МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ НАЦІОНАЛЬНИЙ

УНІВЕРСИТЕТ “ЛЬВІВСЬКА ПОЛІТЕХНІКА”

Кафедра систем штучного інтелекту

Лабораторна робота №5

з дисципліни

“Дискретна математика”

Виконав:

студент групи КН-109

Коржов Володимир

Викладач:

Мельникова Н.І.

Львів – 2018

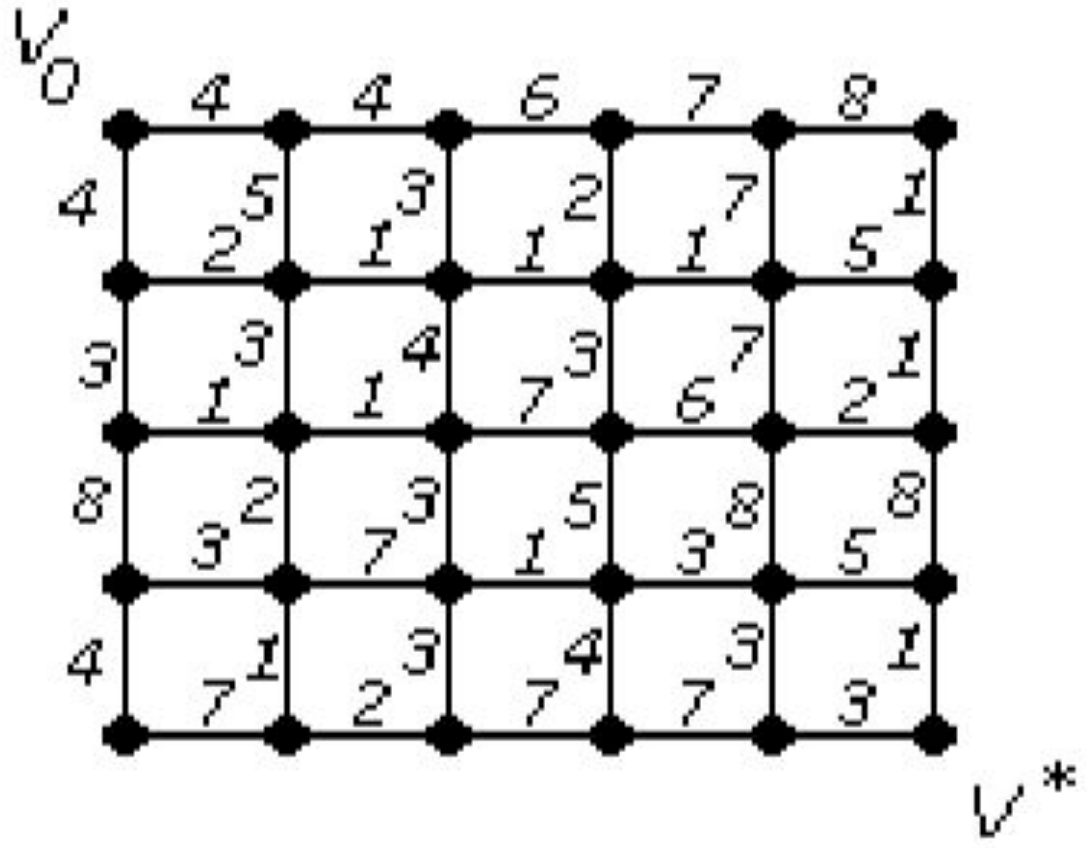
Знаходження найкоротшого маршруту за алгоритмом Дейкстри. Плоскі планарні графи

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

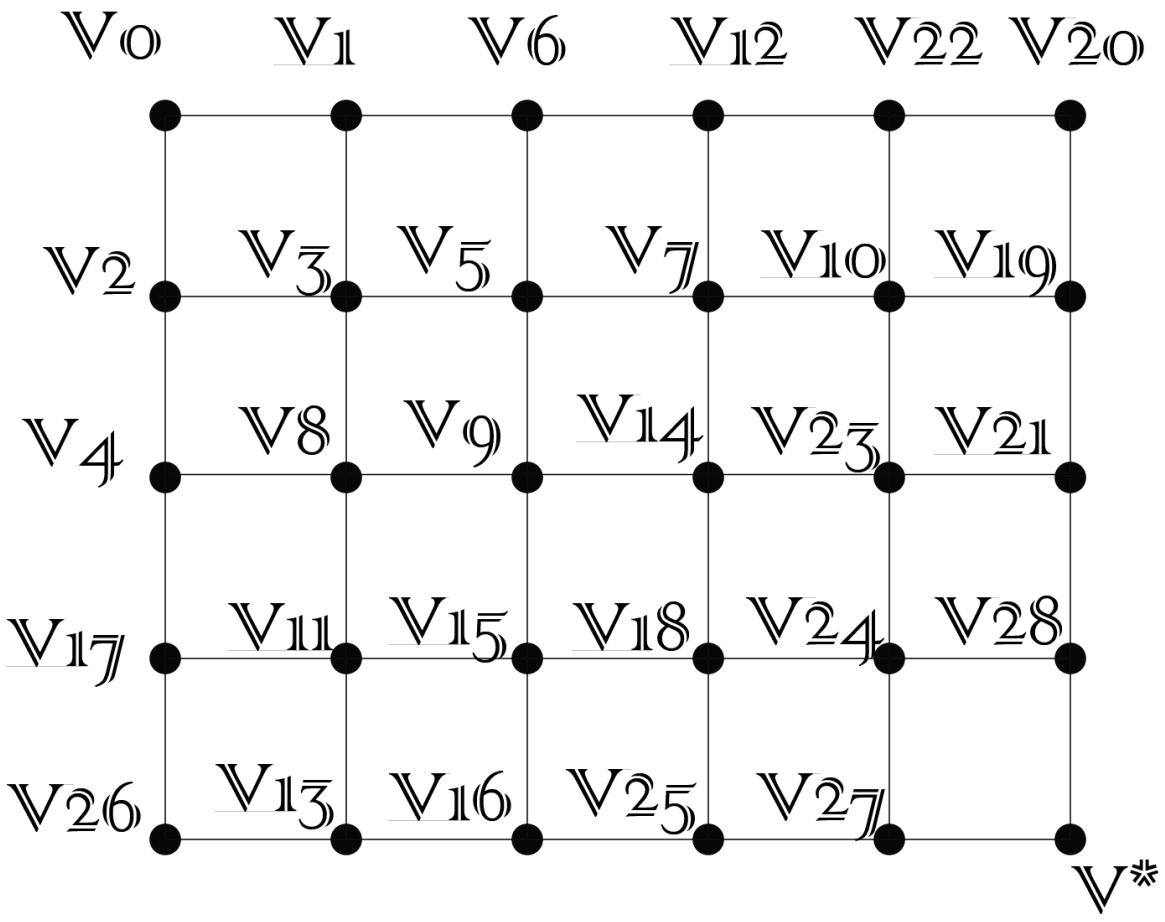
Варіант №7

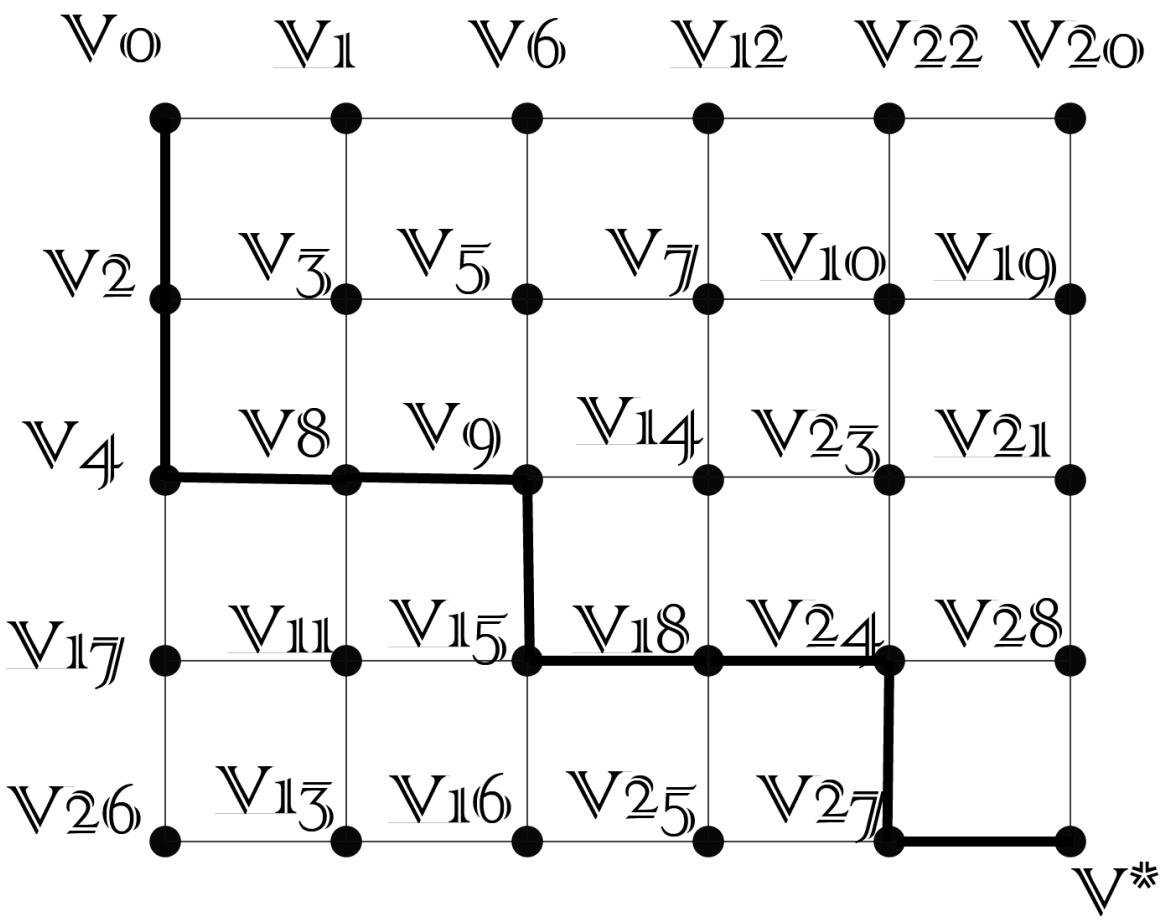
Завдання № 1.

1. За допомогою алгоритму Дейкстра знайти найкоротший шлях у графі поміж парою вершин V0 і V* .



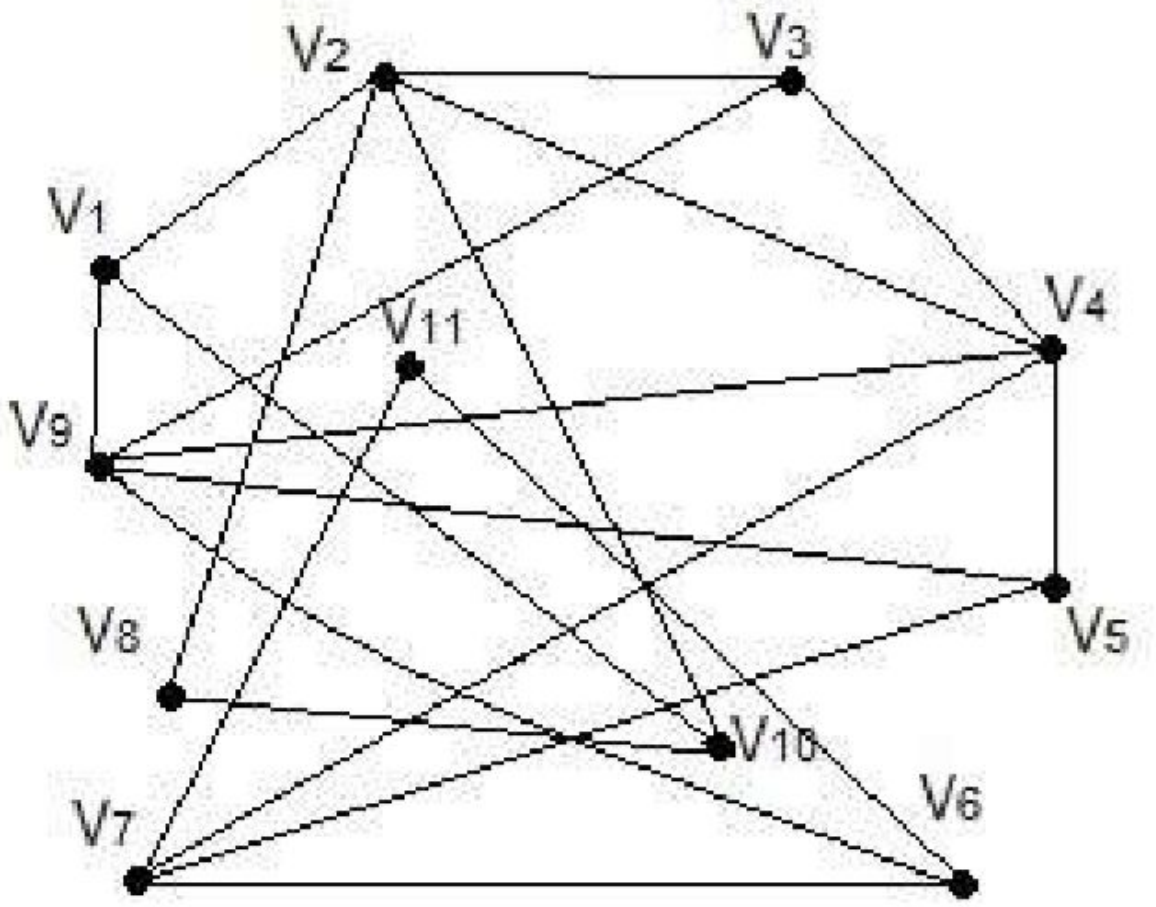
Позначимо вершини у порядку їхньої появи:



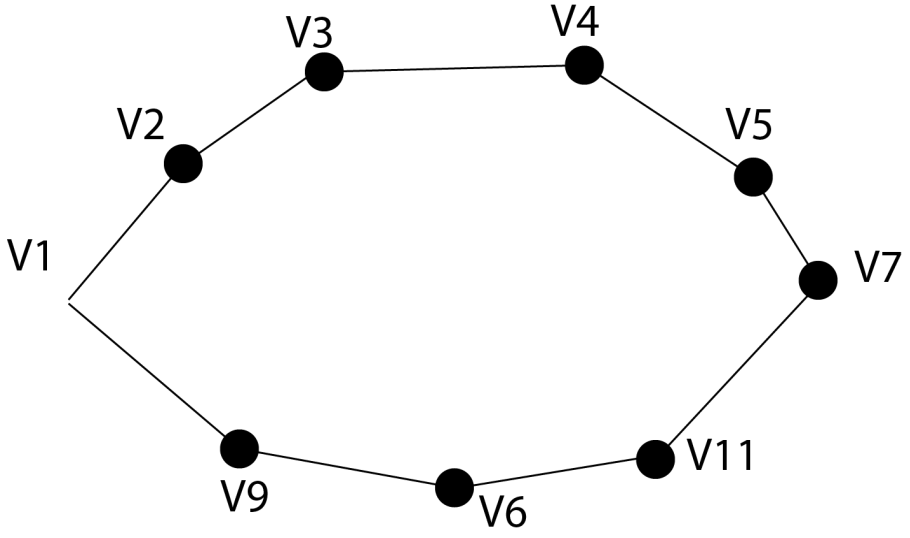


Шуканий ланцюг =
(V0-V2-V4-V8-V9-V15-V18-V24-V27-V*) = 22

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

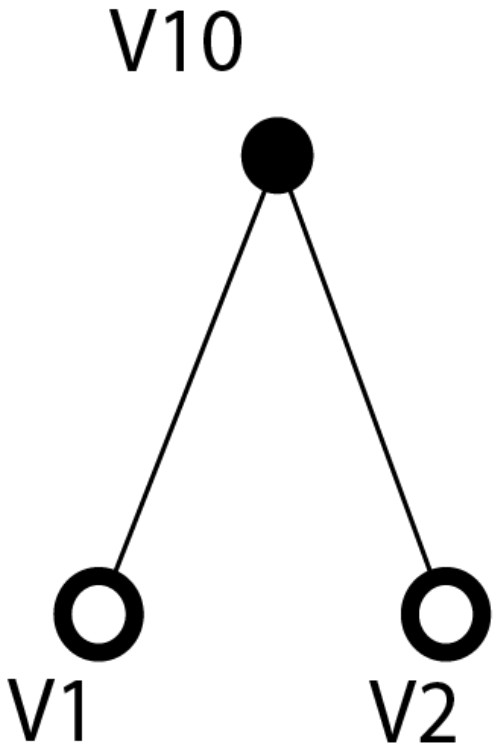


Укладемо цикл C =
[V1-V2-V3-V4-V5-V7-V11-V6-V9-V1]:

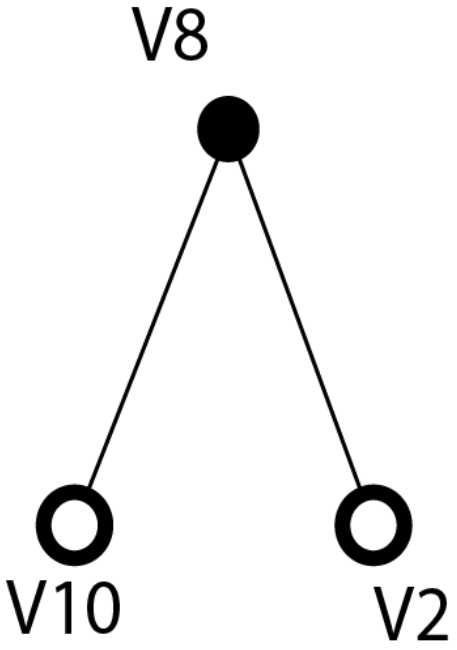


Та сегменти:

S1:



S2:



S3:

S4:

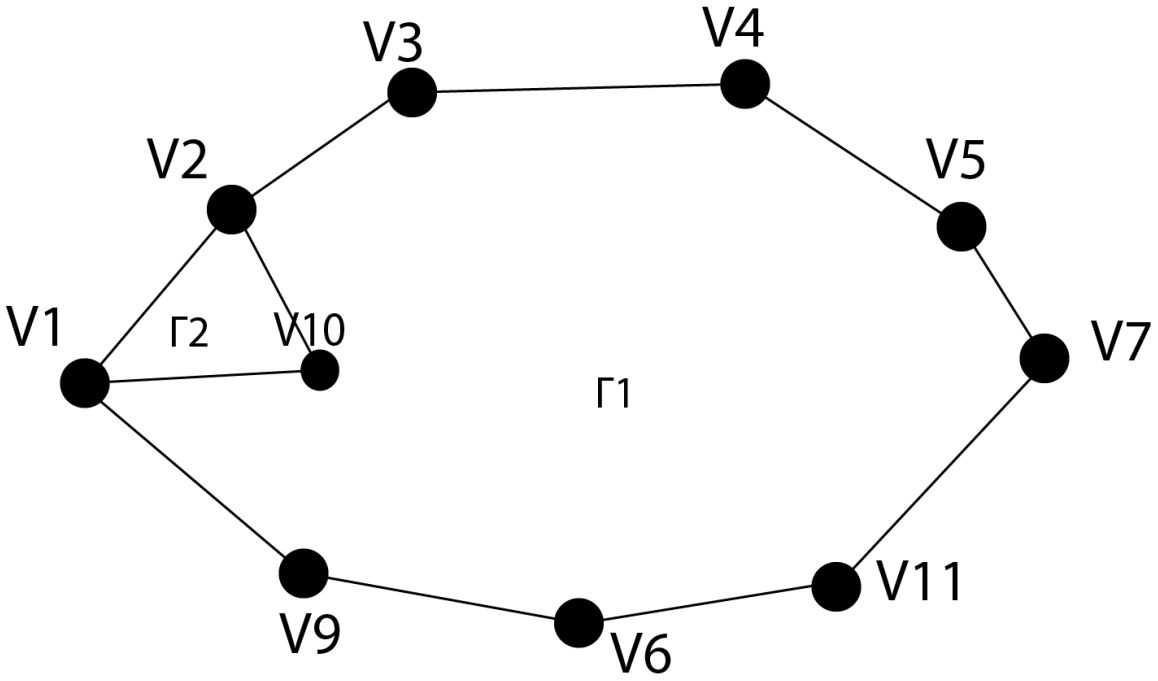
S5:

S6:

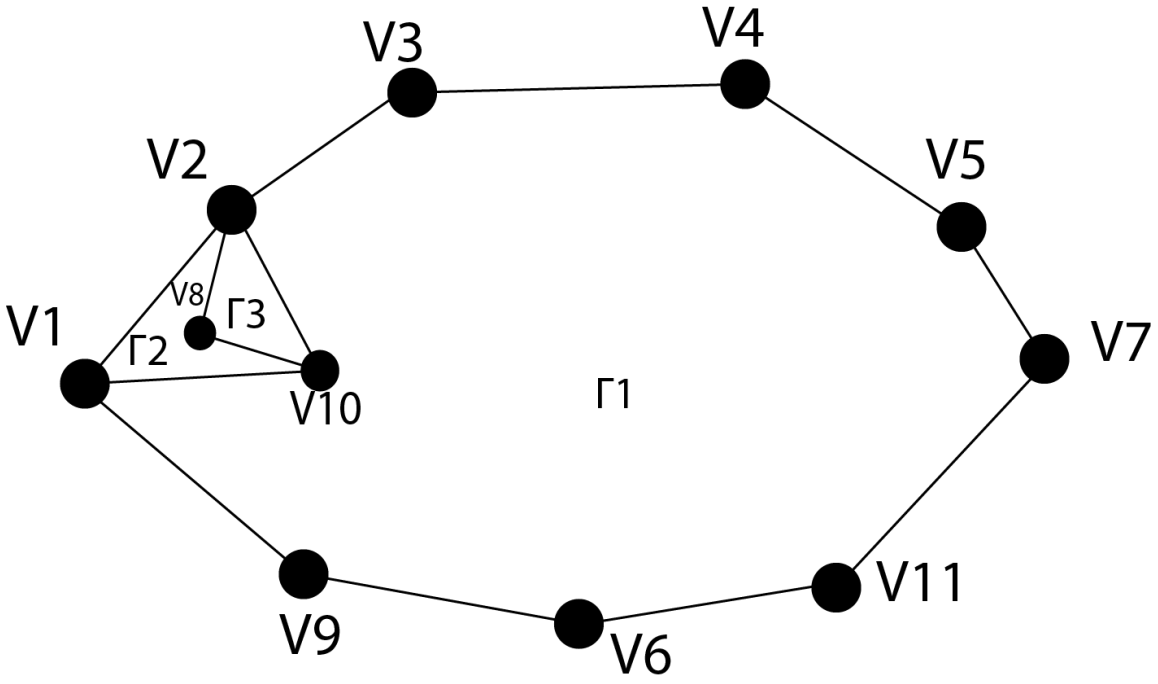
S7:

S8:

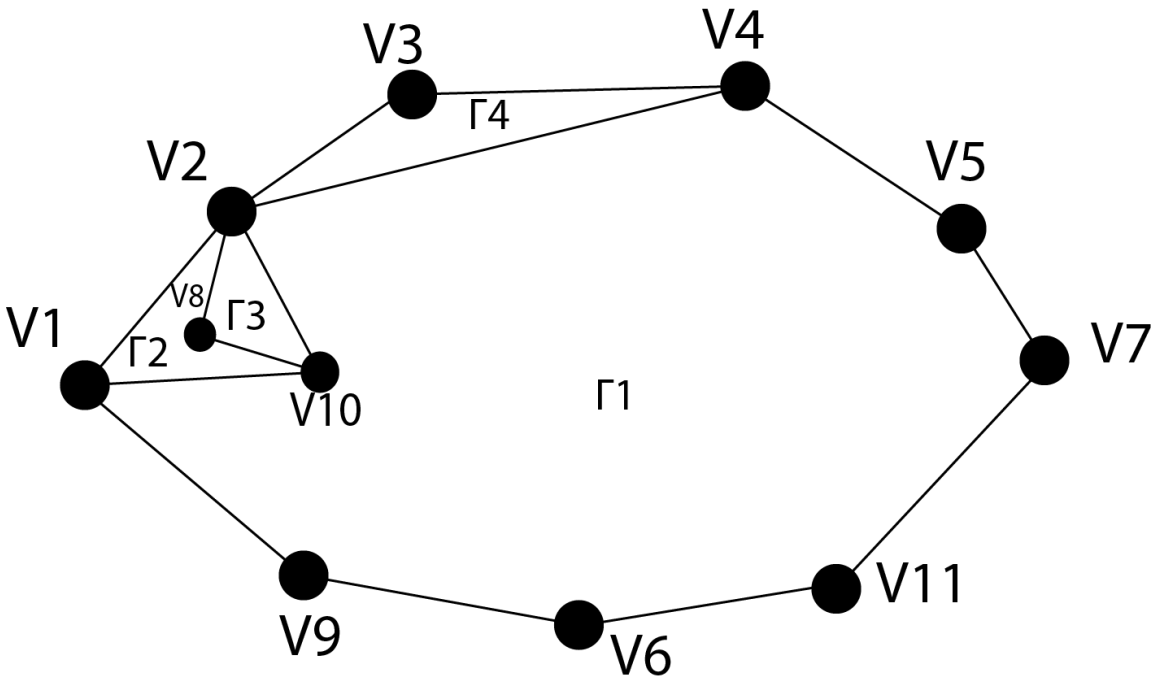
Вставимо ланцюг S1 в грань 1 графа:



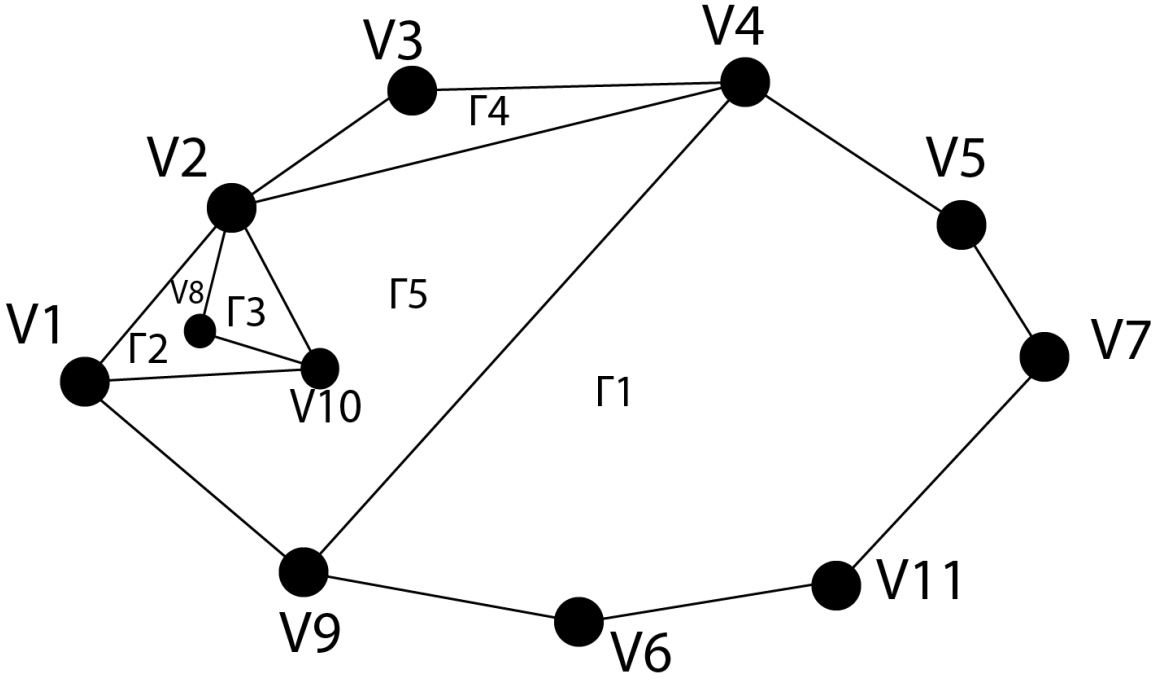
Потім в Г2 вставимо ланцюг S2:



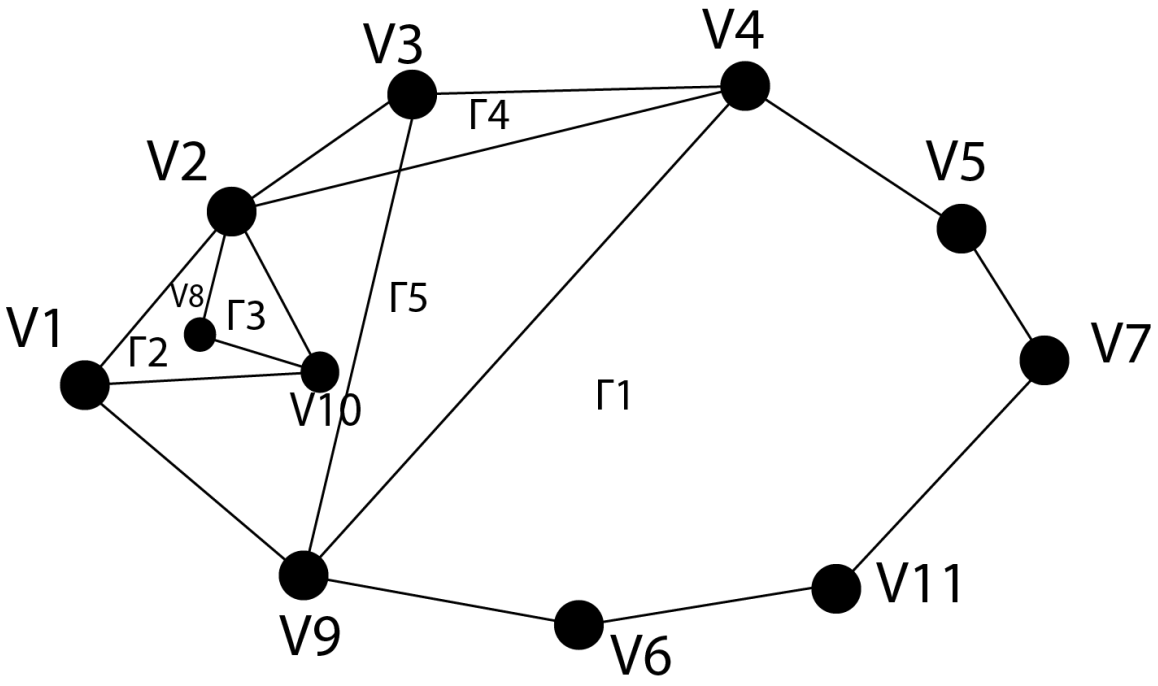
Далі в Г1 вставляє ланцюг S3:



Далі у Г1 вставляєм ланцюг S4:



У Г5 вставляєм ланцюг S5:



Оскільки ребра V3-V9 і V2-V4 перетинаються, цей граф не є планарним.

Завдання №2.

Написати програму, яка реалізує алгоритм Дейкстри знаходження найкоротшого шляху між парою вершин у графі. Протестувати розроблену програму на графі згідно свого варіанту.

Код програми:

```
#include <stdio.h>
#include <stdlib.h>
#define SIZE 6
int main()
{
 int a[SIZE][SIZE], d[SIZE], v[SIZE];
 int temp, minindex, min;
 for (int i = 0; i < SIZE; i++)
```

```
{
 a[i][i] = 0;
 for (int j = i + 1; j < SIZE; j++)
 {
  printf("Enter the weight for %d - %d: ", i + 1, j + 1);
  scanf("%d", &temp);
  a[i][j] = temp;
  a[j][i] = temp;
 }
}
for (int i = 0; i < SIZE; i++)
{
 for (int j = 0; j < SIZE; j++)
 {
  printf(" %5d ", a[i][j]);
 }
 printf("\n");
}
for (int i = 0; i < SIZE; i++)
{
 d[i] = 10000;
 v[i] = 1;
}
d[0] = 0;
do
{
 minindex = 10000;
 min = 10000;
 for (int i = 0; i < SIZE; i++)
 {
  if ((v[i] == 1) && (d[i] < min))
  {
   min = d[i];
   minindex = i;
  }
 }
 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);
printf("\nThe smallest distance for the vertexes: \n");
for (int i = 0; i < SIZE; i++)
{
 printf("%d ", d[i]);
}
int ver[SIZE];
int end = 4;
ver[0] = end + 1;
int k = 1;
int weight = d[end];

while (end > 0)
{
 for (int i = 0; i < SIZE; i++)
 {
  if (a[end][i] != 0)
  {
          temp = weight - a[end][i];
   if (temp == d[i])
   {
    weight = temp;
    end = i;
    ver[k] = i + 1;
    k++;
   }
  }
 }
}
printf("\nThe shortest route:\n");
for (int i = k - 1; i >= 0; i--)
{
 printf("%d ", ver[i]);
```

```
 }
 printf("\n");
 return 0;
}
```