Лабораторна робота № 5.

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

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

ТЕОРЕТИЧНІ ВІДОМОСТІ ТА ПРИКЛАДИ РОЗВ'ЯЗАННЯ ЗАДАЧ

Задача знаходження найкоротшого шляху з одним джерелом полягає у знаходженні найкоротших(мається на увазі найоптимальніших за вагою) шляхів від деякої вершини(джерела) до всіх вершин графа G. Для розв'язку цієї задачі використовується «жадібний» алгоритм, який називається алгоритмом Дейкстри.

«Жадібними» називаються алгоритми, які на кожному кроці вибирають оптимальний із можливих варіантів.

Задача про найкоротший ланцюг. Алгоритм Дейкстри.

Дано n-вершинний граф $G=(V,E)$, у якому виділено пару вершин $v_0, v^* \in V$, і кожне ребро зважене числом $w(e) \geq 0$. Нехай $X=\{x\}$ – множина усіх простих ланцюгів, що з'єднують v_0 з v^*, $x=(V_x, E_x)$. Цільова функція $F(x)=\sum_{e\in E_x} w(e) \to \min$. Потрібно знайти найкоротший ланцюг, тобто $x_0 \in X:\ F(x_0)=\min_{x\in X} F(x)$

Перед описом алгоритму Дейкстри подамо визначення термінів “k-а найближча вершина і “дерево найближчих вершин”. Перше з цих понять визначається індуктивно так.

1-й крок індукції. Нехай зафіксовано вершину x_0, E_1 – множина усіх ребер $e\in E$, інцидентних v_0. Серед ребер $e\in E_1$ вибираємо ребро $e(1)=(v_0, v_1)$, що має мінімальну вагу, тобто $w(e(1))=\min_{e\in E_1} w(e)$. Тоді v_1 називаємо першою найближчою вершиною (НВ), число $w(e(1))$ позначаємо $l(1)=l(v_1)$ і називаємо відстанню до цієї НВ. Позначимо $V_1=\{v_0, v_1\}$ – множину найближчих вершин.

2-й крок індукції. Позначимо E_2 – множину усіх ребер $e=(v', v'')$, $e \in E$, таких що $v' \in V_1$, $v'' \in (V \setminus V_1)$. Найближчим вершинам $v \in V_1$ приписано відстані $l(v)$ до кореня v_0, причому $l(v_0)=0$. Введемо позначення: $\overline{V_1}$ – множина таких вершин $v'' \in (V \setminus V_1)$, що $\exists$ ребра виду $e=(v, v'')$, де $v \in V_1$. Для всіх ребер $e \in E_2$ знаходимо таке ребро $e_2=(v', v_2)$, що величина $l(v')+w(e_2)$ найменша. Тоді v_2 називається другою найближчою вершиною, а ребра e_1, e_2 утворюють зростаюче дерево для виділених найближчих вершин $D_2=\{e_1, e_2\}$.

(s+1)-й крок індукції. Нехай у результаті s кроків виділено множину найближчих вершин $Vs=\{v_0, v_1, \ldots, v_s\}$ і відповідне їй зростаюче дерево $D_s=\{e_1, e_2, \ldots, e_s\}$... Для кожної вершини $v \in V_s$ обчислена відстань $l(v)$ від кореня v_0 до v; $\overline{V_s}$ – множина вершин $v \in (V \setminus V_s)$, для яких існують ребра вигляду $e=(v_r, v)$, де $v_r \in V_s$, $v \in (V \setminus V_s)$. На кроці s+1 для кожної вершини $v_r \in V_s$ обчислюємо відстань до вершини v_r: $L(s+1)(v_r)=l(v_r)+\min\limits_{v^* \in \underline{V_s}} w(v_r, v^*)$, де min береться по всіх ребрах $e=(v_r, v*)$, $v^* \in \overline{V}_s$, після чого знаходимо min серед величин $L(s+1)(v_r)$. Нехай цей min досягнуто для вершин v_{r0} і відповідної їй $v^* \in \overline{V}_s$, що назвемо v_{s+1}. Тоді вершину v_{s+1} називаємо (s+1)-ю НВ, одержуємо множину $V_{s+1}=V_s \bigcup v_{s+1}$ і зростаюче дерево $D_{s+1}=D_s \bigcup (v_{r0}, v_{s+1})$. (s+1)-й крок завершується перевіркою: чи є чергова НВ v_{s+1} відзначеною вершиною, що повинна бути за умовою задачі зв'язано найкоротшим ланцюгом з вершиною v_0. Якщо так, то довжина шуканого ланцюга дорівнює $l(v_{s+1})=l(v_{r0})+w(v_{r0}, v_{s+1})$; при цьому шуканий ланцюг однозначно відновлюється з ребер зростаючого дерева D_{s+1}. У противному випадку випливає перехід до кроку s+2.

Приклад. У графі на рис.5.1 знайти найкоротший ланцюг для виділеної пари вершин v_0 , $v*$.

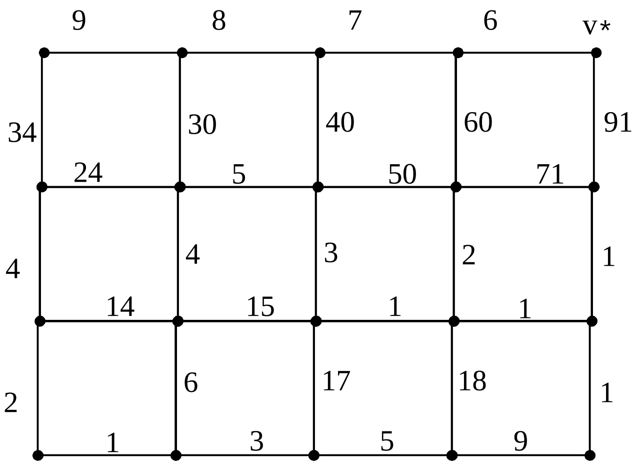


Рисунок 5.1

Розв'язання.

Будемо позначати найближчі вершини v_1, v_2, v_3, … у порядку їхньої появи (див. рис. 5.2): $l(v_1)=1$, $l(v_2)=2$, $l(v_3)=4$, $l(v_4)=6$, $l(v_5)=7$,

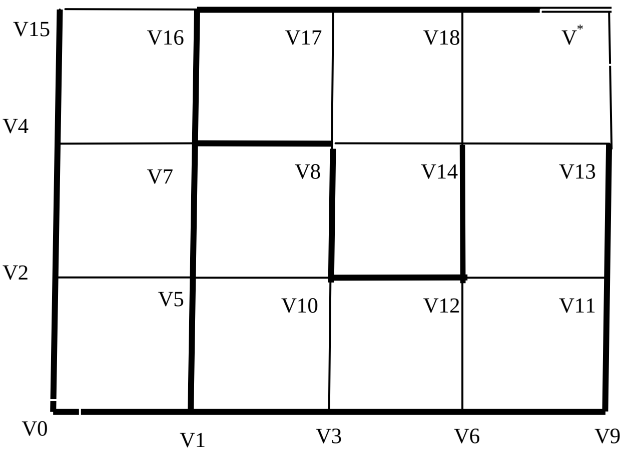


Рисунок 5.2

$l(v_6)=9$, $l(v_7)=11$, $l(v_8)=16$, $l(v_9)=18$, $l(v_{10})=19$, $l(v_{11})=19$, $l(v_{12})=20$, $l(v_{13})=20$, $l(v_{14})=22$, $l(v_{15})=40$, $l(v_{16})=41$, $l(v_{17})=49$, $l(v_{18})=56$, $l(v^*)=62$.

Дерево найближчих вершин виділено на рисунку 5.2 жирними лініями і є кістяковим деревом, тому що містить усі вершини графа.

Шуканий найкоротший ланцюг: $[v_0, v_1, v_5, v_7, v_{16}, v_{17}, v_{18}, v^*]$, довжина ланцюга $l = l(v^*)=62$.

Плоскі і планарні графи

Плоским графом називається граф, вершини якого є точками площини, а ребра – безперервними лініями без самоперетинань, що з'єднують відповідні вершини так, що ніякі два ребра не мають спільних точок крім інцидентної їм обом вершини. Граф називається *планарним*, якщо він є ізоморфним плоскому графу.

Гранню плоского графа називається максимальна по включенню множина точок площини, кожна пара яких може бути з'єднана жордановою кривою, що не перетинає ребра графа. Границею грані будемо вважати множину вершин і ребер, що належать цій грані.

Алгоритм $\underline{\gamma}$ укладання графа $\underline{G}$ являє собою процес послідовного приєднання до деякого укладеного підграфа $\tilde{G}$ графа G нового ланцюга, обидва кінці якого належать $\tilde{G}$. При цьому в якості початкового плоского графа $\tilde{G}$ вибирається будь-який простий цикл графа G. Процес продовжується доти, поки не буде побудовано плоский граф, ізоморфний графові G, або приєднання деякого ланцюга виявиться неможливим. В останньому випадку граф G не є планарным.

Нехай побудоване деяке укладання підграфа $\tilde{G}$ графа G.

Сегментом S відносно $\tilde{G}$ будемо називати підграф графа G одного з наступних виглядів:

- ребро $e \in E$, $e = (u, v)$, таке, що $e \notin \tilde{E}$, $u, v \in \tilde{V}$, $\tilde{G} = (\tilde{V}, \tilde{E})$;
- зв'язний компонент графа $G - \tilde{G}$, доповнений всіма ребрами графа G, інцидентними вершинам узятого компонента, і кінцями цих ребер.

Вершину v сегмента S відносно $\tilde{G}$ будемо називати *контактною*, якщо $v \in \tilde{V}$.

Припустимою гранню для сегмента S відносно $\tilde{G}$ називається грань Г графа $\tilde{G}$, що містить усі контактні вершини сегмента S. Через Г(S) будемо позначати множину припустимих граней для S.

Назвемо α-*ланцюгом* простий ланцюг L сегмента S, що містить дві різні контактні вершини і не містить інших контактних вершин.

Тепер формально опишемо алгоритм γ.

0. Виберемо деякий простий цикл C графа G і укладемо його на площині; покладемо $\tilde{G} = G$.

1. Знайдемо грані графа $\tilde{G}$ і сегменти відносно $\tilde{G}$. Якщо множина сегментів порожня, то перейдемо до пункту 7.

2. Для кожного сегмента S визначимо множину Γ(S).

3. Якщо існує сегмент S, для якого Γ(S)=∅, то граф G не планарний. Кінець. Інакше перейдемо до п. 4.

4. Якщо існує сегмент S, для якого мається єдина припустима грань Γ, то перейдемо до п. 6. Інакше до п. 5.

5. Для деякого сегмента S Γ(S)>1. У цьому випадку вибираємо довільну припустиму грань Γ.

6. Розмістимо довільний α- ланцюг L∈S у грань Γ; замінимо $\tilde{G}$ на $\tilde{G} \bigcup L$ і перейдемо до п. 1.

7. Побудовано укладання $\tilde{G}$ графа G на площині. Кінець.

Кроком алгоритму γ будемо вважати приєднання до $\tilde{G}$ α- ланцюга L.

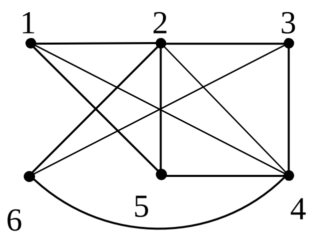


Рисунок 5.3

<u>Приклад.</u> Нехай граф G зображено на рисунку 5.3. Укладемо спочатку цикл C=[1, 2, 3, 4, 1], що розбиває площину на дві грані Γ_1 і Γ_2 .

На рисунку 5.4 зображено граф $\tilde{G} = C$ і сегменти S_1, S_2, S_3 відносно $\tilde{G}$ з контактними вершинами, що обведені колами. Так як $\Gamma(S_i) = \{\Gamma_1, \Gamma_2\}$ (i=1, 2, 3), то кожний α- ланцюг довільного сегмента можна укладати в будь-яку припустиму для нього грань.

Помістимо, наприклад, α- ланцюг L=[2, 5, 4] у Γ_1. Виникає новий граф $\tilde{G}$ і його сегменти (рис. 5.5). При цьому $\Gamma(S_1)= \{\Gamma_3\}$, $\Gamma(S_2)=\{\Gamma_1, \Gamma_2\}$, $\Gamma(S_3)=\{\Gamma_1, \Gamma_2, \Gamma_3\}$.

Укладаємо L=(1, 5) у грань Γ_3 (рис. 5.6).

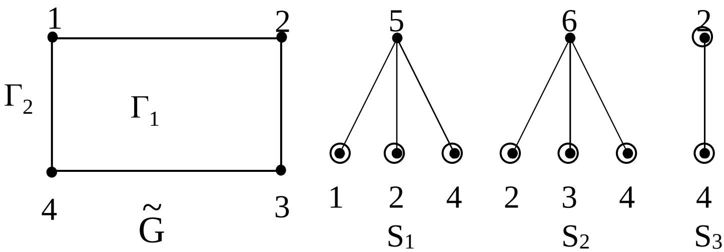


Рисунок 5.4

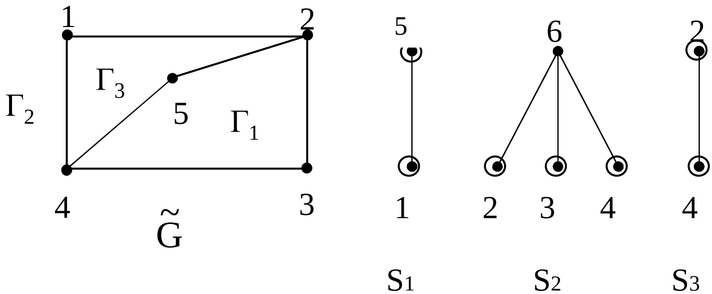


Рисунок 5.5

Тоді $\Gamma(S_1)=\{\Gamma_1, \Gamma_2\}$, $\Gamma(S_2)= \{\Gamma_1, \Gamma_2\}$. Далі укладемо α- ланцюг L=(2, 6, 4) сегмента S_1 у Γ_1 (рис. 5.7). У результаті маємо $\Gamma(S_1)=\{\Gamma_5\}$, $\Gamma(S_2)=\{\Gamma_1, \Gamma_2, \Gamma_5\}$. Нарешті, уклавши ребро (6, 3) у Γ_5, а ребро (2, 4) – наприклад, у Γ_1, одержуємо укладання графа G на площині (рис. 5.8).

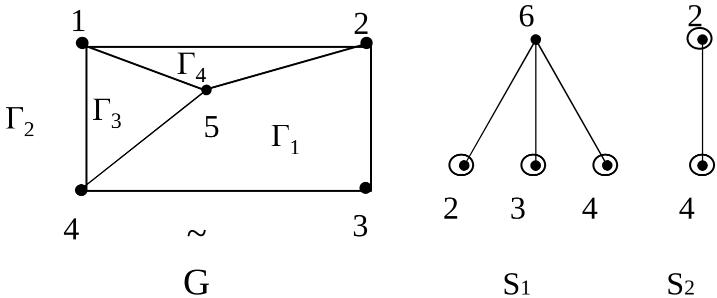


Рисунок 5.6

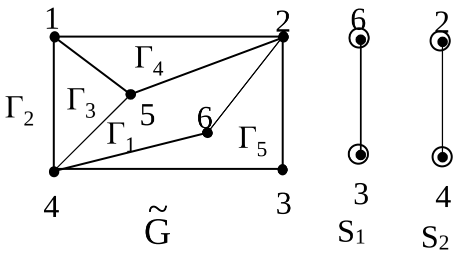


Рисунок 5.7

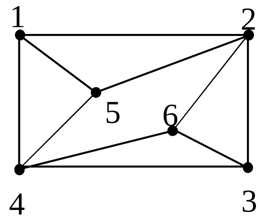


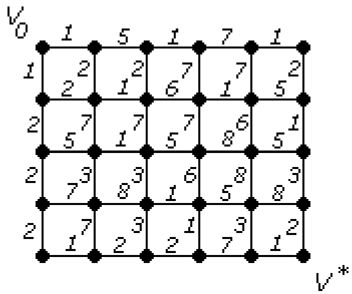
Рисунок 5.8

ІНДИВІДУАЛЬНІ ЗАВДАННЯ

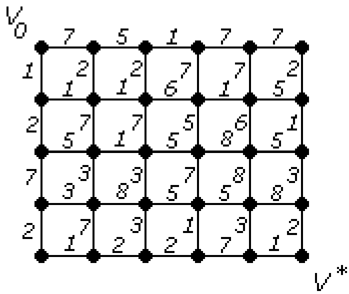
Завдання № 1. Розв'язати на графах наступні 2 задачі:

1. За допомогою алгоритму Дейкстра знайти найкоротший шлях у графі поміж парою вершин V_0 і V^*.

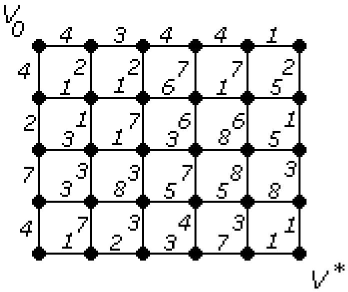
1



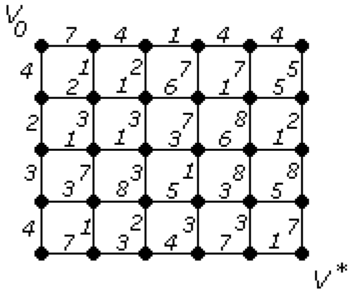
2



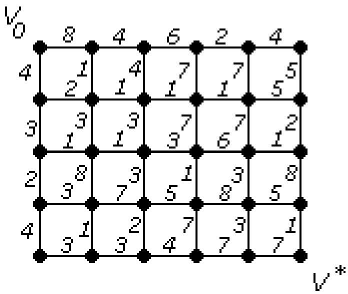
3


V_0
4 3 4 4 1
4 1 2 1 2 6 7 1 7 5 2
2 3 1 1 7 3 6 8 6 5 1
7 3 3 8 3 5 7 5 8 8 3
4 1 7 2 3 3 4 7 3 1 1
V^*

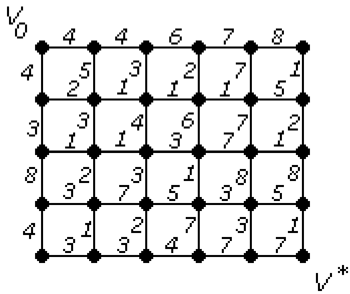
4


V_0
7 4 1 4 4
4 2 1 1 2 6 7 1 7 5 5
2 1 3 1 3 3 7 6 8 1 2
3 3 7 8 3 5 1 3 8 5 8
4 7 1 3 2 4 3 7 3 1 7
V^*

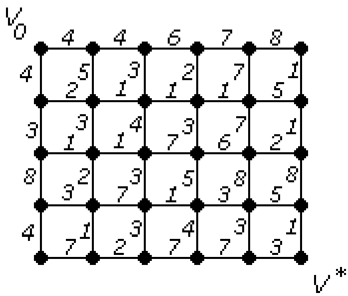
5


V_0
8 4 6 2 4
4 2 1 1 4 1 7 1 7 5 5
3 1 3 1 3 3 7 6 7 1 2
2 3 8 7 3 5 1 8 3 5 8
4 3 1 3 2 4 7 7 3 7 1
V^*

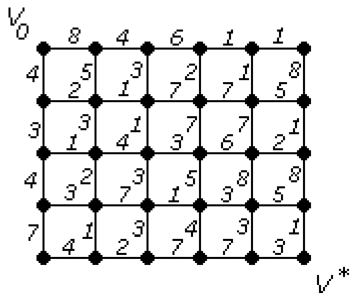
6


V_0
4 4 6 7 8
4 2 5 1 3 1 2 1 7 5 1
3 1 3 1 4 3 6 7 7 1 2
8 3 2 7 3 5 1 3 8 5 8
4 3 1 3 2 4 7 7 3 7 1
V^*

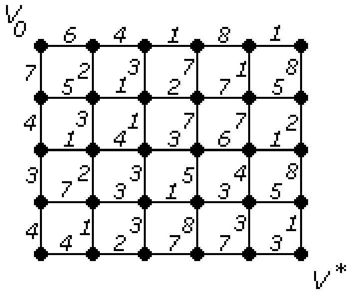
7


V_0
4 4 6 7 8
4 2 5 1 3 1 2 1 7 5 1
3 1 3 1 4 7 3 6 7 2 1
8 3 2 7 3 1 5 3 8 5 8
4 7 1 2 3 7 4 7 3 3 1
V^*

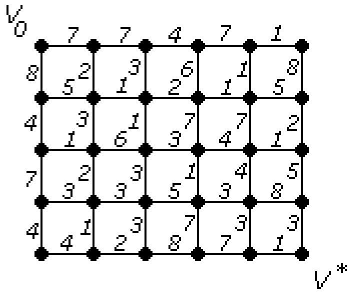
8


V_0
8 4 6 1 1
4 2 5 1 3 7 2 7 1 5 8
3 1 3 4 1 3 7 6 7 2 1
4 3 2 7 3 1 5 3 8 5 8
7 4 1 2 3 7 4 7 3 3 1
V^*

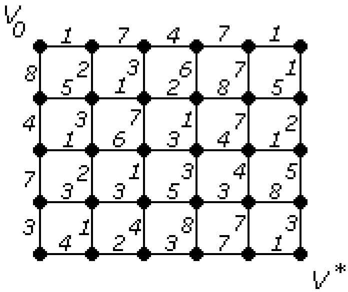
9



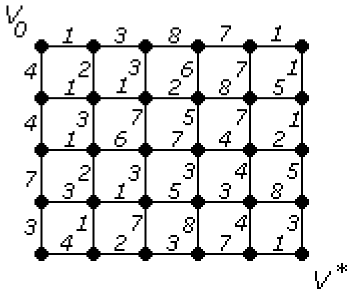
10



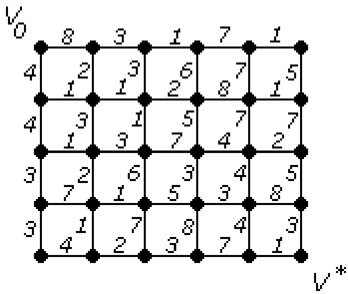
11



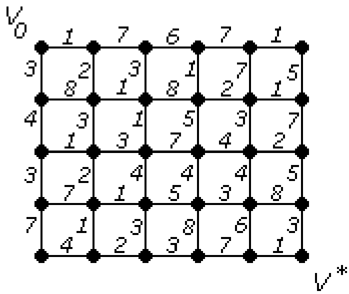
12



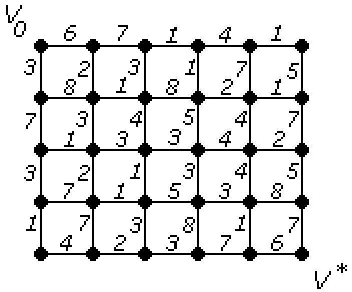
13



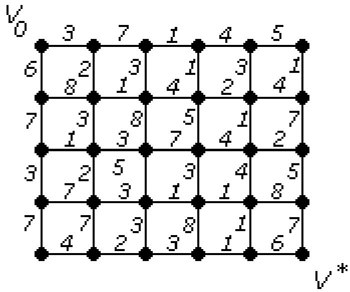
14



15

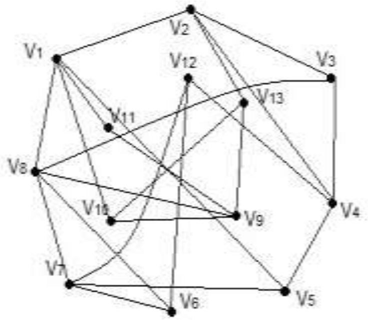


16

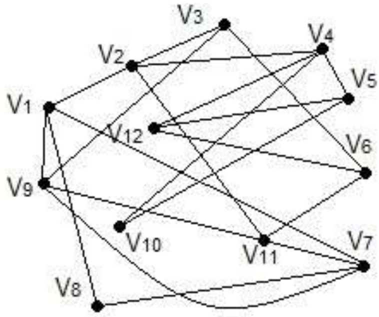


2. За допомогою γ -алгоритма зробити укладку графа у площині, або довести що вона неможлива.

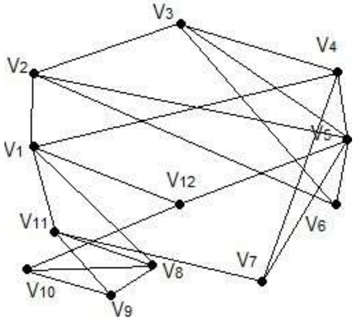
1



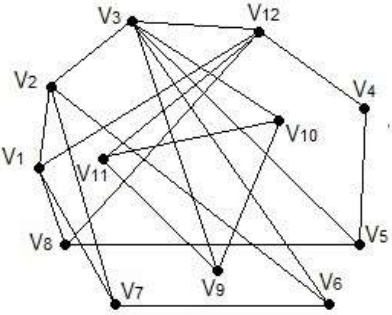
2



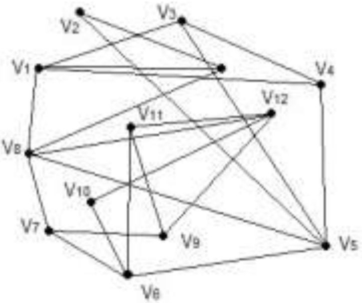
3


V3
V4
V2
V5
V1
V12
V11
V6
V7
V8
V10
V9

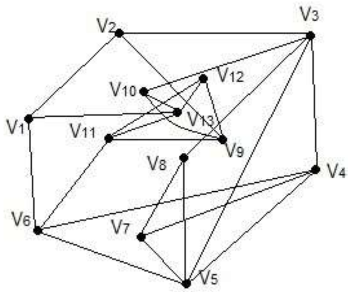
4


V3
V12
V2
V4
V10
V1
V11
V8
V5
V7
V9
V6

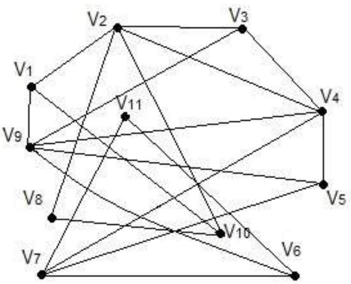
5


V2
V3
V1
V4
V12
V11
V8
V10
V7
V9
V5
V6

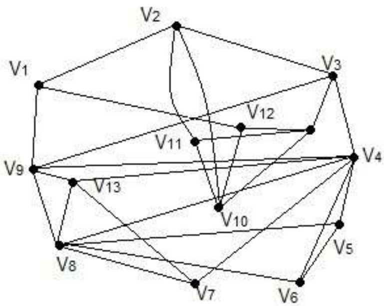
6


V2
V3
V12
V10
V1
V13
V11
V9
V8
V4
V6
V7
V5

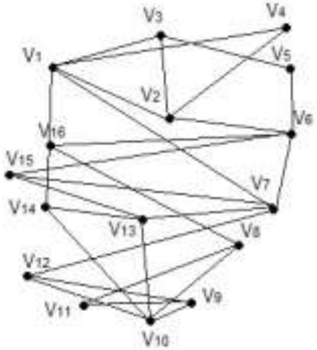
7


V2
V3
V1
V4
V11
V9
V8
V5
V10
V7
V6

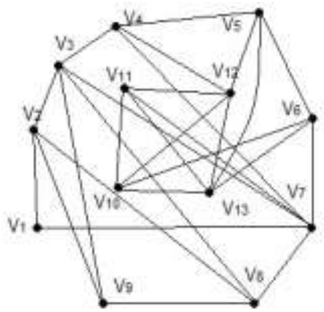
8


V2
V1
V3
V12
V11
V4
V9
V13
V10
V8
V5
V7
V6

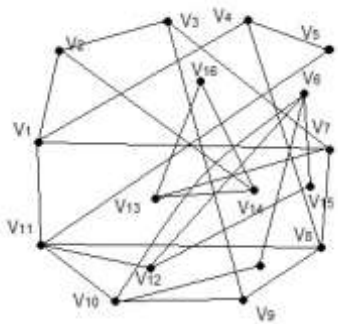
9


V1
V2
V3
V4
V5
V6
V7
V8
V9
V10
V11
V12
V13
V14
V15
V16

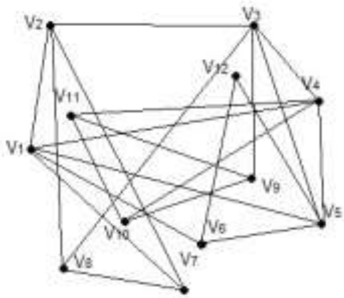
10


V1
V2
V3
V4
V5
V6
V7
V8
V9
V10
V11
V12
V13

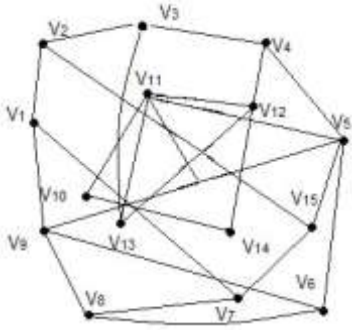
11

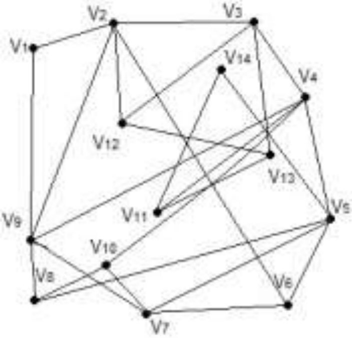

V1
V2
V3
V4
V5
V6
V7
V8
V9
V10
V11
V12
V13
V14
V15
V16

12

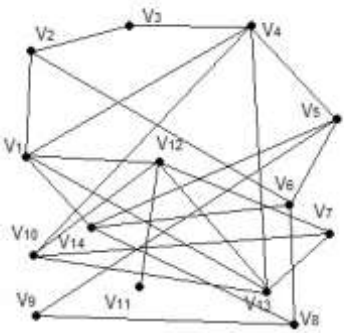

V1
V2
V3
V4
V5
V6
V7
V8
V9
V10
V11
V12

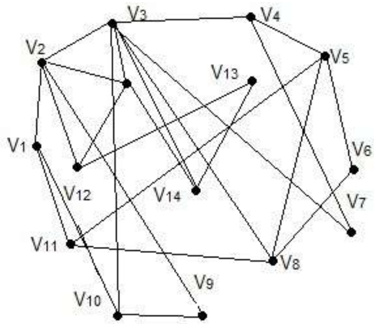
13 **14**





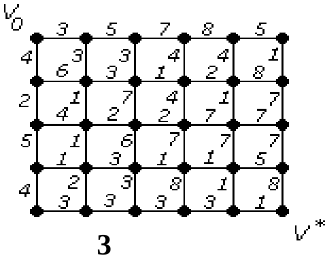
15 **16**

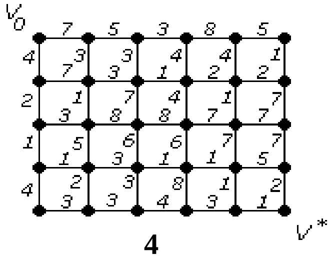




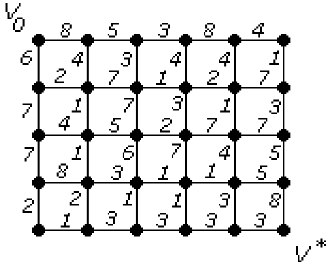
Завдання №2. Написати програму, яка реалізує алгоритм Дейкстри знаходження найкоротшого шляху між парою вершин у графі. Протестувати розроблену програму на графі згідно свого варіанту.

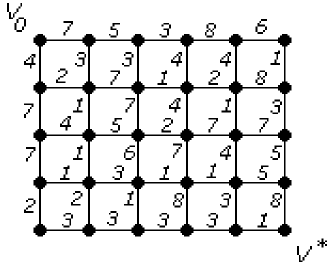
1 **2**



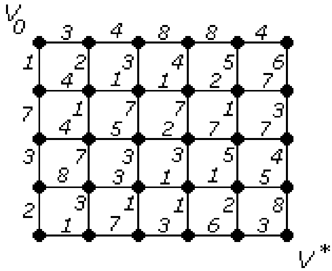


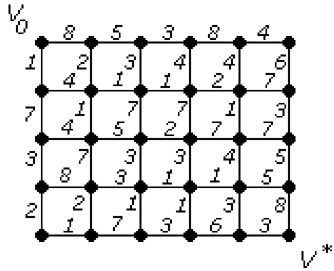
3 **4**



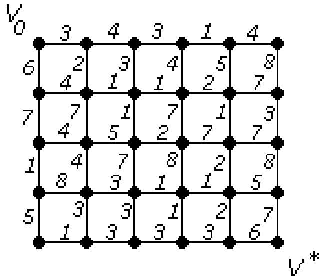


5 **6**

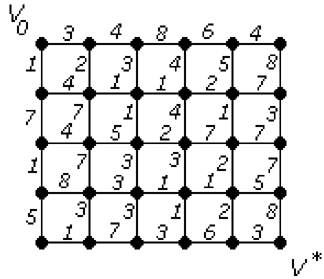




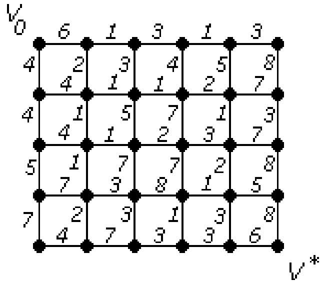
7 **8**



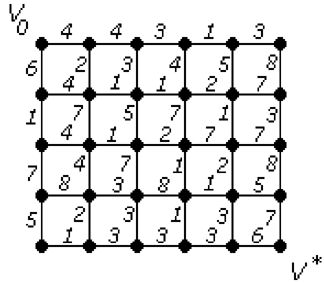
9



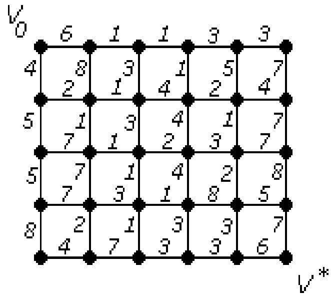
10



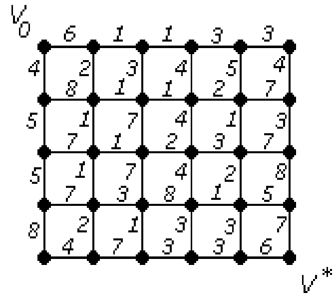
11



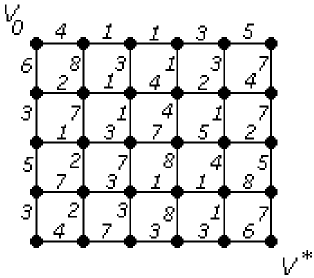
12

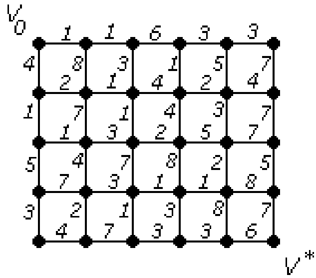


13



14





15

16