Лабораторна робота №6

Алгоритми на графах. Пошук найкоротшого шляху

Метою виконання лабораторної роботи є набуття практичних навичок із проектування, реалізації, тестування та аналізу алгоритмів Дейкстри (Dijkstra) та Флойда (Floyd) пошуку найкоротших шляхів між парами вершин (all-pairs shortest-paths problem)

Зміст:

1. Задача пошуку найкоротшого шляху.

Розглянемо задачу пошуку найкоротшого шляху з однієї вершини: *для даної вершини зваженого зв'язного графа, що називається вихідною (source), треба знайти найкоротший шлях до вихідної вершини (stock).* Важливо підкреслити, що тут нас *не цікавить єдиний найкоротший шлях, що починається у вихідній вершині і проходить через всю решту вершин*. Це значно складніша задача (що являє собою версію задачі комівояжера). Завдання пошуку найкоротших шляхів з однієї вершини вимагає знайти сімейство шляхів, кожен з яких веде від *вхідної* вершини до будь-якої іншої вершини графа, хоча деякі шляхи, звичайно ж, можуть мати спільні ребра.

Задача про найкоротший шлях - одна з найпоширеніших в алгоритмах. Наприклад, побудова дорожнього маршруту – найбільш очевидна з усіх задач про найкоротший шлях, коли вам потрібно якнайшвидше дістатися з пункту відправлення до пункту призначення. Найкоротший шлях може позначатися як загальна відстань, яку вам потрібно покрити, або ж, як мінімально потрібний час, за який ви потрапити до вашого пункту призначення.

2. Алгоритм Дейкстри

Алгоритм Дейкстри використовується для пошуку найкоротших шляхів у графі з не від'ємними вагами ребер від заданої стартової вершини s до всіх інших вершин графу G.

Вхідні дані: $G=(V,E)$ – граф, де V – множина вершин, а E – множина ребер; s – стартова вершина, з якої розпочинається пошук найкоротших шляхів.

Вихідні дані: *pred* – масив, що зберігає позначку вершини попередника для кожної вершини в найкоротшому шляху; dist – масив, що зберігає довжини найкоротших шляхів від стартової вершини s до кожної вершини графу.

Тоді псевдокод алгоритму Дейкстри показаний у вигляді Листингу 6.1

Лістинг 6.1 – Псевдокод алгоритму Дейкстри

```
Dijkstra(G, s) → (pred, dist)
//Input: G = (V,E) - graph,  s - starting node
//Output: pred - array of size |V | such that pred[i] is the
predecessor of node i in the shortest path from s, dist -  array
of size |V | such that dist[i] is the length of the
shortest path calculated from node s to i
1 pred ← CreateArray(|V |)
2 dist ← CreateArray(|V |)
3 pq ← CreatePQ()
4 foreach v in V do //ініціалізуємо масиви pred[] та dist[]
5    pred[v]←-1
6    if v ≠ s then
7         dist[v]←∞
8    else
9         dist[v] ← 0
10   InsertInPQ(pq, v, dist[v])
11 while SizePQ(pq) ≠ 0 do
12   u ← ExtractMinFromPQ(pq)
13   foreach v in Adjacency List(G, u) do   // перебор суміжних
вершин
14        if dist[v] > dist[u] + Weight(G, u, v) then
15             dist[v] ← dist[u] + Weight(G, u, v)
16             pred[v] ← u
17             UpdatePQ(pq, v, dist[v])
18 return (pred, dist)
```

Наведемо наступні пояснення.

1. Ініціалізація (Рядки 1–10)

В рядках 1-3 створюються масиви *pred* та *dist* і черга з пріоритетом *pq*.

В рядках 4-10 для кожної вершини *v* із множини вершин графа *V* виконується наступна ініціалізація:

– Масив pred[v] ініціалізується значеннями -1, що вказує на відсутність попередників на початку

– Якщо вершина v не є стартовою, то dist[v] встановлюється в нескінченність (∞), що означає, що спочатку відстань до неї невідома.

– Для стартової вершини s dist[s] встановлюється в 0, оскільки відстань від вершини до самої себе дорівнює нулю.

– Всі вершини додаються в чергу pq, де вони зберігаються разом із поточною відстанню dist[v].

2. Основний цикл (Рядки 11–17)

В рядку 11 виконується цикл while доки черга *pq* не порожня. У черзі з пріоритетом завжди видаляється елемент із найменшим значенням *dist*[*u*], що означає, що спершу обробляються найближчі до s вершини.

В рядку 12 видаляємо вершину *u* з черги *pq*. Це вершина, для якої знайдено мінімальну відстань від *s*.

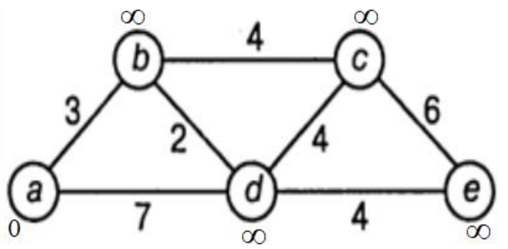
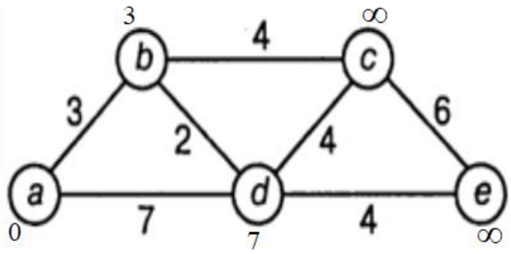
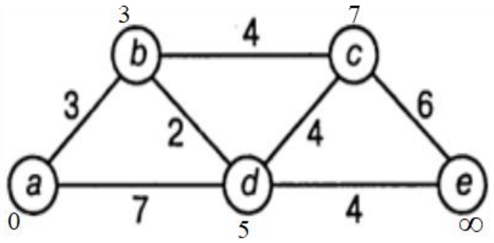
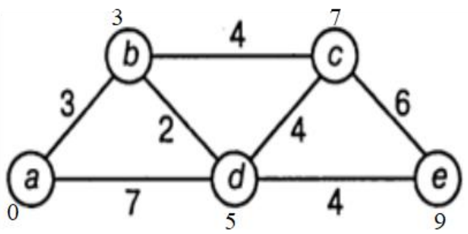
В рядках 13-17 виконується перевірка кожної суміжної вершини *v* для *u*.

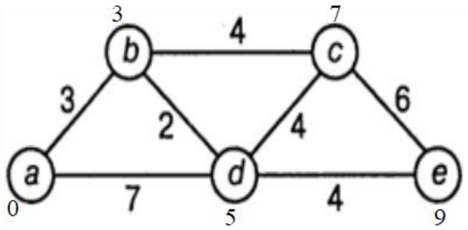
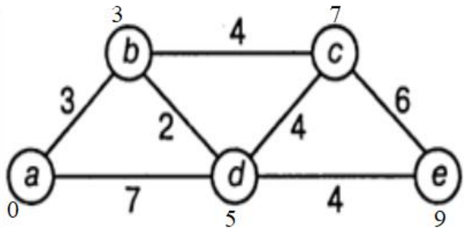
В рядку 14 перевіряється умова для виконання релаксації – якщо поточна відстань до *v* (*dist*[*v*]) більше, ніж відстань до *u* плюс вага ребра між *u* та *v*, тоді відстань до *v* може бути скорочена.
В рядках 15-17 в разі необхідності оновлюємо *dist*[*v*], встановлюючи його як *dist*[*u*] + *Weight*(*G*, *u*, *v*) та змінюємо значення попередника *pred*[*v*] в масиві на *u* та оновлюємо відстань *dist*[*v*] у черзі *pq* для вершини v, щоб зберігати правильний пріоритет під час наступних ітерацій.

3. *Повернення результату* (Рядок 18)

У таблиці 6.1 показаний хід побудови мінімального шляху від вершини *a* до всіх інших вершин графа *G* за алгоритмом Дейкстри

Таблиця 6.1 – Побудова мінімального шляху від *a* до всіх інших вершин графа G за алгоритмом Дейкстри

  	**Ініціалізація** *dist*[a]=0, *dist*[b]=∞, *dist*[c]=∞, *dist*[d]=∞, *dist*[e]=∞, *pred*[a]=-1, *pred*[b]=-1, *pred*[c]=-1, *pred*[d]=-1, *pred*[e]=-1 pq={(a,0),(b, ∞),(c, ∞),(d, ∞),(e, ∞)} **1.**u=a ExtractMinFromPQ(pq) →a Суміжні вершини Adj(a):b, d. Релаксація ребер v=b: dist[b] > dist[a] + Weight(a, b) → ∞ > 0 + 3=3. dist [b]← 3; pred[b] ←a; UpdatePQ(pq, b, 3). v=d: dist[d] > dist[a] + Weight(a, d) → ∞ > 0 + 7. dist [d]← 7; pred[d] ←a; UpdatePQ(pq, d, 7) **Поточний стан** *dist*[a]=0, *dist*[b]=*3*, *dist*[c]=∞, *dist*[d]=*7*, *dist*[e]=∞, *pred*[a]=-1, *pred*[b]=*a*, *pred*[c]=-1, *pred*[d]=*a*, *pred*[e]=-1 pq={(b, *3*), (d, *7*), (c, ∞),(e, ∞)}
 	**2.** u=b ExtractMinFromPQ(pq) →b з dist=3 Суміжні вершини Adj(b): a (оброблена), c, d. Релаксація ребер v=c: dist[c] > dist[b] + Weight(b, c) → ∞ > 3 + 4=**7**. dist [c]← **7**; pred[c] ←b; UpdatePQ(pq, c, 7). v=d: dist[d] > dist[b] + Weight(b, d) → 7 > 3 + 2=**5**. dist [d]← **5**; pred[d] ←b; UpdatePQ(pq, d, 5) **Поточний стан** *dist*[a]=0, *dist*[b]=*3*, *dist*[c]=*7*, *dist*[d]=*5*, *dist*[e]=∞, *pred*[a]=-1, *pred*[b]=*a*, *pred*[c]=*b*, *pred*[d]=*b*, *pred*[e]=-1 pq={(d, *5*), (c, *7*), (e, ∞)}
 	**3.**u=d ExtractMinFromPQ(pq) →d з dist=5 Суміжні вершини Adj(d): a, b (оброблені), c, e Релаксація ребер v=c: dist[c] > dist[d] + Weight(d, c) → **7** > 5 + 4=9. v=e: dist[e] > dist[d] + Weight(d, e) → ∞ > 5 + 4=9. dist [e]← 9; pred[e] ←d; UpdatePQ(pq, e, 9) **Поточний стан** *dist*[a]=0, *dist*[b]=*3*, *dist*[c]=*7*, *dist*[d]=*5*, *dist*[e]=*9*, *pred*[a]=-1, *pred*[b]=*a*, *pred*[c]=*b*, *pred*[d]=*b*, *pred*[e]=*d* pq={(c, *7*), (e, *9*)}

	4. u=c ExtractMinFromPQ(pq) →c з dist=7 Суміжні вершини Adj(c): a, d (оброблені), e Релаксація ребер v=e: dist[e] > dist[c] + Weight(c, e) → 9 > 7 + 6=13. **Поточний стан** *dist*[a]=0, *dist*[b]=*3*, *dist*[c]=*7*, *dist*[d]=*5*, *dist*[e]=*9*, *pred*[a]=-1, *pred*[b]=*a*, *pred*[c]=*b*, *pred*[d]=*b*, *pred*[e]=*d* pq={(e, *9*)}
	5. u=e ExtractMinFromPQ(pq) →e з dist=9 Суміжні вершини Adj(e): c, d (оброблені),оновлень немає pq={}

Операція використання міток та використання черги алгоритму Дейкстри схожі з алгоритмом Прима. В обох випадках будується і розширюється піддерево вершин шляхом вибору чергової вершини з черги з пріоритетами, що містить вершини, що залишаються поза деревом. Однак алгоритми призначені для вирішення різних задач та працюють з пріоритетами, які обчислюються різними способами: *алгоритм Дейкстри* порівнює *довжини шляхів* і повинен *підсумовути ваги ребер*, у той час як *алгоритм Прима* порівнює *ваги ребер* такими як вони є. Наведемо псевдокод алгоритму Дейкстри:

Для порівняння за алгоритмом Прима

for i ← 1 to |V| - 1 **do**

Пошук ребра з мінімальною вагою $e^* = (v^*,u^*)$ серед усіх

ребер (v, u) таких, що $v \in V_T$ та $u \in V - V_T$

$V_T \leftarrow V_T \cup \{u^*\}$

$\boldsymbol{E_T \leftarrow E_T \cup \{e^*\}}$

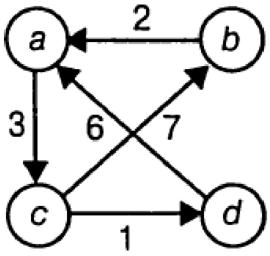
У таблиці 6.2 показані довжини найкоротших відстаней до решти вершин графа з вершини a

Таблиця 6.2 – Довжини найкоротших відстаней до решти вершин графа *G* з вершини *a*

Найкоротші шляхи			
З Вершини	В Вершину	Шлях	Вага
a	b	a→b	3
	c	a→b→c	7
	d	a→b→d	5
	e	a→b→d→e	9

4. Алгоритм Флойда

Як ми вже знаємо задача пошуку найкоротших шляхів між усіма парами вершин (all-pairs shortest-paths problem) полягає у пошуку для даного орієнтованого або неорієнтованого зваженого зв'язного графа відстаней від кожної вершини до всіх інших вершин. Для запису довжин найкоротших шляхів зручно скористатися матрицею D розміром n×n, яка називається матрицею відстаней (*distance matrix*): елемент d_{ij} на перетині i-ого рядка и j-го стовпця такої матриці вказує довжину найкоротшого шляху від i-ої вершини до j-ої вершини ($1 \le i,j \le n$) (рис. 6.1).



$$W = \begin{array}{c} \\ a \\ b \\ c \\ d \end{array} \begin{array}{c} \begin{array}{cccc} a & b & c & d \end{array} \\ \begin{bmatrix} 0 & \infty & 3 & \infty \\ 2 & 0 & \infty & \infty \\ \infty & 7 & 0 & 1 \\ 6 & \infty & \infty & 0 \end{bmatrix} \end{array} \qquad D = \begin{array}{c} \\ a \\ b \\ c \\ d \end{array} \begin{array}{c} \begin{array}{cccc} a & b & c & d \end{array} \\ \begin{bmatrix} 0 & 10 & 3 & 4 \\ 2 & 0 & 5 & 6 \\ 7 & 7 & 0 & 1 \\ 6 & 16 & 9 & 0 \end{bmatrix} \end{array}$$

а б в

Рисунок 6.1 – Орієнтований граф (а), його матриця суміжності (б) і матриця відстаней (в)

Для побудови матриці відстаней розроблено алгоритм Флойда (Floyd's algorithm) (Флойда-Уоршелла). За допомогою алгоритму Флойда обчислюється матриця відстаней зваженого графа з n вершинами за допомогою побудови послідовності матриць:

$$D^{(0)}, D^{(1)}, \dots D^{(k)}, \dots D^{(n)} \qquad (1),$$

де елемент $d_{ij}^{(k)}$ на перетині i-ого рядка та j-го стовпця матриці $D^{(k)}$, k= 0,1,..., n дорівнює довжині найкоротшого шляху серед всіх шляхів від i-ої вершини до j-ої, у яких проміжні вершини не можуть мати номери, що перевищують k. Зокрема, послідовність починається з матриці $D^{(0)}$ у якій у шляхах не може бути проміжних вершин, $D^{(0)}$ являє собою просто вихідну вагову матрицю графу W. Остання матриця послідовності, $D^{(n)}$ містить довжини найкоротших шляхів серед усіх шляхів, в яких як проміжні вершини можуть бути будь-які з n вершин графу, так що матриця $D^{(n)}$ і є матриця відстаней графу, яку ми обчислюємо. При цьому ми обчислюємо всі елементи кожної матриці $D^{(k)}$ на основі інформації про елементи попередньої матриці $D^{(k-1)}$ у послідовності (1). Якщо $d_{ij}^{(k)}$ - елемент на перетині i-ого рядка та j-го стовпця матриці $D^{(k)}$, то $d_{ij}^{(k)}$ - дорівнює довжині найкоротшого шляху серед всіх шляхів від i-ої вершини v_i до j-ої вершини v_j проміжні вершини яких мають номери не вище k. Таким чином, всі такі шляхи можна розбити на дві непересічні підмножини: ті шляхи, в яких як проміжна не бере участі k-а вершина v_k, та ті, у яких вона є однією із проміжних.

Оскільки шляхи в першій підмножині містять проміжні вершини з номерами не вище k - 1, то найкоротший шлях між цими вершинами, за визначенням наших матриць, має довжину $d_{ij}^{(k-1)}$.

У другій підмножині ми можемо обмежити розгляд лише тих вершин, у яких вершина v_k входить лише один раз. Тобто кожний з таких шляхів складається з шляху від v_i до v_k, причому всі проміжні вершини мають номери не вище k - 1, і шляхи від v_k до v_j, у якого всі проміжні вершини також мають номери не вище k - 1.

Оскільки довжина найкоротшого шляху від v_i до v_k серед усіх шляхів, які не використовують проміжних вершин з номерами більше k - 1, дорівнює

$d_{ik}^{(k-1)}$, а довжина найкоротшого шляху від v_k до v_j серед усіх шляхів, які не використовують проміжних вершин з номерами більше k - 1, дорівнює

$d_{kj}^{(k-1)}$, довжина найкоротшого шляху від v_i до v_j через вершину v_k дорівнює $d_{ik}^{(k-1)}+\ d_{kj}^{(k-1)}$. З урахуванням довжини найкоротших шляхів в обох підмножинах отримуємо наступне рекурентне співвідношення:

$$d_{ij}^{(k)}=\min\{\ d_{ij}^{(k-1)},\ d_{ik}^{(k-1)}+\ d_{kj}^{\ (k-1)}\} \text{ для } k\geq 1\ ,\ d_{ij}^{(0)}=\ w_{ij} \qquad (2)$$

Тобто елемент на перетині i-ого рядка та j-го стовпця поточної матриці відстаней $D^{(k-1)}$ замінюється сумою елементів на перетині того ж i-ого рядка та k-го стовпця, та k-ого рядка та того ж j-го стовпця тоді і тільки тоді, коли ця сума виявляється меншою за поточне значення.

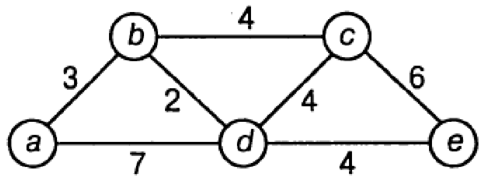
Псевдокод алгоритму Флойда з урахуванням того, що кожна наступна матриця в послідовності (1) може бути записана поверх попередньої має (Лістинг 6.2).

Лістинг 6.2 – Псевдокод алгоритма Флойда

```
Алгоритм Floyd (W [1..n, 1..n])
// Вхідні дані: Вагова матриця W графу
// Вихідні дані: Матриця довжин найкоротших шляхів D
D ← W // Не вимагається, якщо W може бути перезаписана
for k ←1 to n do
     for i ←1 to n do
          for j ←1 to n do
               D[i,j] ←min{D[i,j], D[i,k]+D[k,j]}
return D
```

У таблиці 6.3 показаний хід знаходження найкоротшого шляху у графі за алгоритмом Флойда $D[i,j] \leftarrow \mathbf{min}\{\ D[i,k]+D[k,j],\ D[i,j]\}$

Таблиця 6.3 Знаходження найкоротшого шляху у графі за алгоритмом Флойда

<table>
<tr>
<td>


i=1 j=1-5 (от 1 до 5)
$d_{1,1}^{1}=\min\{d_{1,1}+d_{1,1;}\ d_{1,1}\}=\min\{0+0;\ 0\}=0$
$d_{1,2}^{1}=\min\{d_{1,1}+d_{1,2;}\ d_{1,2}\}=\min\{0+3;\ 3\}=3$
$d_{1,3}^{1}=\min\{d_{1,1}+d_{1,3;}\ d_{1,3}\}=\min\{0+\infty;\ \infty\}=\infty$
$d_{1,4}^{1}=\min\{d_{1,1}+d_{1,4;}\ d_{1,4}\}=\min\{0+7;\ 7\}=7$
$d_{1,5}^{1}=\min\{d_{1,1}+d_{1,5;}\ d_{1,5}\}=\min\{0+\infty;\ \infty\}=\infty$
i=2 j=1-5
$d_{2,1}^{1}=\min\{d_{2,1}+d_{1,1;}\ d_{2,1}\}=\min\{3+0;\ 3\}=3$
$d_{2,2}^{1}=\min\{d_{2,1}+d_{1,2;}\ d_{2,2}\}=\min\{3+3;\ 0\}=0$
$d_{2,3}^{1}=\min\{d_{2,1}+d_{1,3;}\ d_{2,3}\}=\min\{3+\infty;\ 4\}=4$
$d_{2,4}^{1}=\min\{d_{2,1}+d_{1,4;}\ d_{2,4}\}=\min\{3+7;\ 2\}=2$
$d_{2,5}^{1}=\min\{d_{2,1}+d_{1,5;}\ d_{2,5}\}=\min\{3+\infty;\ \infty\}=\infty$
i=3 j=1-5
</td>
<td>
$$\begin{array}{cc|ccccc} & & & & j & & \\ & & 1(a) & 2(b) & 3(c) & 4(d) & 5(e) \\ & 1(a) & 0 & 3 & \infty & 7 & \infty \\ & 2(b) & 3 & 0 & 4 & 2 & \infty \\ i & 3(c) & \infty & 4 & 0 & 4 & 6 \\ & 4(d) & 7 & 2 & 4 & 0 & 4 \\ & 5(e) & \infty & \infty & 6 & 4 & 0 \end{array}$$
k=1 $D^{(1)}$
$\mathbf{d_{i,j}^{1}=\min\{\ d_{i,1}+\ d_{1,j};\ d_{i,j}\}.}$
</td>
</tr>
</table>

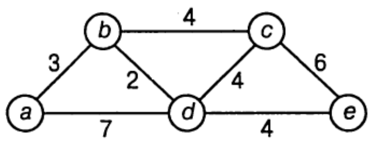
$d_{3,1}^{1}=\min\{d_{3,1}+d_{1,1};\ d_{3,1}\}=\min\{\infty+0;\ \infty\}=\infty$ $d_{3,2}^{1}=\min\{d_{3,1}+d_{1,2};\ d_{3,2}\}=\min\{\infty+3;\ 4\}=4$ $d_{3,3}^{1}=\min\{d_{3,1}+d_{1,3};\ d_{3,3}\}=\min\{\infty+\infty;\ 0\}=0$ $d_{3,4}^{1}=\min\{d_{3,1}+d_{1,4};\ d_{3,4}\}=\min\{\infty+7;\ 4\}=4$ $d_{3,5}^{1}=\min\{d_{3,1}+d_{1,5};\ d_{3,5}\}=\min\{\infty+\infty;\ 6\}=6$ $i=4\ j=1\text{-}5$ $d_{4,1}^{1}=\min\{d_{4,1}+d_{1,1};\ d_{4,1}\}=\min\{7+0;\ 7\}=7$ $d_{4,2}^{1}=\min\{d_{4,1}+d_{1,2};\ d_{4,2}\}=\min\{7+3;\ 2\}=2$ $d_{4,3}^{1}=\min\{d_{4,1}+d_{1,3};\ d_{4,3}\}=\min\{7+\infty;\ 4\}=4$ $d_{4,4}^{1}=\min\{d_{4,1}+d_{1,4};\ d_{4,4}\}=\min\{7+7;\ 0\}=0$ $d_{4,5}^{1}=\min\{d_{4,1}+d_{1,5};\ d_{4,5}\}=\min\{7+\infty;\ 4\}=4$ $i=5\ j=1\text{-}5$ $d_{5,1}^{1}=\min\{d_{5,1}+d_{1,1};\ d_{5,1}\}=\min\{\infty+0;\ \infty\}=\infty$ $d_{5,2}^{1}=\min\{d_{5,1}+d_{1,2};\ d_{5,2}\}=\min\{\infty+3;\ \infty\}=\infty$ $d_{5,3}^{1}=\min\{d_{5,1}+d_{1,3};\ d_{5,3}\}=\min\{\infty+\infty;\ 6\}=6$ $d_{5,4}^{1}=\min\{d_{5,1}+d_{1,4};\ d_{5,4}\}=\min\{\infty+7;\ 4\}=4$ $d_{5,5}^{1}=\min\{d_{5,1}+d_{1,5};\ d_{5,5}\}=\min\{\infty+\infty;\ 0\}=0$	$\begin{array}{cccccc} & 1(a) & 2(b) & 3(c) & 4(d) & 5(e) \\ 1(a) & 0 & 3 & \infty & 7 & \infty \\ 2(b) & 3 & 0 & 4 & 2 & \infty \\ 3(c) & \infty & 4 & 0 & 4 & 6 \\ 4(d) & 7 & 2 & 4 & 0 & 4 \\ 5(e) & \infty & \infty & 6 & 4 & 0 \end{array}$ **k=2 $D^{(2)}$** **$d_{i,j}^{2}=\min\{\ d_{i,2}^{1}+\ d_{2,j}^{1};\ d_{i,j}^{1}\}.$**
$i=1\ j=1\text{-}5$ $d_{1,1}^{2}=\min\{d_{1,2}^{1}+d_{2,1}^{1}\ d_{1,1}^{1}\}=\min\{3+3;\ 0\}=0$ $d_{1,2}^{2}=\min\{d_{1,2}^{1}+d_{2,2}^{1}\ d_{1,2}^{1}\}=\min\{3+0;\ 3\}=3$ **$d_{1,3}^{2}=\min\{d_{1,2}^{1}+d_{2,3}^{1}\ d_{1,3}^{1}\}=\min\{3+4;\ \infty\}=7$** **$d_{1,4}^{2}=\min\{d_{1,2}^{1}+d_{2,4}^{1}\ d_{1,4}^{1}\}=\min\{3+2;\ 7\}=5$** $d_{1,5}^{2}=\min\{d_{1,2}^{1}+d_{2,5}^{1}\ d_{1,5}^{1}\}=\min\{3+\infty;\ \infty\}=\infty$ $i=2\ j=1\text{-}5$ $d_{2,1}^{2}=\min\{d_{2,2}^{1}+d_{2,1}^{1}\ d_{2,1}^{1}\}=\min\{0+3;\ 3\}=3$ $d_{2,2}^{2}=\min\{d_{2,2}^{1}+d_{2,2}^{1}\ d_{2,2}^{1}\}=\min\{0+0;\ 0\}=0$ $d_{2,3}^{2}=\min\{d_{2,2}^{1}+d_{2,3}^{1}\ d_{2,3}^{1}\}=\min\{0+4;\ 4\}=4$ $d_{2,4}^{2}=\min\{d_{2,2}^{1}+d_{2,4}^{1}\ d_{2,4}^{1}\}=\min\{0+2;\ 2\}=2$ $d_{2,5}^{2}=\min\{d_{2,2}^{1}+d_{2,5}^{1}\ d_{2,5}^{1}\}=\min\{0+\infty;\ \infty\}=\infty$ $i=3\ j=1\text{-}5$ **$d_{3,1}^{2}=\min\{d_{3,2}^{1}+d_{2,1}^{1}d_{3,1}^{1}\}=\min\{4+3;\ \infty\}=7$** $d_{3,2}^{2}=\min\{d_{3,2}^{1}+d_{2,2}^{1}d_{3,2}^{1}\}=\min\{4+0;\ 4\}=4$ $d_{3,3}^{2}=\min\{d_{3,2}^{1}+d_{2,3}^{1}d_{3,3}^{1}\}=\min\{4+4;\ 0\}=0$ $d_{3,4}^{2}=\min\{d_{3,2}^{1}+d_{2,4}^{1}d_{3,4}^{1}\}=\min\{4+2;\ 4\}=4$ $d_{3,5}^{2}=\min\{d_{3,2}^{1}+d_{2,5}^{1}d_{3,5}^{1}\}=\min\{4+\infty;\ 6\}=6$ $i=4\ j=1\text{-}5$ **$d_{4,1}^{2}=\min\{d_{4,2}^{1}+d_{2,1}^{1}\ d_{4,1}^{1}\}=\min\{2+3;\ 7\}=5$** $d_{4,2}^{2}=\min\{d_{4,2}^{1}+d_{2,2}^{1}\ d_{4,2}^{1}\}=\min\{2+0;\ 2\}=2$ $d_{4,3}^{2}=\min\{d_{4,2}^{1}+d_{2,3}^{1}\ d_{4,3}^{1}\}=\min\{2+4;\ 4\}=4$ $d_{4,4}^{2}=\min\{d_{4,2}^{1}+d_{2,4}^{1}\ d_{4,4}^{1}\}=\min\{2+2;\ 0\}=0$ $d_{4,5}^{2}=\min\{d_{4,2}^{1}+d_{2,5}^{1}\ d_{4,5}^{1}\}=\min\{2+\infty;\ 4\}=4$ $i=5\ j=1\text{-}5$ $d_{5,1}^{2}=\min\{d_{5,2}^{1}+d_{2,1}^{1}\ d_{5,1}^{1}\}=\min\{\infty+3;\ \infty\}=\infty$ $d_{5,2}^{2}=\min\{d_{5,2}^{1}+d_{2,2}^{1}\ d_{5,2}^{1}\}=\min\{\infty+0;\ \infty\}=\infty$ $d_{5,3}^{2}=\min\{d_{5,2}^{1}+d_{2,3}^{1}\ d_{5,3}^{1}\}=\min\{\infty+4;\ 6\}=6$ $d_{5,4}^{2}=\min\{d_{5,2}^{1}+d_{2,4}^{1}\ d_{5,4}^{1}\}=\min\{\infty+2;\ 4\}=4$ $d_{5,5}^{2}=\min\{d_{5,2}^{1}+d_{2,5}^{1}\ d_{5,5}^{1}\}=\min\{\infty+\infty;\ 0\}=0$	$\begin{array}{cccccc} & 1(a) & 2(b) & 3(c) & 4(d) & 5(e) \\ 1(a) & 0 & 3 & \mathbf{7} & \mathbf{5} & \infty \\ 2(b) & 3 & 0 & 4 & 2 & \infty \\ 3(c) & \mathbf{7} & 4 & 0 & 4 & 6 \\ 4(d) & \mathbf{5} & 2 & 4 & 0 & 4 \\ 5(e) & \infty & \infty & 6 & 4 & 0 \end{array}$ **k=3 $D^{(3)}$** **$d_{i,j}^{3}=\min\{\ d_{i,3}^{2}+\ d_{3,j}^{2};\ d_{i,j}^{2}\}.$**
$i=1\ j=1\text{-}5$ $d_{1,1}^{3}=\min\{d_{1,3}^{2}+d_{3,1}^{2}\ d_{1,1}^{2}\}=\min\{7+7;\ 0\}=0$ $d_{1,2}^{3}=\min\{d_{1,3}^{2}+d_{3,2}^{2}\ d_{1,2}^{2}\}=\min\{7+4;\ 3\}=3$ $d_{1,3}^{3}=\min\{d_{1,3}^{2}+d_{3,3}^{2}\ d_{1,3}^{2}\}=\min\{7+0;\ 7\}=7$ $d_{1,4}^{3}=\min\{d_{1,3}^{2}+d_{3,4}^{2}\ d_{1,4}^{2}\}=\min\{7+4;\ 5\}=5$	

<table>
<tr>
<td>
$\mathbf{d_{1,5}^3=min\{d_{1,3}^2+d_{3,5}^2\ d_{1,5}^2\}=min\{7+6;\ \infty\}=13}$
i=2 j=1-5
$d_{2,1}^3=\min\{d_{2,3}^2+d_{3,1}^2\ d_{2,1}^2\}=\min\{4+7;\ 3\}=3$
$d_{2,2}^3=\min\{d_{2,3}^2+d_{3,2}^2\ d_{2,2}^2\}=\min\{4+4;\ 0\}=0$
$d_{2,3}^3=\min\{d_{2,3}^2+d_{3,3}^2\ d_{2,3}^2\}=\min\{4+0;\ 4\}=4$
$d_{2,4}^3=\min\{d_{2,3}^2+d_{3,4}^2\ d_{2,4}^2\}=\min\{4+4;\ 2\}=2$
$\mathbf{d_{2,5}^3=min\{d_{2,3}^2+d_{3,5}^2\ d_{2,5}^2\}=min\{4+6;\ \infty\}=10}$
i=3 j=1-5
$d_{3,1}^3=\min\{d_{3,3}^2+d_{3,1}^2\ d_{3,1}^2\}=\min\{0+7;\ 7\}=7$
$d_{3,2}^3=\min\{d_{3,3}^2+d_{3,2}^2\ d_{3,2}^2\}=\min\{0+4;\ 4\}=4$
$d_{3,3}^3=\min\{d_{3,3}^2+d_{3,3}^2\ d_{3,3}^2\}=\min\{0+0;\ 0\}=0$
$d_{3,4}^3=\min\{d_{3,3}^2+d_{3,4}^2\ d_{3,4}^2\}=\min\{0+4;\ 4\}=4$
$d_{3,5}^3=\min\{d_{3,3}^2+d_{3,5}^2\ d_{3,5}^2\}=\min\{0+6;\ 6\}=6$
i=4 j=1-5
$d_{4,1}^3=\min\{d_{4,3}^2+d_{3,1}^2\ d_{4,1}^2\}=\min\{4+7;\ 5\}=5$
$d_{4,2}^3=\min\{d_{4,3}^2+d_{3,2}^2\ d_{4,2}^2\}=\min\{4+4;\ 2\}=2$
$d_{4,3}^3=\min\{d_{4,3}^2+d_{3,3}^2\ d_{4,3}^2\}=\min\{4+0;\ 4\}=4$
$d_{4,4}^3=\min\{d_{4,3}^2+d_{3,4}^2\ d_{4,4}^2\}=\min\{4+4;\ 0\}=0$
$d_{4,5}^3=\min\{d_{4,3}^2+d_{3,5}^2\ d_{4,5}^2\}=\min\{4+6;\ 4\}=4$

i=5 j=1-5
$\mathbf{d_{5,1}^3=min\{d_{5,3}^2+d_{3,1}^2d_{5,1}^2\}=min\{6+7;\ \infty\}=13}$
$\mathbf{d_{5,2}^3=min\{d_{5,3}^2+d_{3,2}^2d_{5,2}^2\}=min\{6+4;\ \infty\}=10}$
$d_{5,3}^3=\min\{d_{5,3}^2+d_{3,3}^2d_{5,3}^2\}=\min\{6+0;\ 6\}=6$
$d_{5,4}^3=\min\{d_{5,3}^2+d_{3,4}^2d_{5,4}^2\}=\min\{6+4;\ 4\}=4$
$d_{5,5}^3=\min\{d_{5,3}^2+d_{3,5}^2d_{5,5}^2\}=\min\{6+6;\ 0\}=0$
</td>
<td>
$$\begin{array}{cccccc} & 1(a) & 2(b) & 3(c) & 4(d) & 5(e) \quad \mathbf{k=4\ D^{(4)}} \\ 1(a) & 0 & 3 & 7 & 5 & \mathbf{13} \\ 2(b) & 3 & 0 & 4 & 2 & \mathbf{10} \\ 3(c) & 7 & 4 & 0 & 4 & 6 \\ 4(d) & 5 & 2 & 4 & 0 & 4 \\ 5(e) & \mathbf{13} & \mathbf{10} & 6 & 4 & 0 \end{array}$$
$\mathbf{d_{i,j}^4=min\{\ d_{i,4}^3+\ d_{4,j}^3;\ d_{i,j}^3\}.}$
</td>
</tr>
<tr>
<td>
i=1 j=1-5
$d_{1,1}^4=\min\{d_{1,4}^3+d_{4,1}^3\ d_{1,1}^3\}=\min\{5+5;\ 0\}=0$
$d_{1,2}^4=\min\{d_{1,4}^3+d_{4,2}^3\ d_{1,2}^3\}=\min\{5+2;\ 3\}=3$
$d_{1,3}^4=\min\{d_{1,4}^3+d_{4,3}^3\ d_{1,3}^3\}=\min\{5+4;\ 7\}=7$
$d_{1,4}^4=\min\{d_{1,4}^3+d_{4,4}^3\ d_{1,4}^3\}=\min\{5+0;\ 5\}=5$
$\mathbf{d_{1,5}^4=min\{d_{1,4}^3+d_{4,5}^3\ d_{1,5}^3\}=min\{5+4;\ 13\}=9}$
i=2 j=1-5
$d_{2,1}^4=\min\{d_{2,4}^3+d_{4,1}^3\ d_{2,1}^3\}=\min\{2+5;\ 3\}=3$
$d_{2,2}^4=\min\{d_{2,4}^3+d_{4,2}^3\ d_{2,2}^3\}=\min\{2+2;\ 0\}=0$
$d_{2,3}^4=\min\{d_{2,4}^3+d_{4,3}^3\ d_{2,3}^3\}=\min\{2+4;\ 4\}=4$
$d_{2,4}^4=\min\{d_{2,4}^3+d_{4,4}^3\ d_{2,4}^3\}=\min\{2+0;\ 2\}=2$
$\mathbf{d_{2,5}^4=min\{d_{2,4}^3+d_{4,5}^3\ d_{2,5}^3\}=min\{2+4;\ 10\}=6}$
i=3 j=1-5
$d_{3,1}^4=\min\{d_{3,4}^3+d_{4,1}^3d_{3,1}^3\}=\min\{4+5;\ 7\}=7$
$d_{3,2}^4=\min\{d_{3,4}^3+d_{4,2}^3d_{3,2}^3\}=\min\{4+2;\ 4\}=4$
$d_{3,3}^4=\min\{d_{3,4}^3+d_{4,3}^3d_{3,3}^3\}=\min\{4+4;\ 0\}=0$
$d_{3,4}^4=\min\{d_{3,4}^3+d_{4,4}^3d_{3,4}^3\}=\min\{4+0;\ 4\}=4$
$d_{3,5}^4=\min\{d_{3,4}^3+d_{4,5}^3d_{3,5}^3\}=\min\{4+4;\ 6\}=6$
i=4 j=1-5
$d_{4,1}^4=\min\{d_{4,4}^3+d_{4,1}^3\ d_{4,1}^3\}=\min\{0+5;\ 5\}=5$
$d_{4,2}^4=\min\{d_{4,4}^3+d_{4,2}^3\ d_{4,2}^3\}=\min\{0+2;\ 2\}=2$
$d_{4,3}^4=\min\{d_{4,4}^3+d_{4,3}^3\ d_{4,3}^3\}=\min\{0+4;\ 4\}=4$
$d_{4,4}^4=\min\{d_{4,4}^3+d_{4,4}^3\ d_{4,4}^3\}=\min\{0+0;\ 0\}=0$
$d_{4,5}^4=\min\{d_{4,4}^3+d_{4,5}^3\ d_{4,5}^3\}=\min\{0+4;\ 4\}=4$
i=5 j=1-5
$\mathbf{d_{5,1}^4=min\{d_{5,4}^3+d_{4,1}^3d_{5,1}^3\}=min\{4+5;\ 13\}=9}$
$\mathbf{d_{5,2}^4=min\{d_{5,4}^3+d_{4,2}^3d_{5,2}^3\}=min\{4+2;\ 10\}=6}$
</td>
<td>
$$\begin{array}{cccccc} & 1(a) & 2(b) & 3(c) & 4(d) & 5(e) \quad \mathbf{k=5\ D^{(5)}} \\ 1(a) & 0 & 3 & 7 & 5 & \mathbf{9} \\ 2(b) & 3 & 0 & 4 & 2 & \mathbf{6} \end{array}$$
</td>
</tr>
</table>

$d_{5,3}^4=\min\{d_{5,4}^3+d_{4,3}^3 d_{5,3}^3\}=\min\{4+4; 6\}=6$ $d_{5,4}^4=\min\{d_{5,4}^3+d_{4,4}^3 d_{5,4}^3\}=\min\{4+0; 4\}=4$ $d_{5,5}^4=\min\{d_{5,4}^3+d_{4,5}^3 d_{5,5}^3\}=\min\{4+4; 0\}=0$	3(c) 7 4 0 4 6 4(d) 5 2 4 0 4 5(e) **9** **6** 6 4 0 **$d_{i,j}^5=\min\{ d_{i,5}^4+ d_{5,j}^4; d_{i,j}^4\}$**
i=1 j=1-5 $d_{1,1}^5=\min\{d_{1,5}^4+d_{5,1}^4\ d_{1,1}^4\}=\min\{9+9; 0\}=0$ $d_{1,2}^5=\min\{d_{1,5}^4+d_{5,2}^4\ d_{1,2}^4\}=\min\{9+6; 3\}=3$ $d_{1,3}^5=\min\{d_{1,5}^4+d_{5,3}^4\ d_{1,3}^4\}=\min\{9+6; 7\}=7$ $d_{1,4}^5=\min\{d_{1,5}^4+d_{5,4}^4\ d_{1,4}^4\}=\min\{9+4; 5\}=5$ $d_{1,5}^5=\min\{d_{1,5}^4+d_{5,5}^4\ d_{1,5}^4\}=\min\{9+0; 9\}=9$ i=2 j=1-5 $d_{2,1}^5=\min\{d_{2,5}^4+d_{5,1}^4 d_{2,1}^4\}=\min\{6+9; 3\}=3$ $d_{2,2}^5=\min\{d_{2,5}^4+d_{5,2}^4 d_{2,2}^4\}=\min\{6+6; 0\}=0$ $d_{2,3}^5=\min\{d_{2,5}^4+d_{5,3}^4 d_{2,3}^4\}=\min\{6+6; 4\}=4$ $d_{2,4}^5=\min\{d_{2,5}^4+d_{5,4}^4 d_{2,4}^4\}=\min\{6+4; 2\}=2$ $d_{2,5}^5=\min\{d_{2,5}^4+d_{5,5}^4 d_{2,5}^4\}=\min\{6+0; 6\}=6$ i=3 j=1-5 $d_{3,1}^5=\min\{d_{3,5}^4+d_{5,1}^4\ d_{3,1}^4\}=\min\{6+9; 7\}=7$ $d_{3,2}^5=\min\{d_{3,5}^4+d_{5,2}^4\ d_{3,2}^4\}=\min\{6+6; 4\}=4$ $d_{3,3}^5=\min\{d_{3,5}^4+d_{5,3}^4\ d_{3,3}^4\}=\min\{6+6; 0\}=0$ $d_{3,4}^5=\min\{d_{3,5}^4+d_{5,4}^4\ d_{3,4}^4\}=\min\{6+4; 4\}=4$ $d_{3,5}^5=\min\{d_{3,5}^4+d_{5,5}^4\ d_{3,5}^4\}=\min\{6+0; 6\}=6$ i=4 j=1-5 $d_{4,1}^5=\min\{d_{4,5}^4+d_{5,1}^4\ d_{4,1}^4\}=\min\{4+9; 5\}=5$ $d_{4,2}^5=\min\{d_{4,5}^4+d_{5,2}^4\ d_{4,2}^4\}=\min\{4+6; 2\}=2$ $d_{4,3}^5=\min\{d_{4,5}^4+d_{5,3}^4\ d_{4,3}^4\}=\min\{4+6; 4\}=4$ $d_{4,4}^5=\min\{d_{4,5}^4+d_{5,4}^4\ d_{4,4}^4\}=\min\{4+4; 0\}=0$ $d_{4,5}^5=\min\{d_{4,5}^4+d_{5,5}^4\ d_{4,5}^4\}=\min\{4+0; 4\}=4$ i=5 j=1-5 $d_{5,1}^5=\min\{d_{5,5}^4+d_{5,1}^4 d_{5,1}^4\}=\min\{0+9; 9\}=9$ $d_{5,2}^5=\min\{d_{5,5}^4+d_{5,2}^4 d_{5,2}^4\}=\min\{0+6; 6\}=6$ $d_{5,3}^5=\min\{d_{5,5}^4+d_{5,3}^4 d_{5,3}^4\}=\min\{0+6; 6\}=6$ $d_{5,4}^5=\min\{d_{5,5}^4+d_{5,4}^4 d_{5,4}^4\}=\min\{0+4; 4\}=6$ $d_{5,5}^5=\min\{d_{5,5}^4+d_{5,5}^4 d_{5,5}^4\}=\min\{0+0; 0\}=0$	1(a) 2(b) 3(c) 4(d) 5(e) **k=6 $D^{(6)}$** 1(a) 0 3 7 5 9 2(b) 3 0 4 2 6 3(c) 7 4 0 4 6 4(d) 5 2 4 0 4 5(e) 9 6 6 4 0

Таким чином, на рисунку 6.2 показані зважений граф, представлення його за допомогою матриці ваг, а також матриця відстаней між будь-якими його двома вершинами, яка отримана за допомогою алгоритму Флойда.



а

	1(a)	2(b)	3(c)	4(d)	5(e)
1(a)	0	3	∞	7	∞
2(b)	3	0	4	2	∞
3(c)	∞	4	0	4	6
4(d)	7	2	4	0	4
5(e)	∞	∞	6	4	0

б

	1(a)	2(b)	3(c)	4(d)	5(e)
1(a)	0	3	7	5	9
2(b)	3	0	4	2	6
3(c)	7	4	0	4	6
4(d)	5	2	4	0	4
5(e)	9	6	6	4	0

в

Рисунок 6.2 Зважений граф (а), його матриці суміжності (б) та відстаней (в)

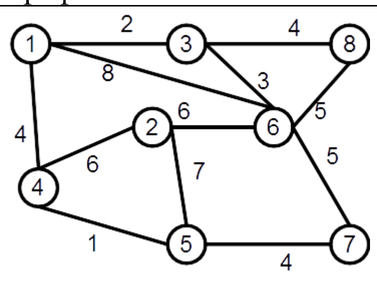
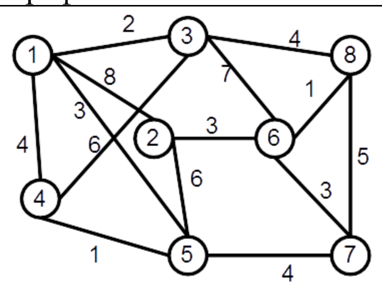
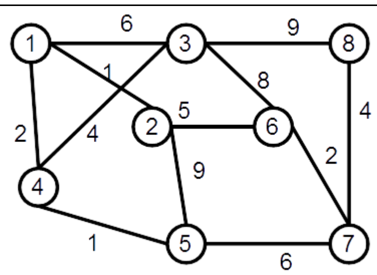
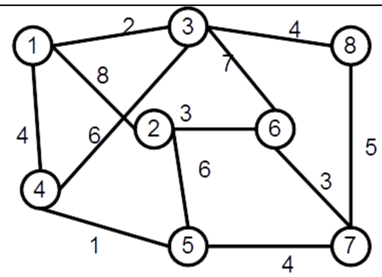
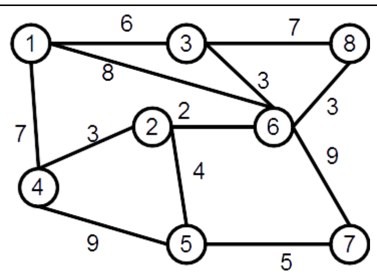
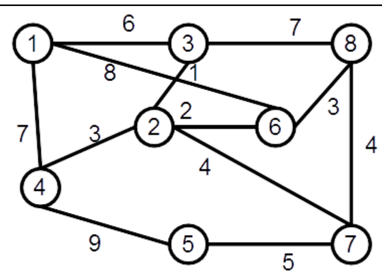
<u>Завдання до лабораторної роботи №6</u>

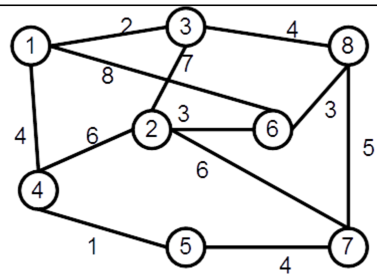
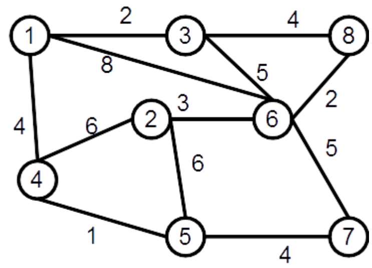
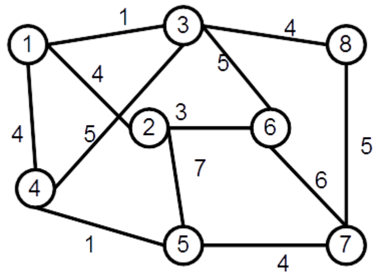
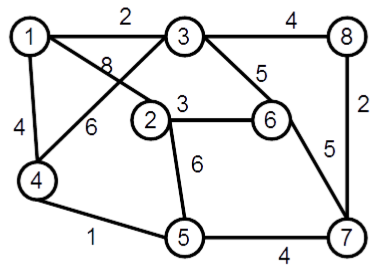
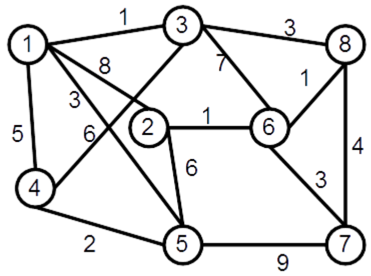
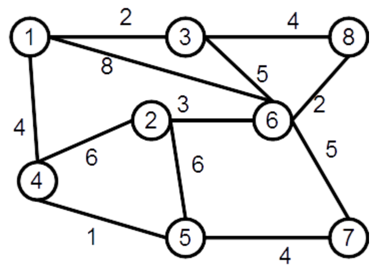
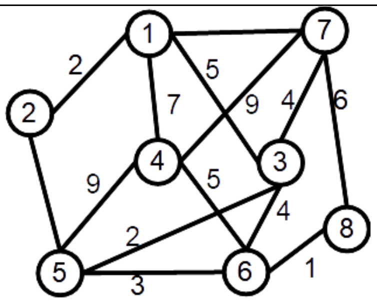
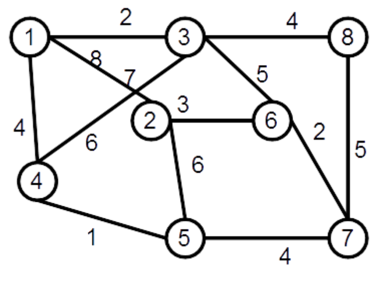
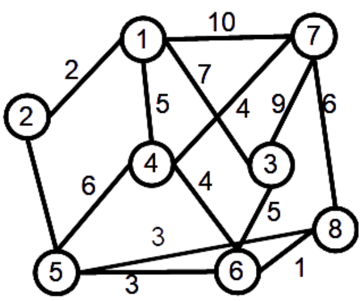
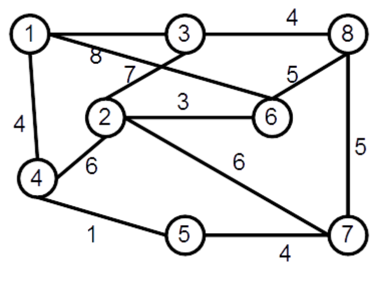
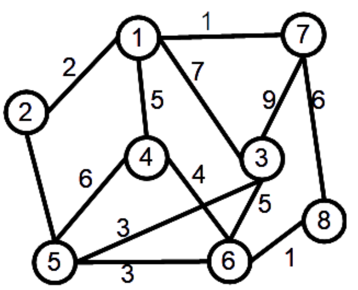
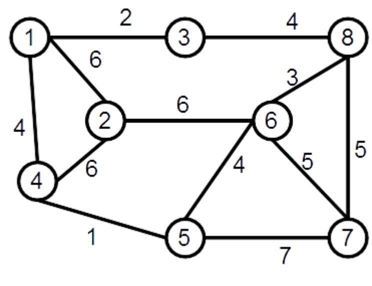
1. Побудувати за допомогою алгоритму Дейкстри (див. Лістинг 6.1) найкоротший шлях із заданої вершини графа за варіантами завдань заповнити таблиці 6.1 та 6.2
2. Показати графічно хід побудови найкоротшого шляху за алгоритмом Дейкстри, використовуючи процедуру *Dijkstra*(*G*, *s*)

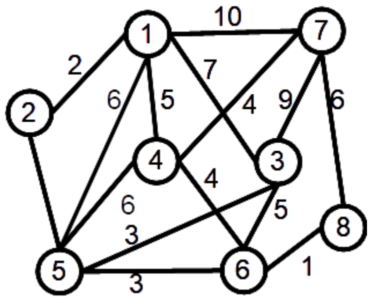
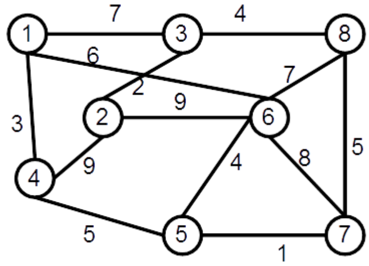
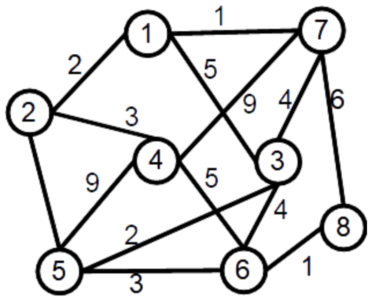
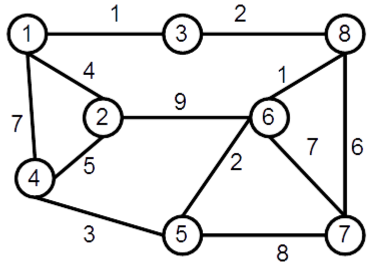
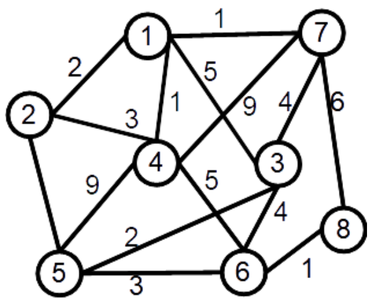
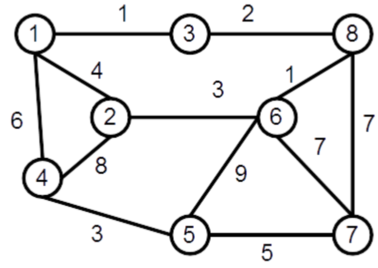
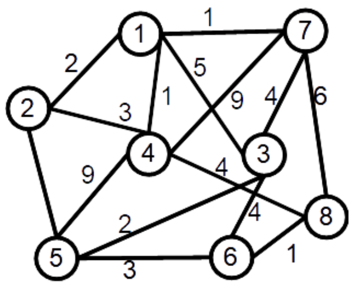
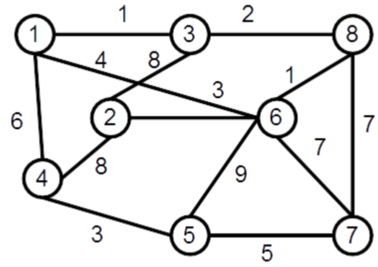
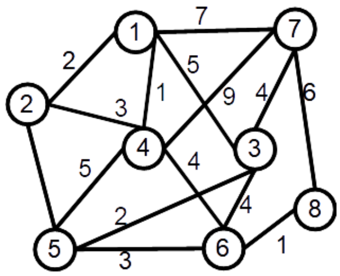
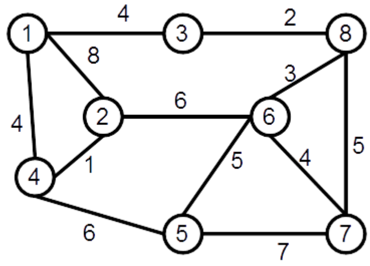
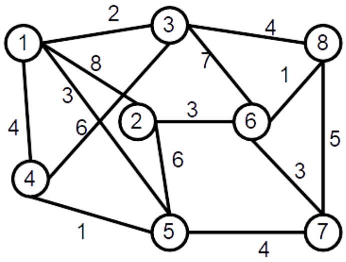
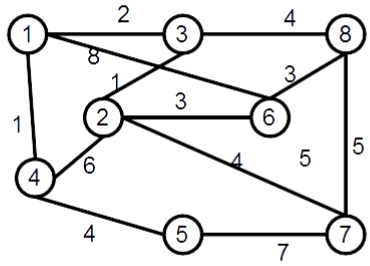
3 Запрограмувати алгоритм Дейкстри. Запустити програму, показати результат за варіантами завдань. Показати, що результати ручної та автоматизованої побудови найкоротшего шляху співпадають
4. Показати хід побудови матриці відстаней за алгоритмом Флойда, використовуючи процедуру *Floyd* (*W* [1..*n*, 1..*n*]). Заповнити таблицю 6.3.
5.Запрограмувати алгоритм Флойда. Запустити програму, показати результат за варіантами завдань. Показати, що результати ручної та автоматизованої побудови матриць відстаней співпадають
6. Порівняти між собою результати моделювання за алгоритмом Дейкстри та Флойда

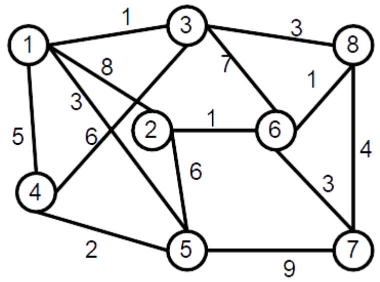
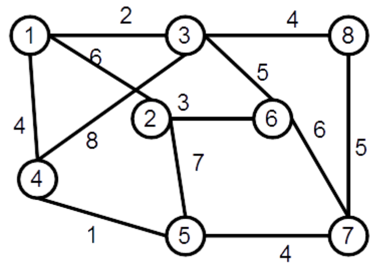
7. Звіт повинен містити:
– хід побудови шляхів за алгоритмами Дейкстри та Флойда заповнені таблиці 6.1,2 та 6.3
– лістинги програм, результати виконання програм з контрольного прикладу за варіантами завдань,
– порівняння алгоритмів Дейкстри та Флойда в висновках щодо практичної роботи №6.

Варіанти завдань*:

№	Граф	№	Граф
1		17	
2		18	
9		25	

10		26	
11		27	
12		28	
13		29	
14		30	
3		19	

4		20	
5		21	
6		22	
7		23	
8		24	
15		31	

16		32	

*Якщо в якомусь графі пропущена довжина ребра або значення довжини дублюються студент може виправити цю помилку зважаючи на власні міркування.