Міністерство освіти та науки України

Вінницький національний технічний університет

Факультет інформаційних технологій та комп’ютерної інженерії

Кафедра комп’ютерних наук

Лабораторна робота №8

З дисципліни: «Дискретна математика»

Тема: «Дослідження потоків у мережах»

Виконав студент:

групи 2КН 18-б

І курсу

Максименюк М. Р.

Перевірив викладач:

Ваховська Л. М.

Вінниця-2019

**Мета:** набути навичок побудови максимального потоку на мережі за допомогою алгоритму Форда-Фалкерсона.

|  |  |
| --- | --- |
| **Варіант №9** | http://add.coolreferat.com/tw_refs/12/11551/11551_html_6ae4525.png  Рисунок 1. Блок-схема програми побудови максимального потоку на мережі за допомогою алгоритму Форда-Фалкерсона. |

**Опис програми:** для виконання роботи було використано Dev-Cpp. Програма написана на мові С++.

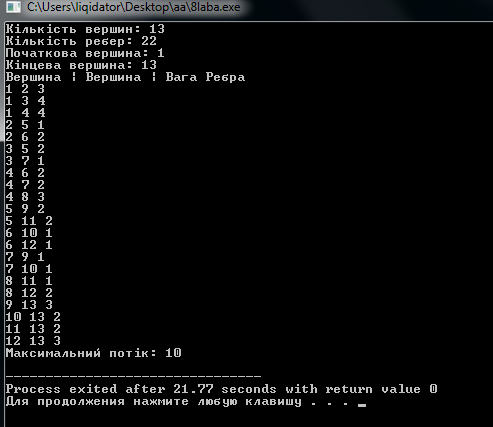


Рисунок 2 - Результат роботи програми побудови максимального потоку на мережі за допомогою алгоритму Форда-Фалкерсона

**Висновок**

Виконаннуючилабораторну роботу було виконанно практичні розрахунки максимального потоку алгоритмом Форда-Фалкерсона, а також створено програму для пошуку максимального потоку алгоритмом Форда-Фалкерсона.

**Додаток 1**

**Інструкція користувача**

1. Для запуску програми потрібно запустити Dev-Cpp, відкрити документ під назвою laba\_8 та натискаємо клавішу F11.
2. Вводимо кількість вершин. Натискаємо "Enter".
3. Вводимо витік -> стік (перша вершина - це 0). Натискаємо "Enter".
4. На екрані програма виведе максимальний потік по заданому графу.

**Додаток 2**

**Лістинг програми обчислення максимального потоку по заданому графу**

|  |  |
| --- | --- |
| #include <iostream>  #include <conio.h>  #include <memory.h>  #include <stdio.h>  using namespace std;  const int max\_vershin = 40 ;  int number\_vershin;  const int inf = 10000;  int g[max\_vershin][max\_vershin];  int k[14][14]  int fw[max\_vershin];  int link[max\_vershin];  int course[max\_vershin];  int cb, ct;  int FindPath(int head, int end)  {  cb = 0; ct = 1; course[0] = head;  link[end] = -1;  int i;  int course\_begin;  memset(fw, 0, sizeof(int)\*number\_vershin);  fw[head] = inf;  while (link[end] == -1 && cb < ct)  {  course\_begin = course[cb];  for (i=0; i<number\_vershin; i++)  (course\_begin,i):  if ((k[course\_begin][i] - g[course\_begin][i])>0 && fw[i] == 0)  {  course[ct] = i; ct++;  link[i] = course\_begin  if (k[course\_begin][i]-g[course\_begin][i] < fw[course\_begin])  fw[i] = k[course\_begin][i];  else  fw[i] = fw[course\_begin];  }  cb++;  }  if (link[end] == -1) return 0;  course\_begin = end;  while (course\_begin != head) { | g[link[course\_begin]][course\_begin] +=fw[end];  course\_begin = link[course\_begin];  }  return fw[end]; }  int max\_fw(int head, int end)  {  memset(g, 0, sizeof(int)\*max\_vershin\*max\_vershin);  int max\_fw = 0;  int add\_fw;  do  {  add\_fw = FindPath(head, end);  max\_fw += add\_fw;  } while (add\_fw >0);  return max\_fw;  }  int main()  {  setlocale(LC\_ALL,"Ukrainian");  int head, end;  cout<<"Кiлькiсть вершин: ";  scanf("%d", &number\_vershin);  cout<<"Витiк -> Стiк: ";  scanf("%d %d", &head, &end);  int i, j;  cout<<"Максимальний потiк: ";  printf("%d", max\_fw(head, end));  getch();} |