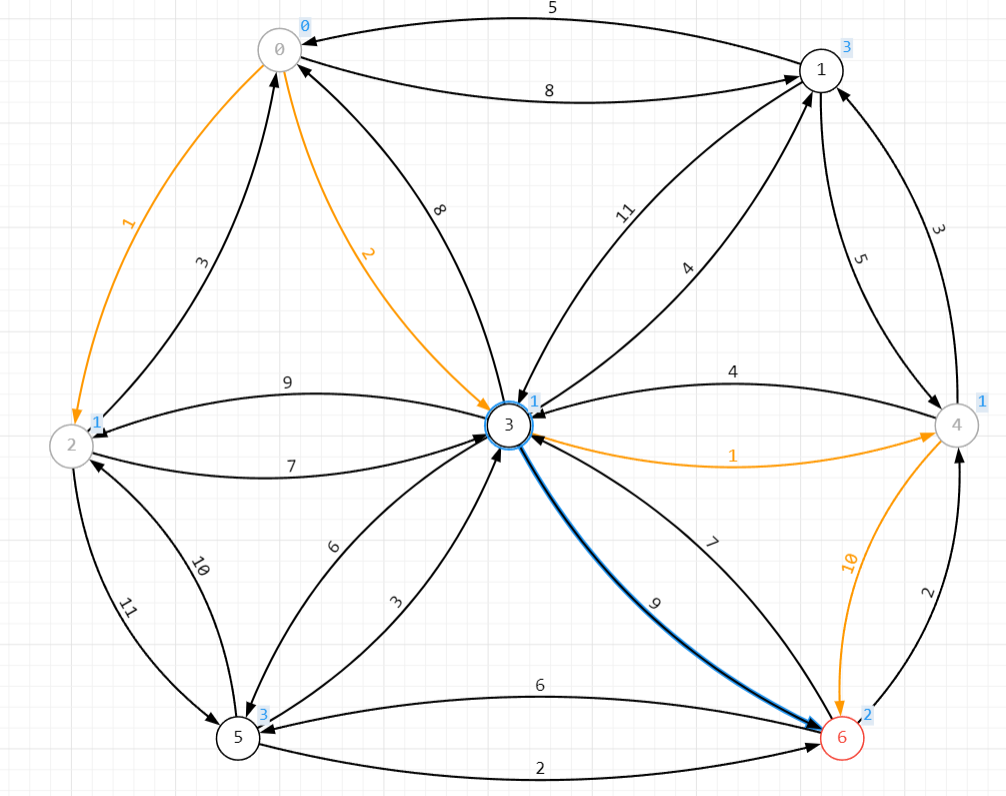
Проверяем все смежные вершины с вершиной2. В случае ненахода, возвращаемся обратно



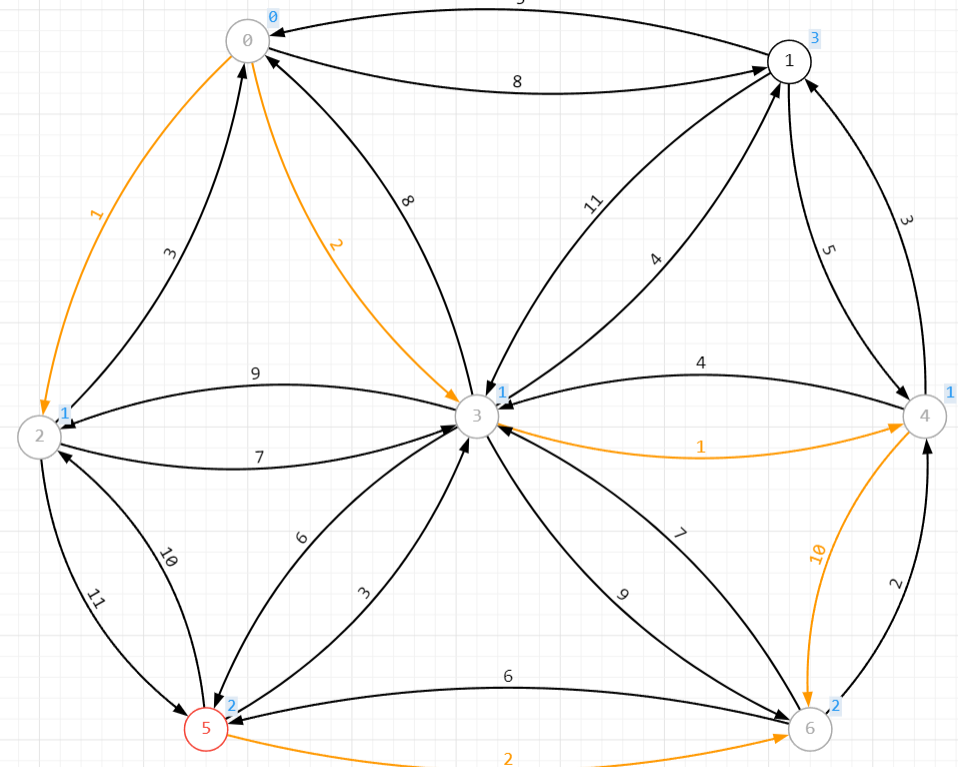
Проверяем все смежные вершины с вершиной3



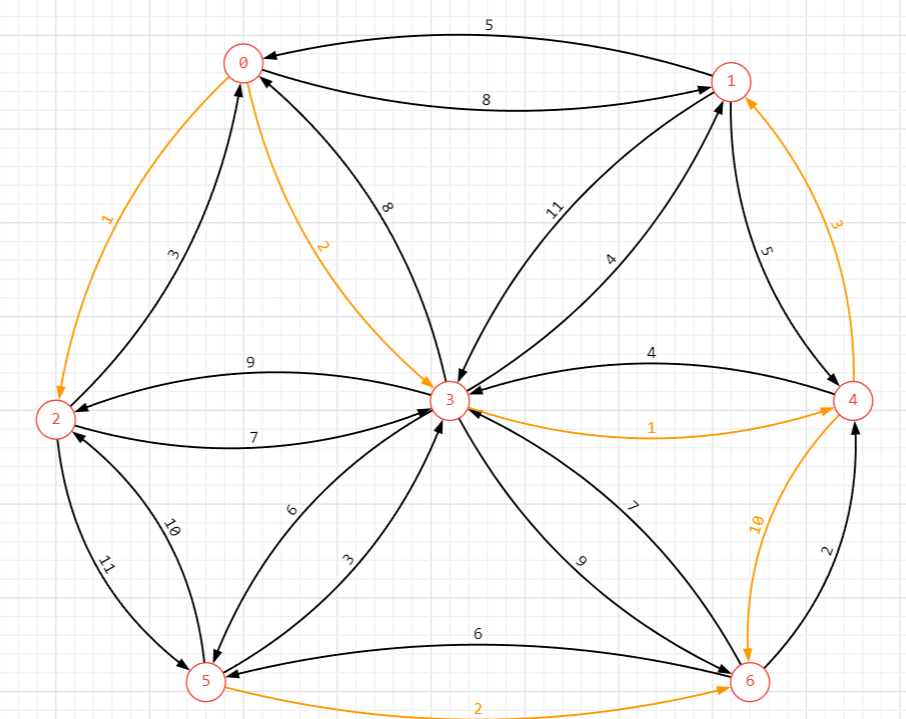
Проверяем все смежные вершины с вершиной4



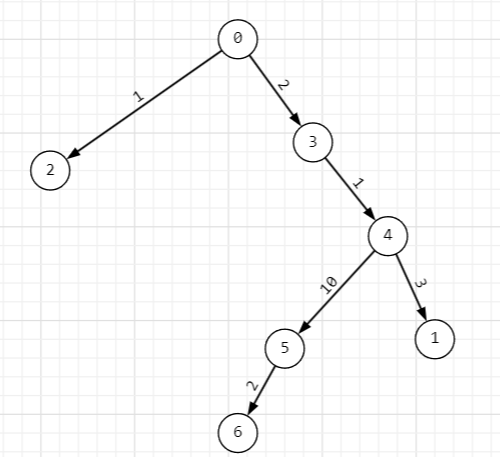
Проверяем все смежные вершины с вершиной6



Проверяем все смежные вершины с вершиной5, но в итоге будем смотреть с вершины4

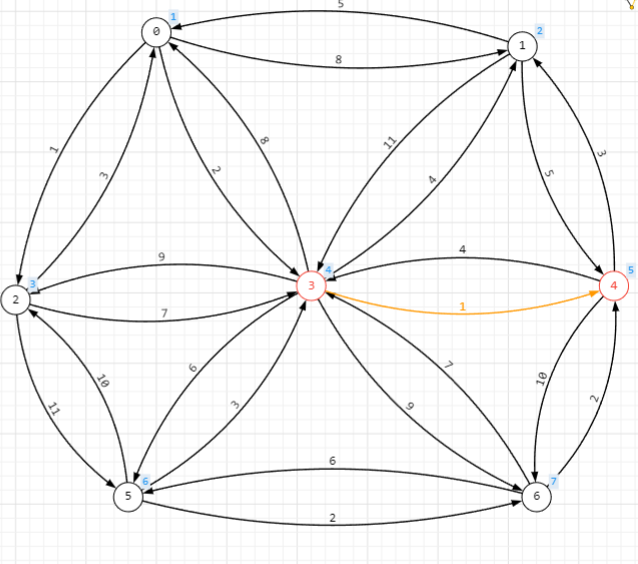


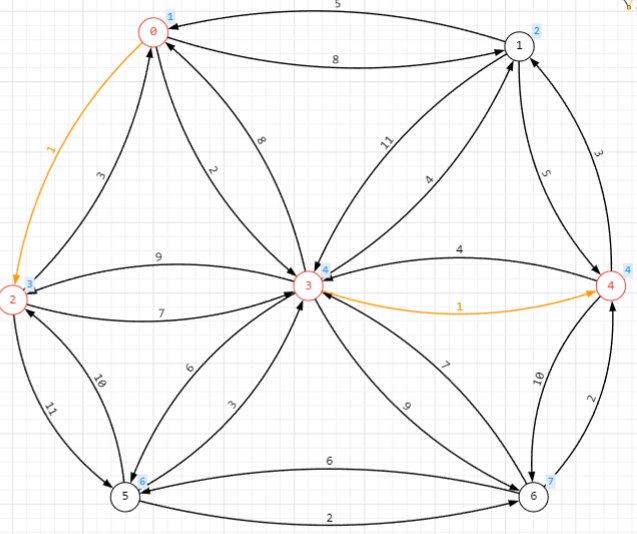
Остовное дерево

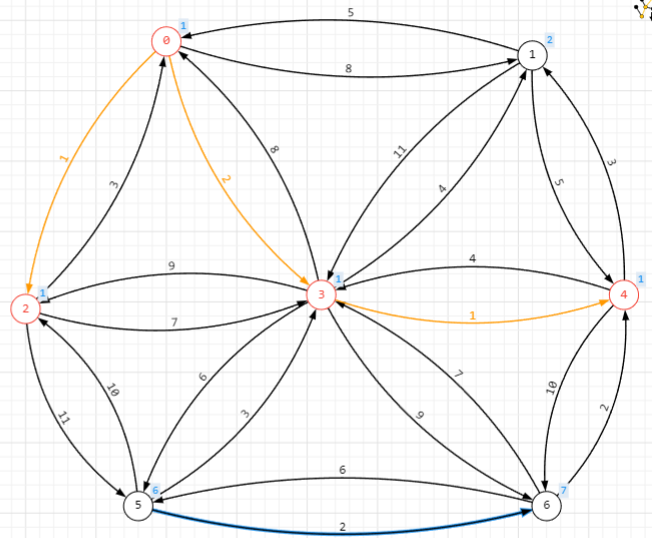


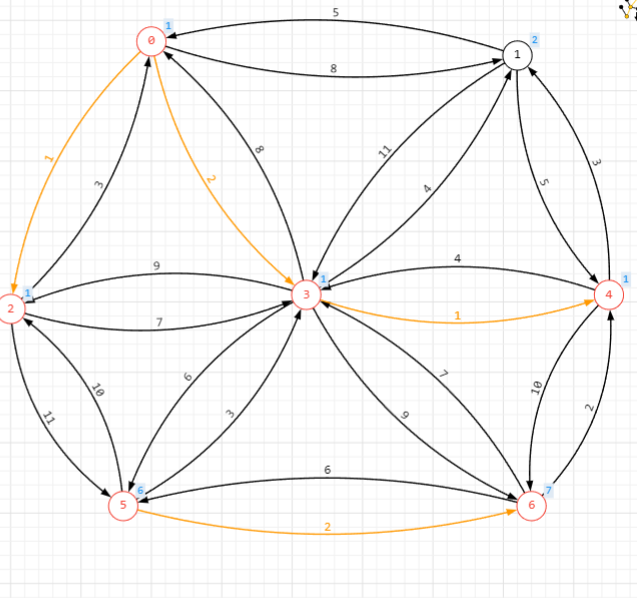
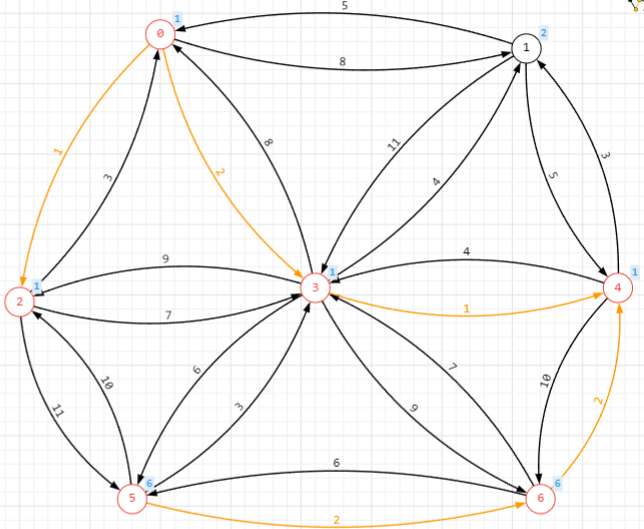
Задание 6

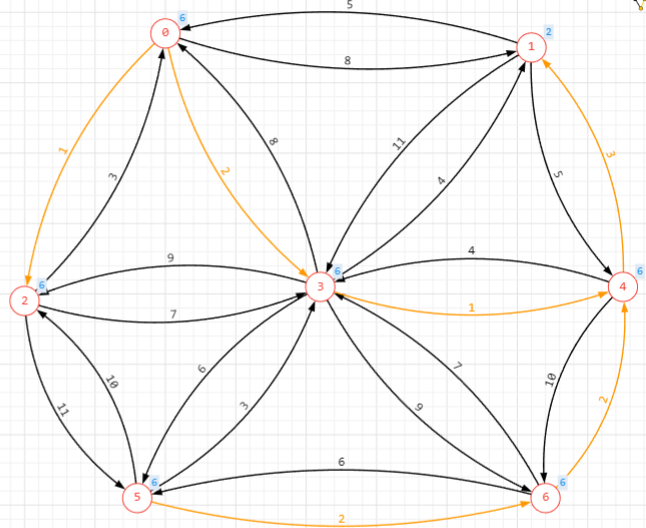
Сортируем рёбра по убыванию веса, помечаем каждую вершину уникальной меткой



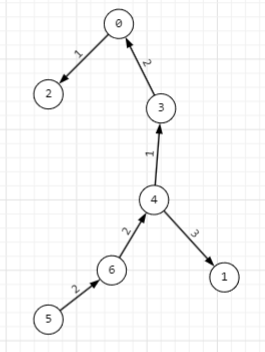






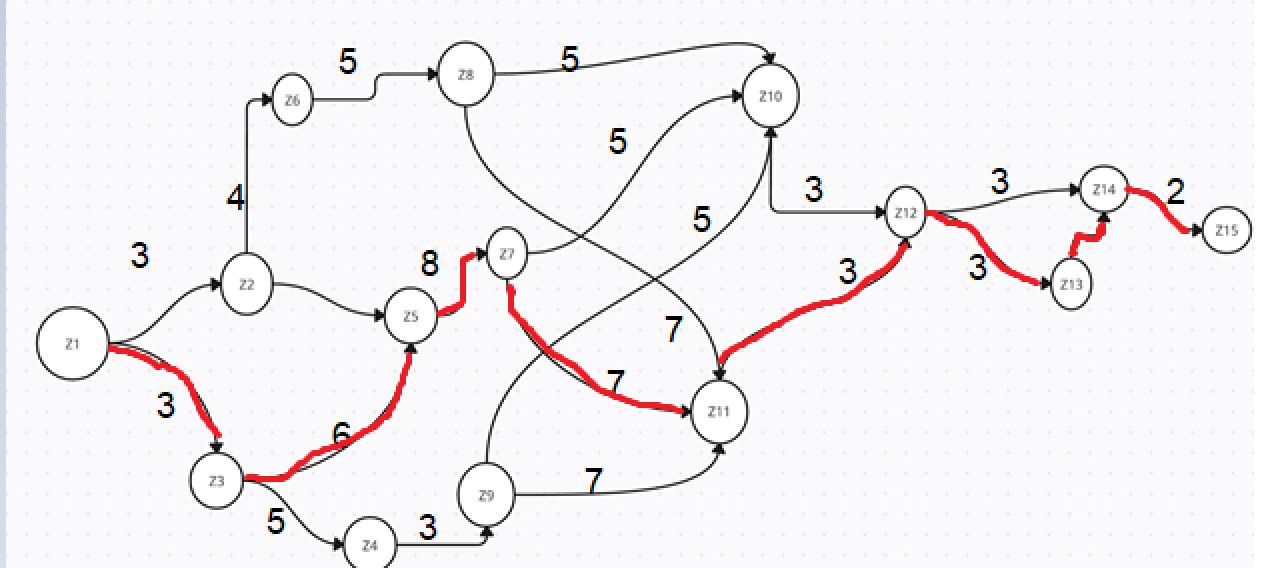
Оставное дерево



Лабораторная работа №7  
«Создание десктопного приложения»

65 дней

|  |  |  |  |
| --- | --- | --- | --- |
| Код  операции | Наименование операции | Предшествующие операции | t |
| I. Этап планирования и анализа | | | |
| Z1 | Сбор и анализ требований |  | 5 |
| Z2 | Определение функциональных возможностей приложения | Z1 | 3 |
| Z3 | Выбор технологий и инструментов | Z1 | 3 |
| II. ПРОЕКТИРОВАНИЕ | | | |
| Z4 | Проектирование базы данных | Z3 | 5 |
| Z5 | Проектирование архитектуры приложения | Z2, Z3 | 6 |
| Z6 | Проектирование интерфейсов пользователей | Z2 | 4 |
| III. Этап разработки | | | |
| Z7 | Реализация основных функций приложения | Z5 | 8 |
| Z8 | Реализация пользовательского интерфейса | Z6 | 5 |
| Z9 | Интеграция с базой данных | Z4 | 3 |
| IV. ТЕСТИРОВАНИЕ | | | |
| Z10 | Функциональное тестирование | Z7, Z8, Z9 | 5 |
| Z11 | Структурное тестирование | Z7, Z8, Z9 | 7 |
| V. Развертывание | | | |
| Z12 | Создание инсталляционного пакета | Z10, Z11 | 3 |
| Z13 | Развертывание приложения | Z12 | 3 |
| Z14 | Испытание | Z12, Z13 | 3 |
| Z15 | Подготовка документации | Z14 | 2 |



Лабораторная работа №8

Max min Z=4x­1 +3x2

|  |  |  |  |
| --- | --- | --- | --- |
| X1 | 0 | 10 | 1\*0+2\*0<=10 верно |
| X2 | 5 | 0 |  |
| X1 | 0 | 2 | 1\*0+2\*0>=2ложь |
| X2 | 1 | 0 |  |
| X1 | 0 | 5 | 2\*0+1\*0<=10верно |
| X2 | 10 | 0 |  |

Grad Z={4;3}=ċ

Zmax=Z(3,7;3,8)=4\*3,7+3\*2,8=23,2 A=(3,7;2,8)

Zmin=Z(0,9;0,6)=5,4 B=(0,9;0,6)

