МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

НАЦІОНАЛЬНОМУ УНІВЕРСИТЕТІ “ЛЬВІВСЬКА ПОЛІТЕХНІКА”

Кафедра систем штучного інтелекту

Лабораторна робота № 5

з дисципліни

«Дискретна математика»

Виконав:

Дребот Владислав Олегович

студент групи КН-112

Викладач:

Мельникова Н.І.

Львів – 2019 р.

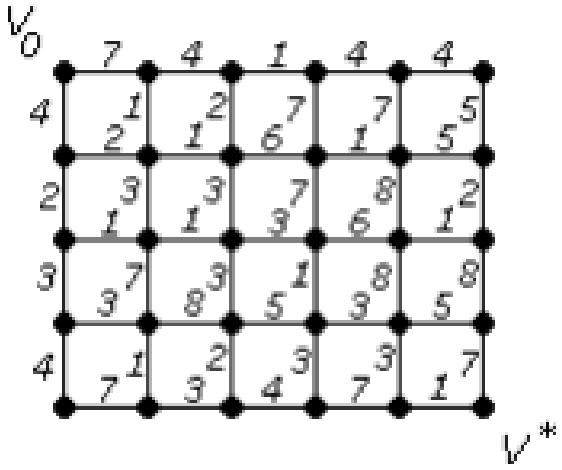
Тема: Знаходження найкоротшого маршруту за алгоритмом Дейкстри. Плоскі планарні графи

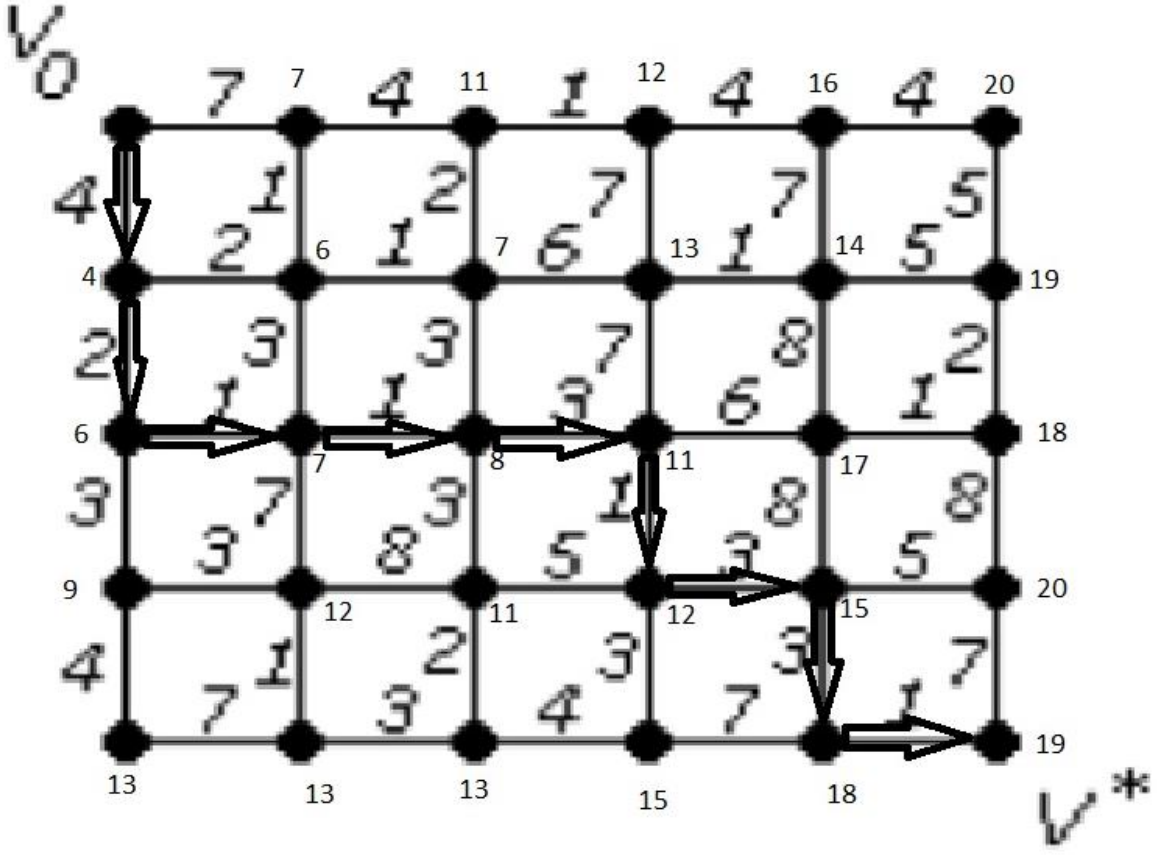
Мета роботи: набуття практичних вмінь та навичок з використання алгоритму Дейкстри.

ІНДИВІДУАЛЬНІ ЗАВДАННЯ

Завдання № 1. Розв'язати на графах наступні 2 задачі: 1. За допомогою алгоритму Дейкстра знайти найкоротший шлях у графі поміж парою вершин V0 і V * .

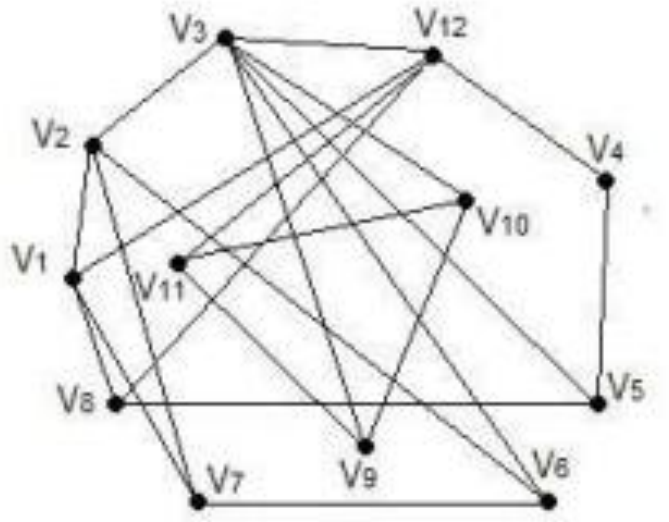
4



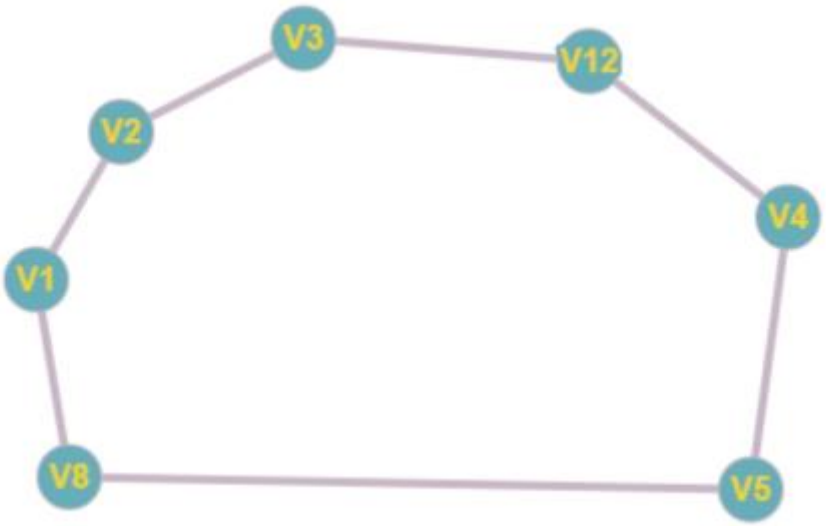


2. За допомогою γ -алгоритма зробити укладку графа у площині, або довести що вона неможлива.

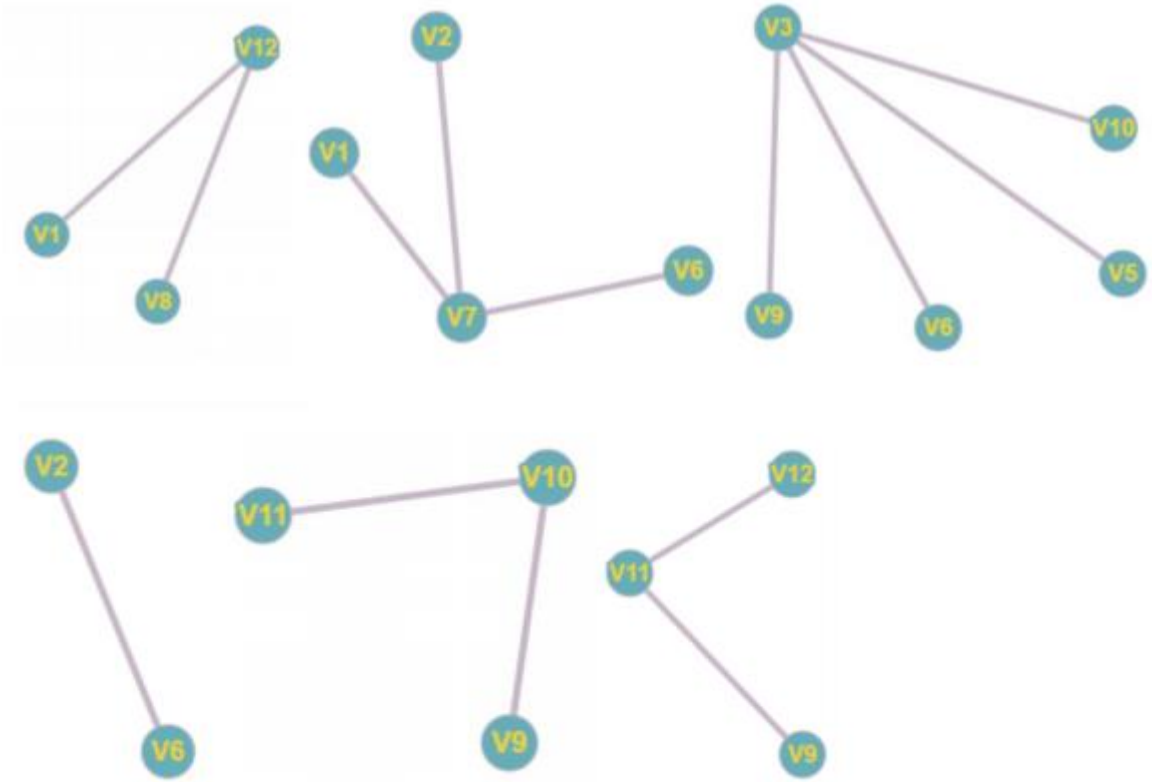
4



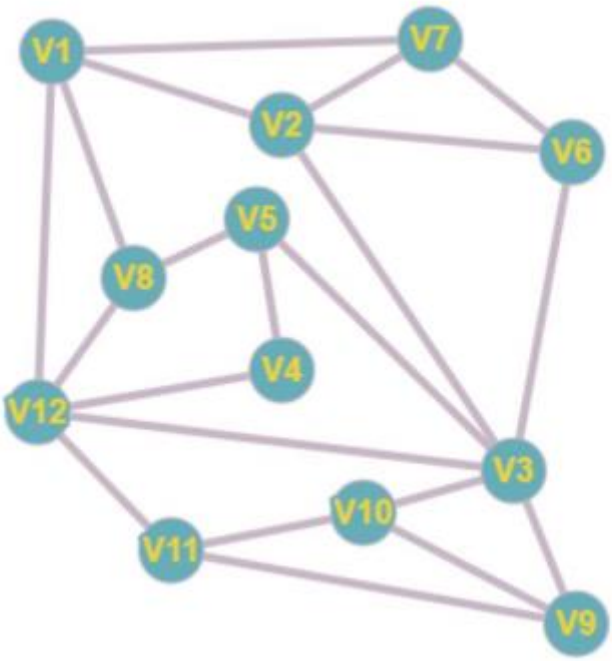
Вибираємо довільний цикл з графа:



Сегменти:



Плоский планарний граф:




```
#include <stdio.h>
#include <stdlib.h>
#define SIZE 6
```

```
int main()
{
  int a[SIZE][SIZE]; // матриця звяз.
  int d[SIZE]; // мінімальна відстань
  int v[SIZE]; // пройдені вершини
  int temp, minindex, min;
  int begin_index = 0;
  system("chcp 1251");
  system("cls");
  // ініцаілізація матриці з.
  for (int i = 0; i<SIZE; i++)
  {
    a[i][i] = 0;
    for (int j = i + 1; j<SIZE; j++) {
      printf("vvedit vidstan: %d - %d: ", i + 1, j + 1);
      scanf("%d", &temp);
      a[i][j] = temp;
      a[j][i] = temp;
    }
  }
  // вивід матриці звязку
  for (int i = 0; i<SIZE; i++)
  {
    for (int j = 0; j<SIZE; j++)
      printf("%5d ", a[i][j]);
    printf("\n");
```

```
}
//Ініціаліхація вершин і відстаней
for (int i = 0; i<SIZE; i++)
{
  d[i] = 10000;
  v[i] = 1;
}
d[begin_index] = 0;
// крок алгоритму
do {
  minindex = 10000;
  min = 10000;
  for (int i = 0; i<SIZE; i++)
  { // Якщо вершину ще не обійшли і вага меньше min
    if ((v[i] == 1) && (d[i]<min))
    { // Присвоюємо значення
      min = d[i];
      minindex = i;
    }
  }
  // Добавляю знайдений мінімальну вагу
  // до поточної ваги вершини
  // і порівнюєм з поточною мінімальною вагою вершини
  if (minindex != 10000)
  {
    for (int i = 0; i<SIZE; i++)
```

```
    {
      if (a[minindex][i] > 0)
      {
        temp = min + a[minindex][i];
        if (temp < d[i])
        {
          d[i] = temp;
        }
      }
    }
    v[minindex] = 0;
  }
} while (minindex < 10000);
// Вивід найкоротших відстаней до вершин
printf("\nNaykorotsha vidstan do vershin: \n");
for (int i = 0; i<SIZE; i++)
  printf("%5d ", d[i]);

// Відновлення шляху
int ver[SIZE]; // масив пройдених вершин
int end = 4; // індекс кінцевої вершини = 5 - 1
ver[0] = end + 1; // початковий елемент - кінцева вершина
int k = 1; // індекс попередньої вершини
int weight = d[end]; // вага кінцевої вершини

while (end != begin_index) // поки не дійшов до початкової вершини
```

```
  {
    for (int i = 0; i<SIZE; i++) // переглядаю всі вершини
      if (a[end][i] != 0)   // якщо звязок є
      {
        int temp = weight - a[end][i]; // знаходимо вагу шляху з попередньої вершини
        if (temp == d[i]) // якщо шлях співпав з розрахованим
        {               // значить з цієї вершини був перехід
          weight = temp; // зберігаю нову вагу
          end = i;       // зберігаю попередню вершину
          ver[k] = i + 1; // і записую її в масив
          k++;
        }
      }
  }
  // Вивід шляху (початоква вершина опинилась в кінці масива з k елементів)
  printf("\nVyvid naymenshogo shliahu\n");
  for (int i = k - 1; i >= 0; i--)
    printf("%3d ", ver[i]);
  getchar(); getchar();
  return 0;
}
```