МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ
«ЧЕРНІГІВСЬКА ПОЛІТЕХНІКА»
ННІ ЕЛЕКТРОННИХ ТА ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ
КАФЕДРА ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ ТА ПРОГРАМНОЇ ІНЖЕНЕРІЇ

КОМП'ЮТЕРНА ДИСКРЕТНА МАТЕМАТИКА
ТЕОРІЯ ГРАФІВ

Методичні вказівки
до виконання лабораторних робіт
з дисципліни **«Комп'ютерна дискретна математика»**
для здобувачів вищої освіти спеціальності
121 – **«Інженерія програмного забезпечення»**
першого (бакалаврського) рівня вищої освіти

Обговорено і рекомендовано
на засіданні кафедри ІТтаПІ
Протокол №1
від 31.08.21

Чернігів НУ «Чернігівська політехніка» 2021

Комп'ютерна дискретна математика. Теорія графів. Методичні вказівки до виконання лабораторних робіт з дисципліни «Комп’ютерна дискретна математика» для здобувачів вищої освіти спеціальності 121 – «Інженерія програмного забезпечення» *першого (бакалаврського)* рівня вищої освіти / Укл. О.В. Трунова – Чернігів: НУ «Чернігівська політехніка», 2021. – 52 с., укр. мовою.

Укладачі: Трунова Олена Василівна, к.пед.н., доцент кафедри інформаційних технологій та програмної інженерії

Відповідальний за випуск: Білоус І.В., завідувач кафедри інформаційних технологій та програмної інженерії, к.т.н., доцент,

Рецензент: Ткач Юлія Миколаївна, завідувач кафедри кібербезпеки та математичного моделювання, д.пед.н., професор

Зміст

Вступ

На рівні наших буденних уявлень граф – це будь-які об'єкти (зазвичай їх малюють у вигляді точок або кружків), поєднані лініями чи стрілками. Зокрема, у вигляді графів можна представити електричні схеми, маршрути перевезень, схеми взаємозв'язків підрозділів підприємства, грошові та ресурсні потоки, системи керування різними об'єктами, тощо. З давніх часів люди помітили, що графи мають спільні властивості незалежно від того, який реальний об'єкт вони представляють. На основі вивчення цих властивостей і виникла наука під назвою «Теорія графів». У процесі її розвитку з'ясувалося, що вона тісно пов'язана з іншими розділами математики: теорією множин і комбінаторним аналізом. Тому в технічних ВНЗ, зазвичай, теорію графів, поряд з теорією множин, комбінаторикою та деякими іншими розділами математики, вивчають у курсі під назвою «Комп'ютерна дискретна математика».

Метою курсу «Комп'ютерна дискретна математика» є демонстрація прикладного характеру математичної теорії при розв'язанні різного роду задач, які виникають в різних областях науки, техніки і виробництва, формування основ математичного моделювання прикладних задач, закріплення та розвиток фахових компетентностей бакалавра в галузі знань *12 – Інформаційні технології* із застосування у повсякденній діяльності та розробки нових методів обробки інформації.

Вивчення курсу «Комп'ютерна дискретна математика» надає змогу здобувачам вищої освіти оволодіти знаннями теоретичних та практичних методів розв'язання типових математичних задач, забезпечує успішне виконання курсових проектів, бакалаврських

випускних робіт і дипломних проектів, науково-дослідної роботи.

Обов'язковою умовою викладання дисципліни є проведення лабораторного практикуму із застосуванням сучасних персональних комп'ютерів.

Практика викладання курсу «Комп'ютерна дискретна математика» для інженерів показує, що для досягнення хороших успіхів у засвоєнні знань, одержаних на заняттях з інформатики та математики, є можливим лише за умови, коли ці дві складові поєднуються. При поєднанні цих складових спостерігається більша зацікавленість здобувачів вищої освіти у вивченні матеріалу, ніж на звичайних практичних чи лабораторних заняттях. Це зумовлено, в першу чергу, можливістю оперативного експерименту та творчого підходу при вирішенні тих чи інших конкретних завдань.

Метою виконання циклу лабораторних робіт є: закріплення теоретичних знань з дисципліни «Комп'ютерна дискретна математика»; придбання практичних навичок з моделювання об'єктів і операцій над ними, що вивчаються в курсі; знайомство з технологіями мережевого програмування з боку клієнта.

Пропоновані у виданні лабораторні роботи, використовуються при проведенні занять зі здобувачами 2-го курсу ОПП 121 – «Інженерія програмного забезпечення» в ході вивчення курсу «Комп'ютерна дискретна математика».

Для захисту лабораторної роботи здобувач вищої освіти повинен відповісти на всі контрольні запитання з методичних вказівок та на два запитання за вибором викладача з лекційного курсу за темою лабораторної роботи. За кожну лабораторну роботу здобувач отримує оцінку з урахуванням якості оформлення звіту та повноти відповідей на запитання при захисті лабораторної роботи.

ЛАБОРАТОРНА РОБОТА №1
Матричні способи представлення графів

Мета роботи. Метою роботи є вивчення матричних способів представлення графів.

Короткі теоретичні відомості

Останнім часом теорія графів стала простим, доступним і потужним засобом вирішення питань, що відносяться до широкого кола проблем. Це проблеми проектування інтегральних схем і схем управління, дослідження автоматів, логічних ланцюгів, блок-схем програм, економіки та статистики, хімії та біології, теорії розкладів і дискретної оптимізації.

Основні означення

Граф G задається множиною точок або вершин $x_1,x_2,...,x_n$ (яка позначається *X*) і множиною ліній або *ребер* $a_1,a_2,...,a_n$ (яка позначається *A*), що з'єднують між собою всі або частину цих точок. Таким чином, граф *G* повністю задається (і позначається) парою (*X, A*).

Якщо ребра орієнтовані, що зазвичай показується стрілкою, то вони називаються дугами, і граф з такими ребрами називається орієнтованим графом (рисунок 1.1 (а)). Якщо ребра не мають орієнтації, то граф називається неорієнтованим (рисунок 1.1 (б)). У разі коли *G = (X, A)* є орієнтованим графом і ми хочемо знехтувати спрямованістю дуг з множини А, то неорієнтований граф, відповідний *G*, також будемо позначати як *G = (X, A)*.

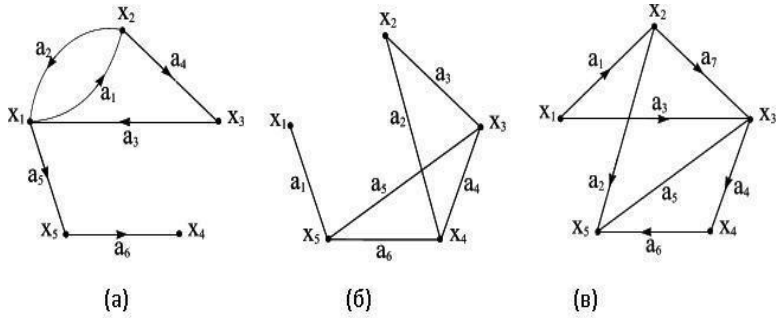


Рисунок 1.1 (а) – орієнтований граф; (б) – неорієнтований граф; (в) – змішаний граф.

Якщо дуга позначається впорядкованою парою, що складається з початкової та кінцевої вершин (тобто двома кінцевими вершинами дуги), то її напрямок вважається заданим від першої вершини до другої. Так, наприклад, на рисунку 1.1 (а) позначення (x_1, x_2) відноситься до дуги a_1, а (x_2,x_1) – до дуги a_2.

Частіше опис орієнтованого графа G полягає в заданні множини вершин X та відповідності Γ, яка показує, як між собою пов'язані вершини. Відповідність Γ називається відображенням множини X в X, а граф в цьому випадку позначається парою $G = (X, \Gamma)$.

Для графа на рисунку 1(а) маємо $\Gamma(x_1)=\{x_2, x_5\}$, тобто вершини x_2 і x_5 є кінцевими вершинами дуг, у яких початковою вершиною є x_1.

$\Gamma(x_2)=\{x_1,x_3\}$, $\Gamma(x_3)=\{x_1\}$, $\Gamma(x_4)=\varnothing$ – порожня множина, $\Gamma(x_5)=\{x_4\}$.

У випадку неорієнтованого графа або графа, який містить і дуги, і неорієнтовані ребра (наприклад, графи, зображені на рисунках 1(б) і 1(в)), передбачається, що відповідність Γ задає такий еквівалентний орієнтований граф, який виходить з початкового графа заміною кожного неорієнтованого ребра двома протилежно спрямованими

дугами, які з'єднують ті ж самі вершини. Так, наприклад, для графа, наведеного на рисунку 1(б), маємо $\Gamma(x_5)=\{x_1,x_3,x_4\}$, $\Gamma(x_1)=\{x_5\}$ та ін.

Оскільки пряма відповідність або образ вершини $\Gamma(x_i)$ є множиною таких вершин $x_j \in X$, для яких у графі G існує дуга (x_i,x_j), то через $\Gamma^{-1}(x_i)$ природно позначити множину вершин x_k, для яких в G існує дуга (x_k,x_i). Таке відношення прийнято називати зворотною відповідністю або прообразом вершини. Для графа, зображеного на рисунку 1(а), маємо

$\Gamma^{-1}(x_1)=\{x_2,x_3\}$, $\Gamma^{-1}(x_2)=\{x_1\}$ та ін.

Для неорієнтованого графа $\Gamma^{-1}(x_i)=\Gamma(x_i)$ для всіх $x_j \in X$.

Коли відображення Γ діє не на одну вершину, а на множину вершин $X_q=\{x_1,x_2,...,x_q\}$, то під $\Gamma(X_q)$ розуміють об'єднання $\Gamma(x_1) \cup \Gamma(x_2) \cup ... \cup \Gamma(x_q)$, тобто $\Gamma(X_q)$ є множиною таких вершин $x_j \in X$, що для кожної з них існує дуга (x_i,x_j) в G, де $x_i \in X_q$. Для графа, наведеного на рисунку 1(а), $\Gamma(\{x_2,x_5\})=\{x_1,x_3,x_4\}$ й $\Gamma(\{x_1,x_3\})=\{x_2,x_5,x_1\}$.

Відображення $\Gamma(\Gamma(x_i))$ записується як $\Gamma(\Gamma(x_i))$. Аналогічно «потрійне» відображення $\Gamma(\Gamma(\Gamma(x_i)))$ записується як $\Gamma^3(x_i)$ та ін. Для графа, наведеного на рисунку 1(а), маємо:

$\Gamma^2(x_1)=\Gamma(\Gamma(x_1))=\Gamma(\{x_2,x_5\})=\{x_1,x_3,x_4\}$;

$\Gamma^3(x_1)=\Gamma(\Gamma^2(x_1))=\Gamma(\{x_1,x_3,x_4\})=\{x_2,x_5,x_1\}$ та ін.

Аналогічно визначаються позначення $\Gamma^{-2}(x_i)$, $\Gamma^{-3}(x_i)$ і т.п.

Дуги $a=(x_i,x_j)$, $x_i \neq x_j$, які мають спільні кінцеві вершини, називаються суміжними. Дві вершини x_i й x_j називаються суміжними, якщо яка-небудь з двох дуг (x_i,x_j) і (x_j,x_i) або обидві одночасно присутні в графі. Так, наприклад, на рисунку 1.2 дуги a_1, a_{10}, a_3 й a_6 як і вершини x_5 та x_3, є суміжними, в той час як дуги a_1 та a_5 або вершини x_1 та x_4 не є суміжними.

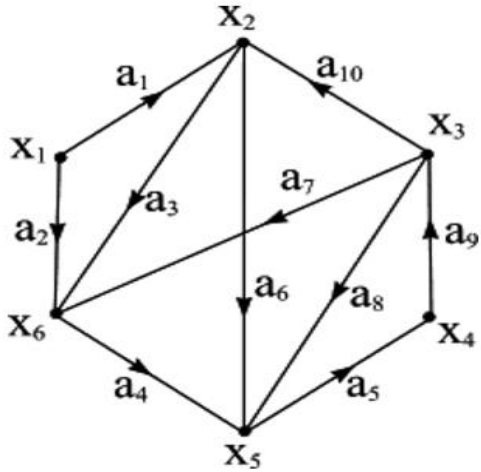


Рисунок 6.2 – Змішаний граф

Число дуг, які беруть початок у вершині x_i, називається півстепенем початку вершини x_i, і, аналогічно, число дуг, які мають x_i за свою кінцеву вершину, називається півстепенем кінця вершини x_i.

Таким чином, на рисунку 1.2 півстепінь початку вершини x_3, позначається як $deg^+(x_3)$, дорівнює $\Sigma\Gamma(x_3)^+ = 3$, і півстепінь кінця вершини x_3, яка позначається як $deg^-(x_3)$, дорівнює $\Sigma\Gamma^{-1}(x_3)^- = 1$.

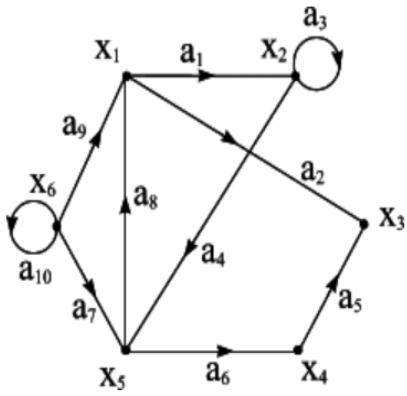
Очевидно, що сума півстепенів кінця всіх вершин графа, а також сума півстепенів початку всіх вершин дорівнює загальній кількості дуг графа G, тобто

$$\sum_{i=1}^{n} \deg^+(x_i) = \sum_{i=1}^{n} \deg^-(x_i) = m, \qquad (1.1)$$

де n- число вершин і m- число дуг графа G.

Для неорієнтованого графа $G=(X,\Gamma)$ степінь вершини x_i визначається аналогічно – за допомогою співвідношення $deg(x_i) \equiv |\Gamma(x_i)| = |\Gamma^{-1}(x_i)|$.

Петлею називається дуга, початкова та кінцева вершини якої збігаються. На рисунку 1.3, наприклад, дуги a_3 й a_{10} є петлями.



	x1	x2	x3	x4	x5	x6
x1	0	1	1	0	0	0
x2	0	1	0	0	1	0
x3	0	0	0	0	0	0
x4	0	0	1	0	0	0
x5	1	0	0	1	0	0
x6	1	0	0	0	1	1

Рисунок 1.3 – Орієнтований граф і його матриця суміжності

Матричні представлення
Матриця суміжності

Нехай задано граф G, його матриця суміжності позначається як $A=[a_{ij}]$ та визначається наступним чином:

$a_{ij}=1$, якщо в G існує дуга (x_i,x_j);

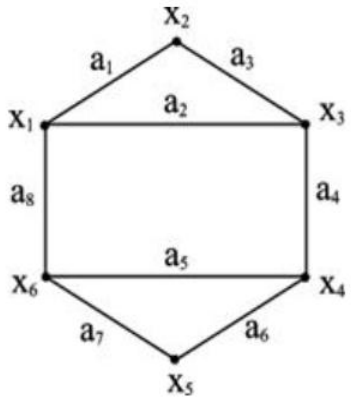
$a_{ij}=0$, якщо в G немає дуги (x_i,x_j).

Таким чином, на рисунку 1.3, наведено граф і відповідну йому матрицю суміжності графа.

Матриця суміжності повністю визначає структуру графа. Наприклад, сума всіх елементів рядка x_i матриці дає півстепінь початку вершини x_i, а сума елементів стовпця x_i – півстепінь кінця вершини x_i. Множина стовпців, які мають 1 у рядку x_i, є множина $\Gamma(x_i)$, а множина рядків, які мають 1 в стовпчику x_i, співпадає з множиною $\Gamma^{-1}(x_i)$.

Петлі на графі являють собою елементи, які мають 1 на головній діагоналі матриці, наприклад, a_{22}, a_{66} для графу зображеного на рисунку 1.3.

У випадку неорієнтованого графа матриця суміжності є симетричною відносно головної діагоналі (рис. 1.4)



	x_1	x_2	x_3	x_4	x_5	x_6
x_1	0	1	1	0	0	1
x_2	1	0	1	0	0	0
x_3	1	1	0	1	0	0
x_4	0	0	1	0	1	1
x_5	0	0	0	1	0	1

Рисунок 1.4 – Неорієнтований граф і його матриця суміжності

Матриця інцидентності

Нехай є граф G з n вершинами й m дугами. Матриця інцидентності графа G позначається через $B=[b_{ij}]$ і є матрицею розміром $n \times m$, яка визначається наступним чином:

$b_{ij} = 1$, якщо x_i є початковою вершиною дуги a_j;

$b_{ij} = -1$, якщо x_i є кінцевою вершиною дуги a_j;

$b_{ij} = 0$, якщо x_i не є кінцевою вершиною дуги a_j.

Для графа, наведеного на рисунку 1.3, матриця інцидентності має вигляд:

	a_1	a_2	a_3	a_4	a_5	a_6	a_7	a_8	a_9	a_{10}
X_1	1	1	0	0	0	0	0	-1	-1	0
X_2	-1	0	1	1	0	0	0	0	0	0
X_3	0	-1	0	0	-1	0	0	0	0	0
X_4	0	0	0	0	1	-1	0	0	0	0
X_5	0	0	0	-1	0	1	-1	1	0	0
X_6	0	0	0	0	0	0	1	0	1	1

Оскільки, кожна дуга інцидентна двом різним вершинам (за винятком, коли дуга утворює петлю), то кожний стовпчик містить один елемент, який дорівнює 1, і один, який дорівнює – 1. Петля в матриці інцидентності не має адекватного математичного представлення (у програмній реалізації допустиме задання одного елемента b_{ij}=1).

Якщо G є неорієнтованим графом (рисунок 1.4), то його матриця інцидентності визначається наступним чином:

b_{ij} = 1, якщо x_i є кінцевою вершиною дуги a_j;

b_{ij} = 0, якщо x_i не є кінцевою вершиною дуги a_j.

	a_1	a_2	a_3	a_4	a_5	a_6	a_7	a_8
X_1	1	1	0	0	0	0	0	1
X_2	1	0	1	0	0	0	0	0
X_3	0	1	1	1	0	0	0	0
X_4	0	0	0	1	1	1	0	0
X_5	0	0	0	0	0	1	1	0
X_6	0	0	0	0	1	0	1	1

Матриця інцидентності, як спосіб задання графів, успішно застосовується при описі мультиграфів (графів, у яких суміжні вершини можуть з'єднуватися декількома паралельними дугами).

Порядок виконання роботи

1. Індивідуальні завдання для виконання лабораторної роботи «Матричне представлення графів» представлені в Додатку А (див. табл. 1).

2. Скласти алгоритм програми, яка реалізує перехід із заданого способу матричного представлення графа в інший, враховуючи при цьому початковий тип графа (неорієнтований, орієнтований, змішаний).

3. Створити програму, яка реалізує перехід із заданого способу матричного представлення графа в інший. Передбачити побудову графу та вивід результатів роботи програми на екран.

4. Інтерфейсна частина повинна включати:

a) заголовок із зазначенням назви роботи, інформації про автора;
b) поле для задання точок графу та ребер;
c) поле для виводу матриці суміжності та матриці інцидентності;
d) кнопку для активізації виводу матриці суміжності;
e) кнопку для активізації виводу матриці інцидентності;
f) кнопку для вибору орієнтованого та неорієнтованого графа.

Приклад виконання роботи

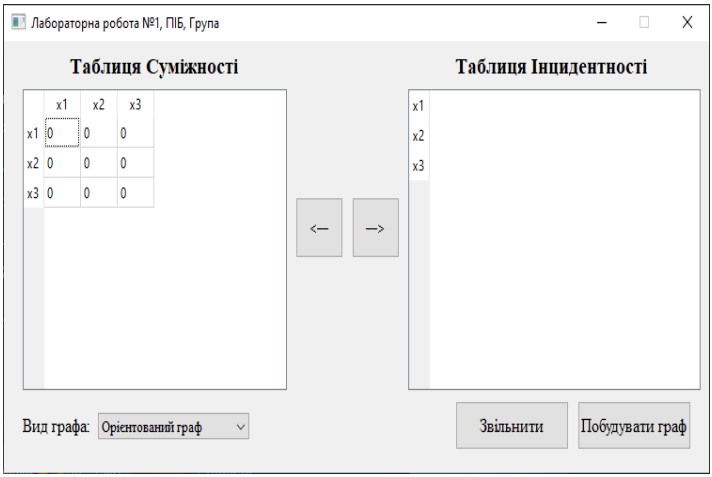


Рисунок 1.5 – Головне вікно програми.

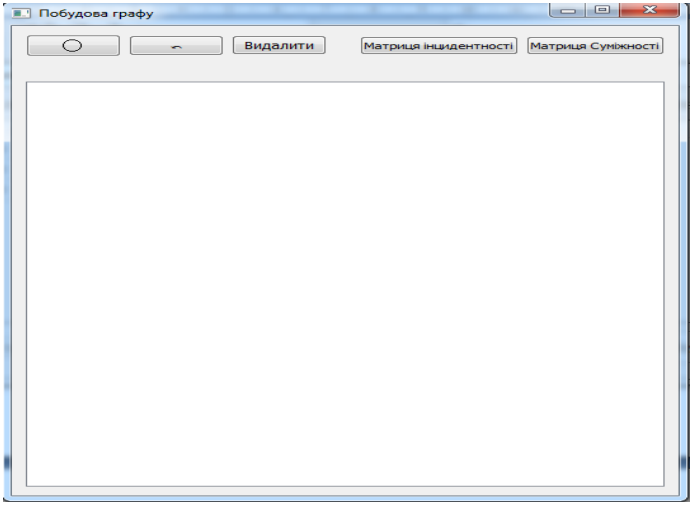


Рисунок 1.6 – Допоміжне вікно програми для побудови графа.

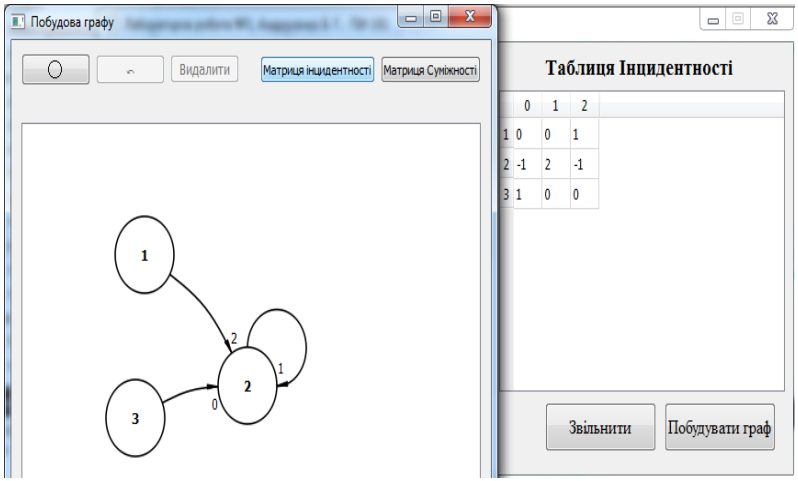


Рисунок 1.7 – Побудова з графа матриці інцидентності.

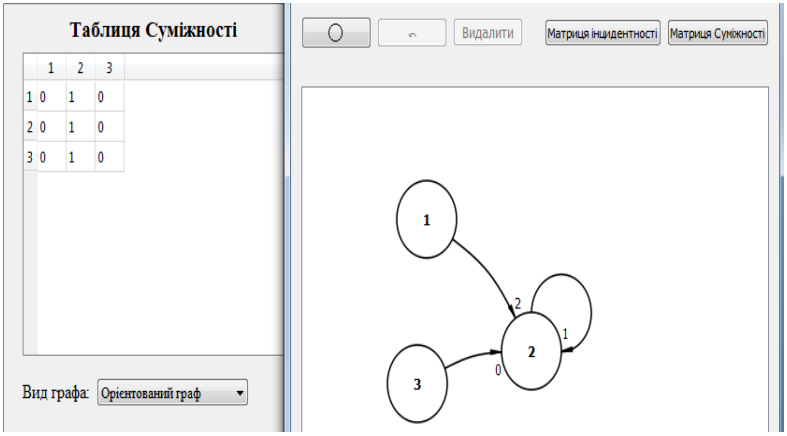


Рисунок 1.8 – Побудова з графа матриці суміжності.

Зміст звіту по роботі

1. Назва та мета роботи.
2. Блок-схема програми за варіантом.
3. Контрольний приклад і результати роботи програми.
4. Висновки по роботі.

Контрольні питання

1. Перерахуйте основні способи представлення графів.
2. Покажіть на прикладі пряму і зворотну відповідність для заданої вершини.
3. Чому дорівнює сума ступенів усіх вершин неорієнтованого графа?
4. У чому відмінності матричного представлення орієнтованих і неорієнтованих графів?
5. У чому особливості представлення графа матрицею суміжності?
6. У чому особливості представлення графа матрицею інцидентності?

ЛАБОРАТОРНА РОБОТА №2
Знаходження ейлеревого циклу в неорієнтованому графі

Мета роботи. Метою роботи є вивчення алгоритмів обходу неорієнтованого графа і знаходження ейлеревого циклу в неорієнтованому графі.

Основні означення

Графом *G(V,E)* називається сукупність двох непорожніх множин – непорожньої множини *V* (множини вершин) і множини *E* двоелементних підмножин множини *V* (*E* – множина ребер). Число вершин графа *G* $p=|V|$, а число ребер – $q=|E|$. Якщо деяка пара вершин з'єднана кількома ребрами, то такий граф називається *мультиграфом*. Ребро, що починається і закінчується в одній вершині, називається *петлею*. Граф без кратних ребер и петель називається *простим.*

Нехай v_1, v_2 – вершини, $e = (v_1,v_2)$ – ребро, у якого початкова і кінцева вершини збігаються. Тоді вершина v_1 і ребро *e інцидентні*, ребро *e* і вершина v_2 також інцидентні. Два ребра, які інцидентні одній вершині, називаються *суміжними*; дві вершини, які інцидентні одному ребру, також називаються суміжними. *Степенем вершини* називається число ребер, які інцидентні даній вершині. Зазвичай граф зображають за допомогою діаграми: вершини – точками (або кружечками), ребра – лініями.

Відомо багато способів представлення графів в пам'яті комп'ютера, які розрізняються за обсягом використаної пам'яті і швидкістю виконання операцій над графами. Подання графа за допомогою квадратної булевої матриці, що відображає суміжності вершин, називається *матрицею суміжності.*

Маршрутом M в графі називається така послідовність вершин і ребер, які чергуються:

$$M : v_1, e_1, v_2, e_2, ..., e_{k-1}, v_k,$$

в який будь-які два сусідніх елементи інцидентні. В орієнтованому графі v_i – початок ребра e_i, v_{i+1} – кінець ребра e_i. Маршрут можна задати як послідовністю вершин, так і послідовністю ребер. Довжиною маршруту називається число ребер, які в ньому містяться. Відстанню між двома вершинами називається довжина найкоротшого маршрута, що їх з'єднує. Ланцюгом будемо називати маршрут, всі ребра якого різні, а простим ланцюгом назвемо ланцюг, в якому всі вершини, можливо, крім крайніх, – різні.

Якщо перша і остання вершини збігаються, то такий ланцюг буде *циклом*. Цикл, у якого всі вершини різні (за винятком першої та останньої), називається *простим циклом. Зв'язний граф* – це граф, в якому існує ланцюг між будь-якою парою вершин *v, u*; іноді такий граф називають *однозв'язним*.

Цикл, що містить всі ребра графа, називається *ейлеревим циклом*. ***Зв'язний неорієнтований граф є ейлеревим тоді і тільки тоді, коли кожна вершина має парну степінь***. Шлях, який включає кожне ребро графа тільки один раз називається ейлеревим шляхом. Ейлеревий шлях, який не є ейлеревим циклом називається *власним ейлеревим шляхом.*

Наведемо ***алгоритм побудови ейлеревого циклу*** в ейлеревому мультиграфі. Цей алгоритм задається наступними правилами:

1. Вибрати довільно деяку вершину *a*.

2. Вибрати довільно деяке ребро *i*, інцидентне *a*, і привласнити йому номер 1 (назвемо це ребро пройденим).

3. Кожне пройдене ребро викреслити і привласнити йому номер, на одиницю більший номера попереднього викресленого ребра.

4. Перебуваючи у вершині *x*, не вибирати ребро, що з'єднує *x* з *a*, якщо є можливість іншого вибору.

5. Перебуваючи у вершині *x*, не вибирати ребро, яке є перешийком (тобто ребром, при видаленні якого граф, що утворився не викресленими ребрами, розпадається на дві компоненти зв'язності, кожна з яких має хоча б по одному ребру).

6. Після того як в графі будуть пронумеровані всі ребра, утворюється Ейлерів цикл, до того ж порядок нумерації відповідає послідовності обходу ребер.

Алгоритм побудови ейлеревого циклу може бути також побудований на основі алгоритму пошуку в глибину. Ребро вважається оберненим, якщо немає жодного непоміченого ребра, що виходить з поточної вершини. Якщо такі ребра записувати по мірі знаходження в стек, то по завершенню обходу в глибину в ньому буде зберігатися цикл Ейлера.

Порядок виконання роботи

1. Індивідуальні завдання для виконання лабораторної роботи «Знаходження ейлеревого циклу в неорієнтованому графі» представлені в Додатку А (див. табл. 2).

2. Скласти алгоритм знаходження ейлеревого циклу із заданого неорієнтованого графу.

3. Реалізувати програму з інтерфейсом користувача, яка буде шукати ейлеровий цикл. Результуючий цикл або ланцюг мусить буди графічно відображеним на самому графі. Якщо ланцюг або цикл знайти неможливо, програма повинна повідомити користувача про це.

4. Інтерфейс повинен включати:

a) вказівку на автора роботи (ім’я у заголовку, вікно з повідомленням, тощо);
b) поле введення матриці суміжності графа;
c) поле із графічно зображеним графом;
d) поле для текстового відображення ейлеревого циклу;
e) кнопку для пошуку ейлеревого циклу за заданою матрицею;
f) кнопку для редагування введених у матрицю значень.

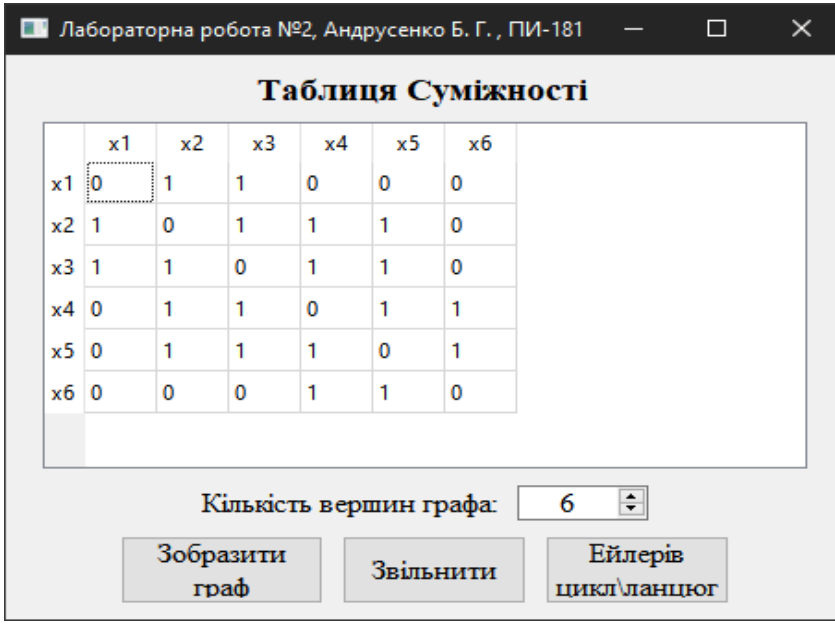
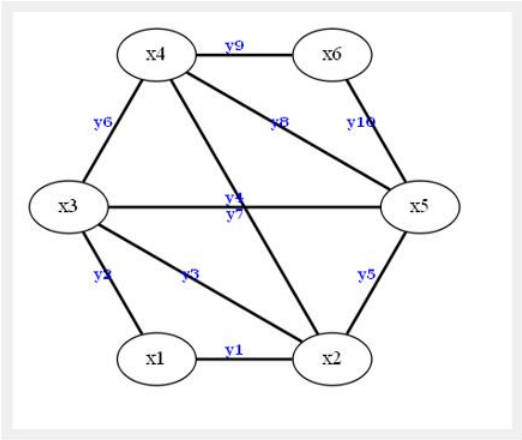


Рисунок 2.1– Загальний інтерфейс програми



Відповідь:
x1x3 : y2
x3x5 : y7
x5x6 : y10
x6x4 : y9
x4x5 : y8
x5x2 : y5
x2x4 : y4
x4x3 : y6
x3x2 : y3
x2x1 : y1

Рисунок 2.2 – Ейлеровий шлях заданого графа

Зміст звіту по роботі

1. Тема і мета лабораторної роботи.
2. Блок-схема алгоритму пошуку ейлеревого циклу.
3. Зображення інтерфейсу програми.
4. Результати виконання роботи на контрольному прикладі.
5. Висновки по роботі.

Контрольні питання

1. Означення ейлеревого циклу.
2. Умова існування ейлеревого циклу.
3. Алгоритм побудови ейлеревого циклу.

ЛАБОРАТОРНА РОБОТА №3
Розфарбування графа

Мета роботи. Метою роботи є вивчення способу правильного розфарбування графа на основі евристичного алгоритму.

Короткі теоретичні відомості

Різноманітні завдання, що виникають при плануванні виробництва, складанні графіків огляду, зберіганні і транспортуванні товарів і т.д., можуть бути представлені часто як задачі теорії графів, тісно пов'язані з так званим «завданням розфарбування». Графи, що розглядаються в даній лабораторній роботі, є **неорієнтованими і не містять петель.**

Основні означення

Граф G називають *r-хроматичним,* якщо його вершини можуть бути розфарбовані з використанням r кольорів (фарб) так, що не знайдеться двох суміжних вершин одного кольору. Найменше число r, таке що граф G є r-хроматичним, називається хроматичним числом графа G і позначається $\chi(G)$. Задача знаходження хроматичного числа графа називається задачею розфарбування графа. Відповідне цьому числу розфарбування вершин розбиває множину вершин графу на r підмножин, кожна з яких містить вершини одного кольору. Ці множини є незалежними, оскільки в межах однієї множини немає двох суміжних вершин.

Задача знаходження хроматичного числа довільного графу була предметом багатьох досліджень в кінці XIX і в

XX столітті. З цього питання отримано багато цікавих результатів.

Хроматичне число графа не можна знайти, знаючи тільки кількість вершин і ребер графа. Недостатньо також знати степінь кожної вершини, щоб обчислити хроматичне число графа. При відомих величинах *n* (число вершин), *m* (число ребер) і $deg(x_1), \ldots, deg(x_n)$ (степені вершин графа) можна отримати лише верхню і нижню оцінки для хроматичного числа графа.

Приклад розфарбування графа наведений на рисунку 3.1. Цей граф є однією із «заборонених» фігур використовуваних для визначення планарності. Цифрами «1» і «2» позначені кольори вершин.

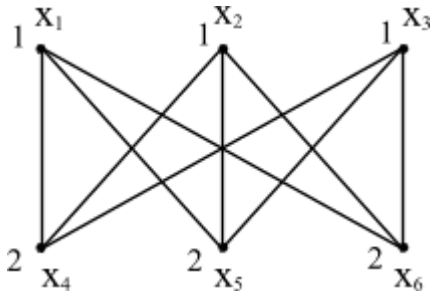


Рисунок 3.1.

Максимальне число незалежних вершин графа $\alpha(G)$, дорівнює потужності найбільшої множини попарно-несуміжних вершин, збігається з потужністю найбільшої множини вершин в *G*, які можуть бути пофарбовані в один колір, відповідно:

$$\gamma(\mathrm{G}) \geq \left[\frac{\mathrm{n}}{\alpha(\mathrm{G})} \right],$$

де *n* – число вершин графа *G*,
[*x*] – найбільше ціле число, що не перевищує *x*.

Ще одна нижня оцінка для $\gamma(G)$ може бути отримана наступним чином:

$$\gamma(\mathrm{G}) \geq \frac{\mathrm{n}^2}{\mathrm{n}^2 - 2\mathrm{m}}.$$

Верхня оцінка хроматичного числа може бути обчислена за формулою:

$$\gamma(G) \leq 1 + \max_{x_i \in X} \{d(x_i) + 1\}.$$

Застосування оцінок для хроматичного числа значно звужує межі рішення. Для визначення оцінки хроматичного числа також можуть використовуватися інші топологічні характеристики графа, наприклад, властивість планарності.

Граф, який можна зобразити на площині так, що жодні два його ребра не перетинаються між собою, називається *планарним.*

Теорема про п'ять фарб. Кожний планарний граф можна розфарбувати за допомогою п'яти кольорів так, щоб будь-які дві суміжні вершини були пофарбовані в різні кольори, тобто якщо граф G – планарний, то $\gamma(G) \leq 5$.

Гіпотеза про чотири фарби (недоведена). Кожний планарний граф можна розфарбувати за допомогою чотирьох кольорів так, що будь-які дві суміжні вершини будуть пофарбовані в різні кольори, тобто $\gamma(G) \leq 4$, якщо граф G - планарний.

У 1852 р. про гіпотезу чотирьох фарб говорилося в листуванні Огюста де Моргана з сером Вільямом Гамільтоном. З того часу ця «теорема» стала, поряд з теоремою Ферма, однією з найзнаменитіших невирішених завдань в математиці.

Повний граф G_n завжди розфарбовується в n кольорів, рівних кількості його вершин.

Евристичні алгоритми розфарбування

Точні методи розфарбування графа складні для програмної реалізації. Проте існує багато евристичних процедур розфарбування, які дозволяють знаходити гарні наближення для визначення хроматичного числа графа. Такі процедури також можуть з успіхом використовуватися при розфарбуванні графів з великою кількістю вершин, де застосування точних методів не виправдане з огляду на високу трудомісткість обчислень.

З евристичних процедур розфарбування слід зазначити послідовні методи, засновані на впорядкуванні множини вершин.

В одному з найпростіших методів вершини спочатку розташовуються в порядку спадання їх степенів. Перша вершина фарбується в колір 1; потім список вершин проглядається за спаданням степенів і в колір 1 забарвлюється кожна вершина, яка не є суміжною з вершинами, пофарбованими в той же колір. Потім повертаємося до першої в списку непофарбованої вершини, фарбуємо її в колір 2 і знову переглядаємо список вершин зверху вниз, фарбуючи в колір 2 будь-яку непофарбовану вершину, яка не з'єднана ребром з іншою, вже пофарбованою в колір 2 вершиною. Аналогічно діємо з кольорами 3, 4 і т.д., поки не будуть пофарбовані всі вершини. Число використаних кольорів буде тоді наближеним значенням хроматичного числа графа.

Евристичний алгоритм розфарбування вершин графа має наступний вигляд:

Крок 1. Сортувати вершини графа за спаданням степеней:

$$\deg(x_i) \geq \deg(x_j),\ \forall x_i, x_j \in G$$

Встановити поточний колір p=1; i=1.

Крок 2. Вибрати чергову НЕ розфарбовану вершину зі списку і призначити їй новий колір: $col(x_i)=p$; $Xp=\{x_i\}$.

Крок 3. $i=i+1$. Вибрати чергову непофарбовану вершину x_i і перевірити умову суміжності:

$$x_i \cap \Gamma(Xp)=\varnothing,$$

де Xp – множина вершин, вже пофарбованих у колір p. Якщо вершина x_i не суміжна з даними вершинами, то їй також присвоїти колір p: $col(x_i)=p$.

Крок 4. Повторювати крок 3, поки не буде досягнутий кінець списку ($i=n$).

Крок 5. Якщо всі вершини графа розфарбовані, то – кінець алгоритму.

Інакше: $p=p+1$; $i=1$. Повторити крок 2.

Порядок виконання роботи

1. Індивідуальні завдання для виконання лабораторної роботи «Розфарбування графа» представлені в Додатку А (див. табл. 2).

2. Скласти алгоритми розфарбування вершин та ребер заданого неорієнтованого графу.

3. Створити програму, яка буде розфарбовувати граф (вершини або ребра в залежності від вибраної операції) та виводити результат на екран.

4. Інтерфейс повинен включати:

a) вказівку на автора роботи (ім’я у заголовку, вікно з повідомленням, тощо);
b) поле введення матриці суміжності графа;
c) поле із графічно зображеним графом;
d) поле для текстового відображення ейлеревого циклу;
e) кнопку для пошуку мінімального розфарбування вершин графу;

f) кнопку для пошуку мінімального розфарбування ребер графу;
g) кнопку для скидання введених у матрицю значень.

Приклад виконання роботи

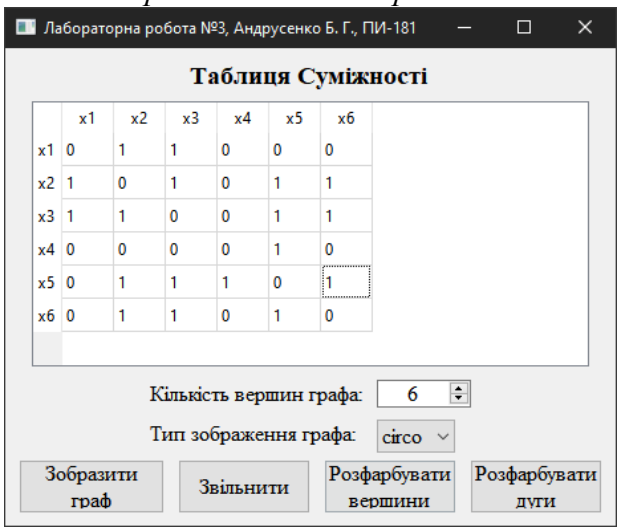


Рис 3.2. Загальний вигляд інтерфейсу

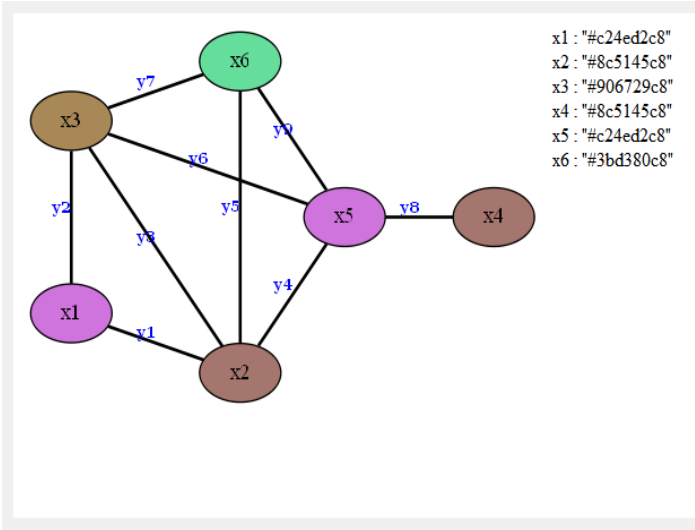


Рис 3.3. Вікно з розфарбованими вершинами

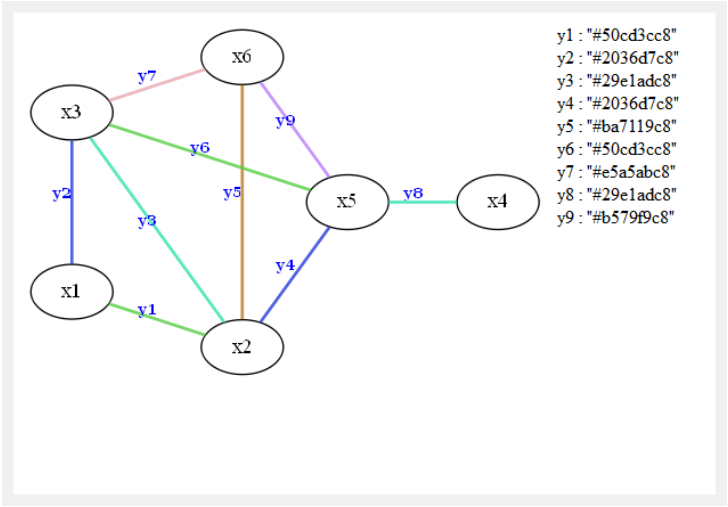


Рис 3.4. Вікно з розфарбованими ребрами

Зміст звіту

1. Назва і тема лабораторної роботи.

2. Блок-схема алгоритму розфарбування вершин.

3. Блок-схема алгоритму розфарбування ребер.

4. Зображення інтерфейсу програми.

5. Результати виконання роботи на контрольних прикладах.

6. Висновки.

Контрольні питання

1. Сформулюйте задачу розфарбування графа.
2. Який граф називається r-хроматичним?
3. Що таке хроматичне число графа?
4. Як визначаються верхня та нижня оцінки хроматичного числа графа?
5. Що таке планарний граф? У скільки кольорів його можна розфарбувати?
6. У скільки кольорів можна розфарбувати повний граф?
7. Евристичний алгоритм розфарбування графа.
9. Реалізація евристичного алгоритму для знаходження хроматичного числа.

ЛАБОРАТОРНА РОБОТА №4
Побудова найкоротших остов дерев графа

Мета роботи. Метою роботи є вивчення методу побудови найкоротшого каркасу дерева графа на прикладі алгоритму Прима-Краскала.

Короткі теоретичні відомості

Одним з найбільш важливих понять теорії графів є *дерево***.**

Неорієнтованим деревом називають зв'язний граф, що не має циклів.

Суграфом графа G називають підграф G_p, що містить всі вершини вихідного графа.

Якщо $G=(X,A)$ – неорієнтований граф з n вершинами, то зв'язний суграф G_p, який не має циклів, називається *остовом або каркасом графа G.*

Для остового дерева справедливе відношення:

$$G_p=(X_p,A_p) \subseteq G,$$

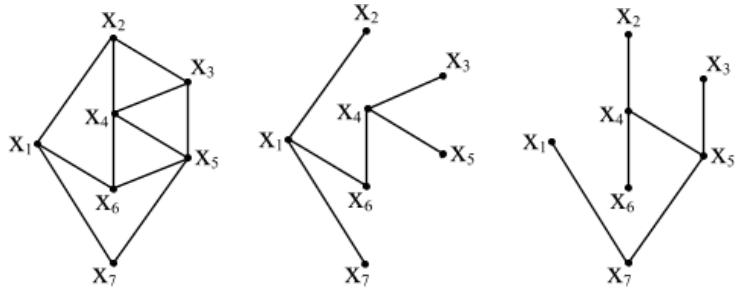
де $X_p = X$, $A_p \subseteq A$.

Досить легко довести, що остове дерево має наступні властивості:

1) остове дерево графа з n вершинами *має n-1 ребер* ($|X_p|=|A_p|-1$);

2) існує єдиний шлях, що з'єднує будь-які дві вершини остова графа: $\forall x_i,x_j \in X_p\ (i \neq j) \rightarrow \exists!\ \mu(x_i,x_j)$.

Наприклад, якщо G – це граф, що показаний на рисунку 4.1(а), то графи на рисунках 4.1(б) і 4.1(в) є остовами графа G. Зі сформульованих вище означень випливає, що остов графа G можна розглядати як мінімальний зв'язний остовий підграф графа G.



(а) (б) (в)

Рисунок 4.1. – Граф і його остови (каркаси)

Поняття дерева як математичного об'єкта було вперше запропоновано Кірхгофом у зв'язку з визначенням фундаментальних циклів, що застосовуються при аналізі електричних ланцюгів. Приблизно десятьма роками пізніше Келі знову (незалежно від Кірхгофа) ввів поняття дерева та отримав більшу частину перших результатів в області дослідження властивостей дерев. Найбільш відомою є його знаменита теорема:

Теорема Келі: На графі з n вершинами можна побудувати n^{n-2} остових дерев.

Орієнтоване дерево являє собою орієнтований граф без циклів, в якому півстепені «входу» кожної вершини, за винятком однієї (вершини r), дорівнює одиниці, а півстепень «заходу» вершини r (названою коренем цього дерева) дорівнює нулю.

На рисунку 4.2 зображено граф, що є орієнтованим деревом з коренем у вершині x_1. З наведеного визначення випливає, що орієнтоване дерево з n вершинами має $(n-1)$ дуг та є зв'язним.

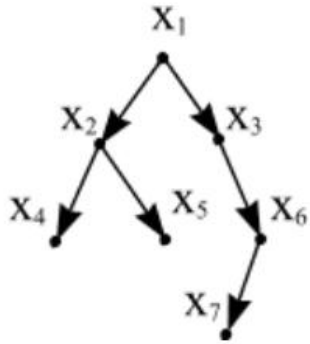


Рисунок 4.2 – Орієнтоване дерево

Неорієнтоване дерево можна перетворити в орієнтоване: необхідно взяти його довільну вершину в якості кореня та приписати ребрам таку орієнтацію, щоб кожна вершина з'єднувалася з коренем (тільки одним) простим ланцюгом.

«Генеалогічне дерево», в якому вершини відповідають особам чоловічої статі, а дуги орієнтовані від батьків до дітей, представляють добре відомий приклад орієнтованого дерева. Корінь в цьому дереві відповідає «засновнику» роду, тобто особі, яка народилася раніше за інших.

Мінімальний остов графа

У даній лабораторній роботі досліджується метод прямої побудови найкоротших остовів дерев у зваженому графі, тобто в якому ваги приписані дугам. Найкоротше остове дерево графа знаходить своє застосування при прокладанні доріг (газопроводів, ліній електропередач і т.п.), коли необхідно зв'язати *n* точок деякою мережею так, щоб загальна довжина «ліній зв'язку» була мінімальною. Якщо точки лежать в евклідовій площині, то їх можна

вважати вершинами повного графа G з вагами дуг, рівними відповідним «прямолінійним» відстаням між кінцевими точками дуг. Якщо «розгалуження» доріг допускається тільки в заданих n точках, мінімальне остове дерево графа G буде саме тією необхідною мережею доріг, що має найменшу вагу.

Розглянемо зважений зв'язний неорієнтований граф $G=(X,A)$. Вагу ребра (x_i,x_j) позначимо c_{ij}. З великого числа остовів графа потрібно знайти один, у якого сума ваг ребер найменша. Така задача виникає, наприклад, в тому випадку, коли вершини представляють міста, які потрібно зв'язати мережею трубопроводів; тоді найменша загальна довжина труб, яка повинна бути використана для будівництва (за умови, що за містами «розгалуження» трубопроводів не допускаються), визначається найкоротшим остовом відповідного графа.

Завдання побудови найкоротшого каркасу графа є однією з небагатьох задач теорії графів, які можна вважати повністю вирішеними.

Алгоритм Прима-Краскала

Даний алгоритм утворює остове дерево засобами розростання тільки одного піддерева, наприклад X_p, що містить більше однієї вершини. Піддерево поступово розростається за рахунок приєднання ребер (x_i,x_j), де $x_i \in X_p$ і $x_j \notin X_p$; до того ж ребро, що додається, повинно мати найменшу вагу c_{ij}. Процес продовжується до того моменту, поки число ребер в A_p не буде дорівнювати $(n-1)$. Тоді піддерево $G_p=(X_p,A_p)$ буде шуканим остовим деревом. Вперше така операція була запропонована Примом і Краскалом (з різницею у способі побудови дерева), тому даний алгоритм і отримав назву Прима-Краскала.

Алгоритм починає свою роботу з включення у піддерево початкової вершини. Оскільки остове дерево включає всі вершини графа G, то вибір початкової вершини не має принципового значення. Будемо кожній черговій вершині привласнювати помітку $\beta(x_i)=1$, якщо вершина x_i належить піддереву X_p і $\beta(x_j)=0$ у протилежному випадку.

Алгоритм має наступний вигляд:

Крок 1. Нехай $X_p=\{x_1\}$, де x_1 – початкова вершина, і $A_p=\varnothing$ (A_p є множиною ребер, що входять до остового дерева). Вершині x_1 привласнити помітку $\beta(x_1)=1$. Для кожної вершини $x_i \notin X_p$ привласнити $\beta(x_i)=0$.

Крок 2. З усіх вершин $x_j \in \Gamma(X_p)$, для яких $\beta(x_j)=0$, знайти вершину x_j^* таку, що $c(x_i, x_j^*) = \min\limits_{x_j \in \Gamma(X_p)} \{c(x_i, x_j)\}$, де $x_i \in X_p$ і $x_j \notin X_p$.

Крок 3. Оновити дані: $X_p = X_p \cup \{x_j^*\}$; $A_p = A_p \cup (x_i, x_j^*)$. Привласнити $(x_j^*)=1$.

Крок 4. Якщо $|X_p|=n$, то необхідно зупинитися. Ребра в A_p утворюють мінімальний остов графа. Якщо $|X_p|<n$, то перейти до *кроку 2.*

Контрольний приклад

В якості прикладу розглянемо граф, що зображено на рисунку 4.3.

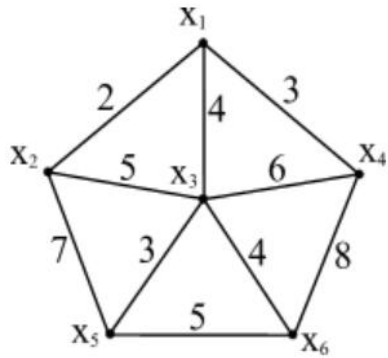


Рисунок 4.3 – Початковий граф

Знайдемо для нього мінімальне остове дерево, використовуючи для цього розглянутий вище алгоритм Прима-Краскала. Позначимо множину суміжних вершин, що не входять у породжене дерево, як $\Gamma^*(X_p)$. Таким чином, для всіх вершин, що входять до даної множини, оцінка $\beta(x_j)=0$, $\forall x_j \in \Gamma^*(X_p)$. Вектор B являє собою множину оцінок $\beta(x_i)$ для всіх вершин графа G: $\forall x_i \in X$. Довжина породженого піддерева позначається як L.

Ітерація 1:

$X_p=\{x_1\}$;

$A_p=\varnothing$;

$\Gamma^*(X_p)=\{x_2, x_3, x_4\}$;

$c(x_1, x_2^*)=2$;

$B=\{1, 1, 0, 0, 0, 0\}$;

$L=2$.

Ітерація 2:

$X_p=\{x_1, x_2\}$;

$A_p=\{(x_1, x_2)\}$;

$\Gamma^*(X_p)=\{x_3, x_4, x_5\}$;

$c(x_1, x_4^*)=3$;

$B=\{1, 1, 0, 1, 0, 0\}$;

L=*2+3=5*.

Ітерація 3:

$X_p=\{x_1, x_2, x_4\}$;

$A_p=\{(x_1, x_2); (x_1, x_4)\}$;

$\Gamma^*(X_p)=\{x_3, x_5, x_6\}$;

$c(x_1, x_3^*)=4$;

$B=\{1, 1, 1, 1, 0, 0\}$;

L=*5+4*.

Ітерація 4:

$X_p=\{x_1, x_2, x_3, x_4\}$;

$A_p=\{(x_1, x_2);(x_1, x_4);(x_1, x_3)\}$;

$\Gamma^*(X_p)=\{x_5, x_6\}$;

$c(x_3, x_5^*)=2$;

$B=\{1, 1, 1, 1, 1, 0\}$;

L=*9+3=12*.

Ітерація 5:

$X_p=\{x_1, x_2, x_3, x_4, x_5\}$;

$A_p=\{(x_1, x_2); (x_1, x_4); (x_1,x_3); (x_3,x_5)\}$;

$\Gamma^*(X_p)=\{x_6\}$;

$c(x_3, x_6^*)=4$;

$B=\{1, 1, 1, 1, 1, 1\}$;

L=*12+4=16*.

Задача розв'язана. Отримані множини вершин X_p і ребер A_p утворюють мінімальне остове дерево:

$X_p=\{x_1, x_2, x_3, x_4, x_5, x_6\}$;

$A_p=\{(x_1, x_2); (x_1, x_4); (x_1,x_3); (x_3,x_5); (x_3,x_6)\}$.

Загальна довжина мінімального остового дерева *L=16.*

Результат розв'язання задачі представлено на рисунку 4.4.

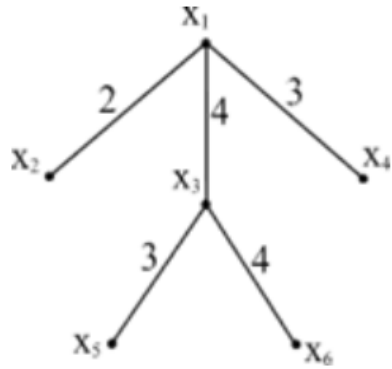


Рисунок 4.4 – Найкоротше остове дерево

Порядок виконання роботи

1. Індивідуальні завдання для виконання лабораторної роботи «Побудова мінімальних остових дерев графа» представлені в Додатку А (таблиця 3).

2. Побудувати блок-схему програми, що визначає мінімальне остове дерево графа за допомогою алгоритму Прима-Краскала.

3. Розробити програму, яка реалізує алгоритм Прима-Краскала. Вихідний граф задається у вигляді матриці суміжності, що вводиться за допомогою консолі. Програма повинна вивести список ребер, що входять до мінімального остового дерева, а також передбачити побудову графу.

4. Інтерфейсна частина повинна містити:

a) заголовок із зазначенням назви роботи та інформації про автора;
b) місце для задання графа у вигляді матриці суміжності;
c) кнопку для побудови остового дерева.

Приклад виконання роботи

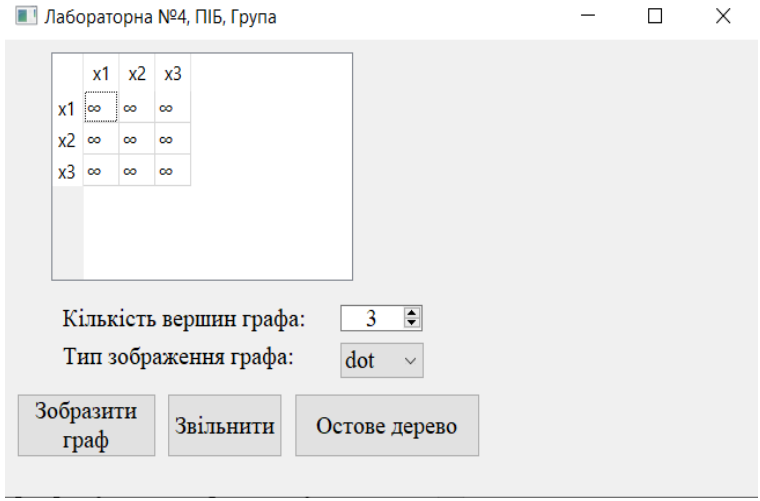


Рисунок 4.5. Головне вікно програми

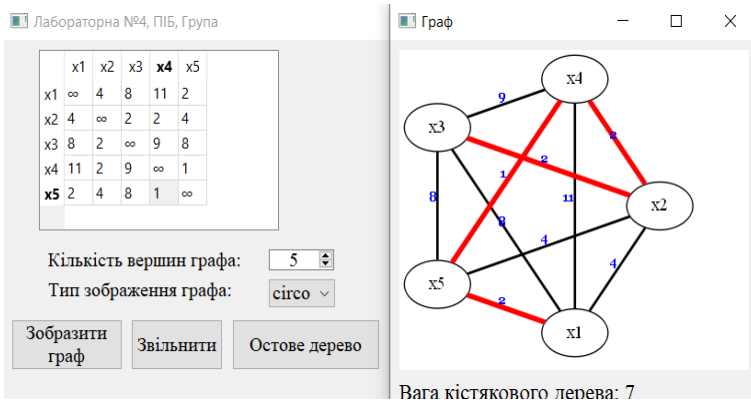


Рисунок 4.6. Побудова остового дерева

Зміст звіту по роботі

1. Вихідне завдання та ціль роботи.
2. Блок-схема програми алгоритму Прима-Краскала.
3. Зображення інтерфейсу програми.
4. Контрольний приклад і результати машинного розрахунку.

5. Висновки стосовно роботи.

Контрольні питання

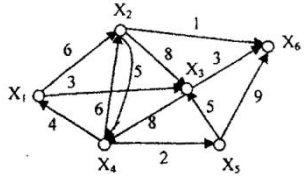
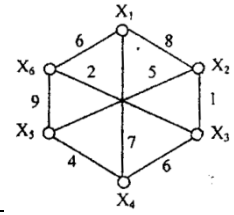
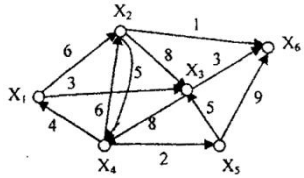
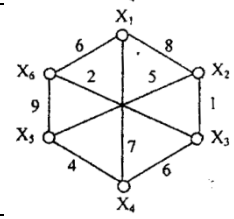
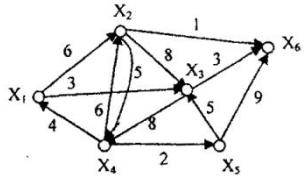
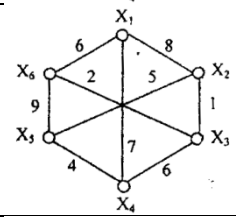
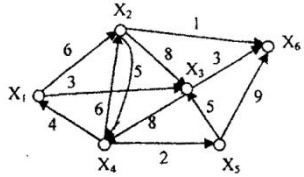
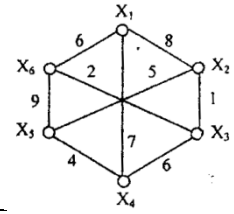
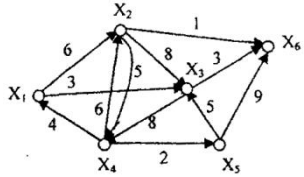
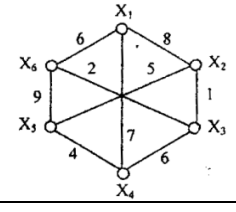
1. Дайте визначення дерева та орієнтованого дерева.
2. Яке дерево називають остовим?
3. Властивості остових дерев. Теорема Келі.
4. Що називають коренем дерева?
5. Як перетворити неорієнтоване дерево в орієнтоване?
6. Скільки ребер містить остове дерево графа?
7. Задача знаходження мінімального остова графа.
8. Наведіть практичні приклади знаходження мінімального остова графа.
9. Реалізація алгоритму Прима-Краскала для знаходження мінімального остова графа.

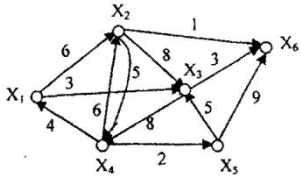
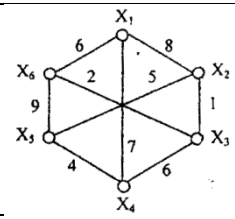
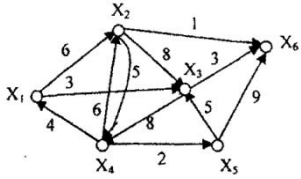
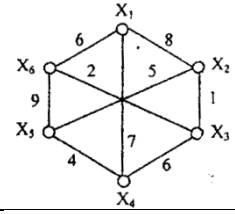
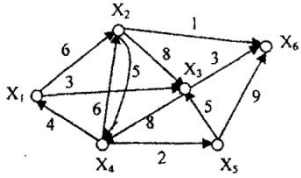
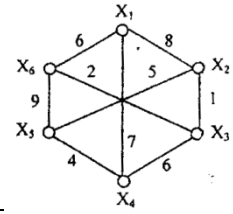
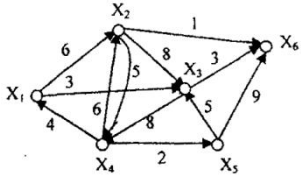
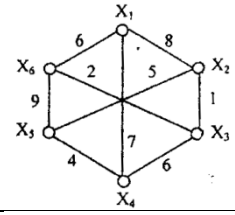
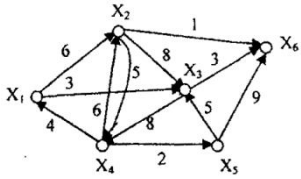
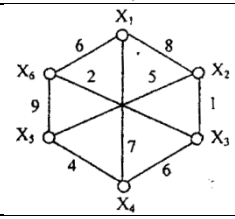
Список використаних джерел

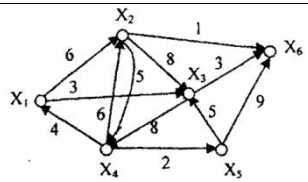
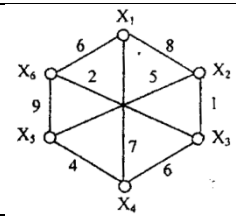
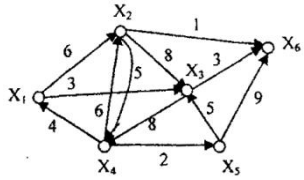
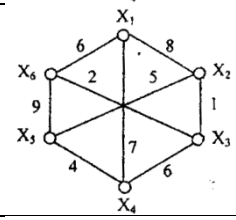
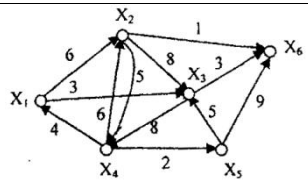
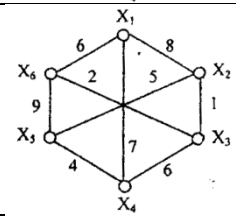
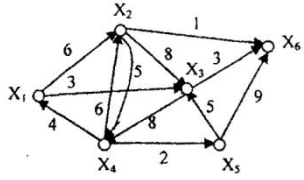
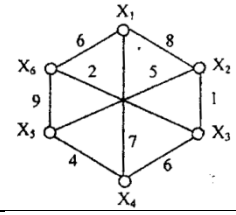
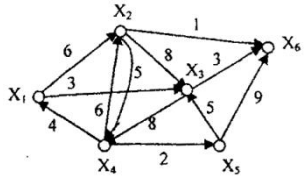
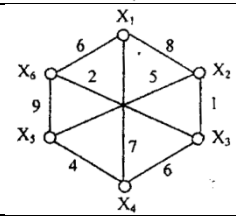
1. Базилевич Л.Є. Дискретна математика у прикладах і задачах: теорія множин, математична логіка, комбінаторика, теорія графів. – Математичний практикум. – Львів, 2013. – 486 с.
2. Бондарчук Ю. В., Олійник Б. В. Основи дискретної математики. – Київ : Видавничий дім «Києво-Могилянська Академія», 2009. – 160 с.
3. Бондарчук Ю.В., Олійник Б. В. Основи дискретної математики. – Київ: Видавничий дім «Києво-Могилянська Академія», 2009. – 160 с.
4. Дискретна математика: Навч. посіб. для студ. ВНЗ / Р.М. Трохимчук. – К. : Вид. дім «Професіонал», 2010. – 528 с.
5. Дискретна математика: підручник / Ю.В. Нікольський, В.В. Пасічник, Ю.М. Щербина; за наук. ред. В.В. Пасічника; М-во освіти і науки. молоді та спорту України. – 3-тє вид., виправл. та доповн. – Львів: Магнолія-2006, 2013. – 432 с.
6. Дрозд Ю.*А.* Дискретна математика. – Київ: Київський Національний університет імені Тараса Шевченка, 2004. – 71 с.
7. Комп'ютерна дискретна математика: Підручник / Бондаренко М.Ф., Білоус Н.В., Руткас А.Г. – Харків: «Компанія СМІТ», 2004. – 480 с.
8. Харари Ф. Теория графов / пер. с англ. предисл. В.П. Козырева. Под ред. Г.П. Гаврилова. Изд. 2-е. – М: Едиториал УРСС, 2003. – 296 с.
9. Ямненко *Р.Є.* Дискретна математика: навчально-методичний посібник. – Київ: Четверта хвиля, 2010. – 105 с.

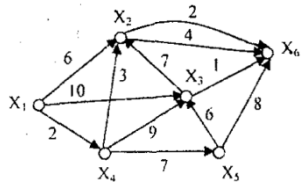
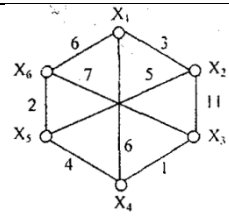
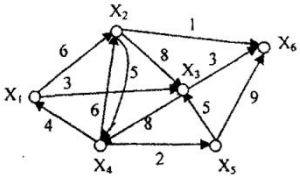
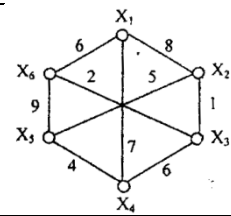
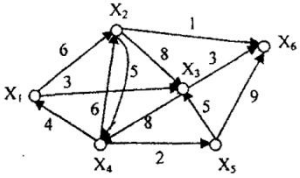
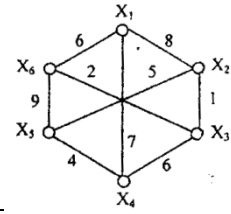
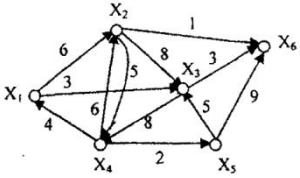
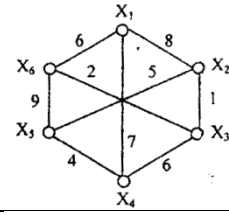
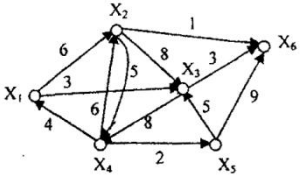
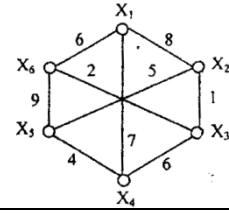
Додаток А

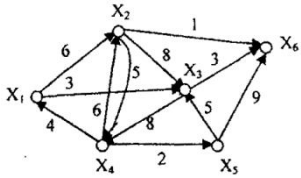
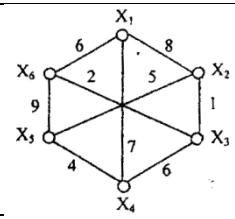
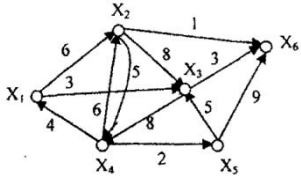
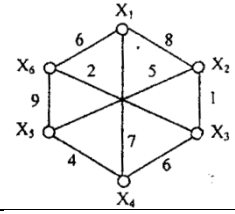
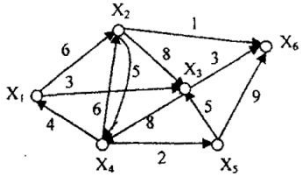
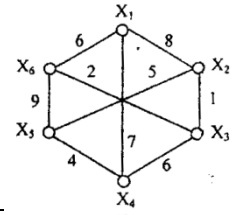
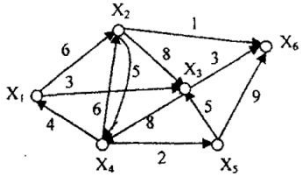
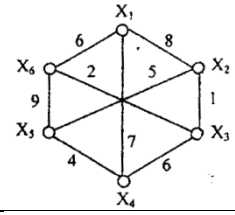
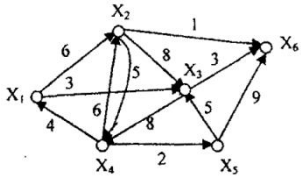
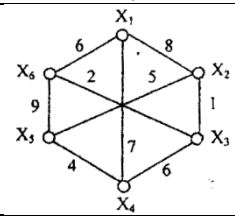
Таблиця 1

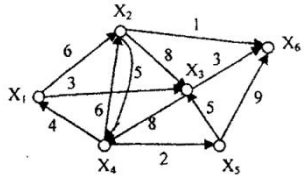
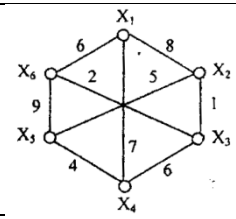
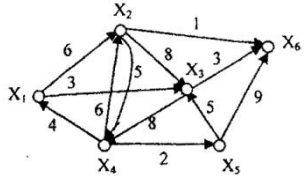
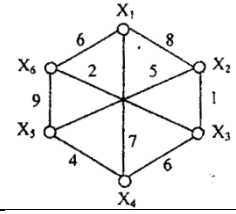
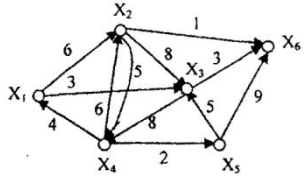
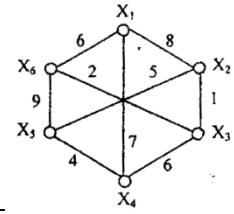
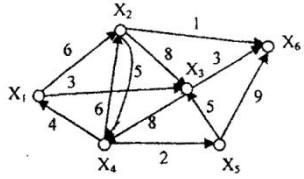
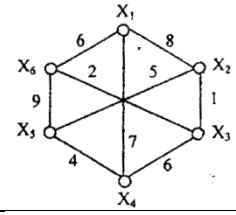
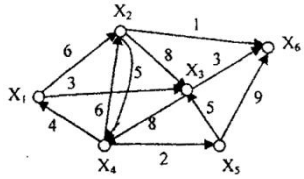
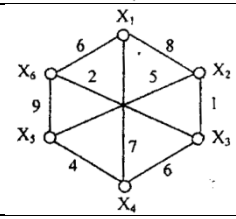
№ вар.	Матриця суміжності	Матриця інцидентності
1		
2		
3		
4		
5		

6		
7		
8		
9		
10		

11		
12		
13		
14		
15		

16		
17		
18		
19		
20		

21		
22		
23		
24		
25		

26		
27		
28		
29		
30		

Таблиця 2

Варіант	m	1	1	1	1	1	2	2	2	2	3	3	3	4	4	5
	n	2	3	4	5	6	3	4	5	6	4	5	6	5	6	6
1	R_{mn}	1	2	1	1	1	0	0	0	1	1	0	0	1	1	1
2	R_{mn}	0	0	1	1	1	0	1	0	1	0	1	0	0	0	2
3	R_{mn}	1	0	0	1	0	0	1	1	1	0	0	0	1	0	1
4	R_{mn}	1	0	1	0	0	1	1	0	2	0	0	1	0	2	0
5	R_{mn}	2	1	1	0	0	1	1	0	1	0	0	0	1	1	0
6	R_{mn}	1	0	0	1	0	1	0	0	0	1	0	0	1	2	2
7	R_{mn}	1	1	1	0	0	0	2	0	2	0	0	1	0	1	0
8	R_{mn}	0	2	0	2	0	1	0	0	0	0	1	0	1	1	1
9	R_{mn}	1	2	1	0	0	0	0	0	1	1	1	0	1	0	0
10	R_{mn}	1	1	1	1	0	0	0	0	1	0	1	0	1	1	1
11	R_{mn}	1	2	0	0	1	1	1	0	0	2	1	0	1	0	0
12	R_{mn}	0	0	0	1	1	1	0	0	2	0	0	1	1	0	2
13	R_{mn}	0	1	0	1	0	0	1	2	0	1	0	0	0	1	1
14	R_{mn}	1	1	0	0	0	1	0	1	1	0	1	1	1	0	1
15	R_{mn}	1	0	2	0	2	0	0	0	1	0	0	0	1	0	1
16	R_{mn}	1	1	0	0	0	1	1	0	1	0	0	0	1	1	2
17	R_{mn}	1	0	0	0	0	1	1	2	0	0	0	1	1	0	1
18	R_{mn}	2	1	0	1	0	1	1	0	0	0	1	0	1	1	1
19	R_{mn}	1	1	0	0	0	2	1	0	2	1	0	0	1	1	1
20	R_{mn}	2	0	1	1	0	0	1	0	1	0	0	0	0	2	1
21	R_{mn}	1	1	0	0	0	1	1	0	1	1	0	1	1	1	0
22	R_{mn}	2	0	0	0	0	1	1	1	1	0	1	0	1	0	0
23	R_{mn}	1	1	1	1	0	1	1	0	1	0	0	0	1	1	0

24	R_{mn}	0	0	1	0	1	2	1	0	1	1	0	1	0	1	0
25	R_{mn}	2	0	1	0	1	1	0	0	1	0	0	1	1	0	1
26	R_{mn}	0	2	1	0	1	0	0	1	1	0	0	0	0	1	1
27	R_{mn}	0	1	0	1	0	1	1	0	0	1	1	0	1	1	1
28	R_{mn}	2	0	1	1	0	1	1	0	0	0	1	0	1	1	1
29	R_{mn}	1	0	1	0	2	1	1	1	0	0	0	1	1	1	0
30	R_{mn}	1	1	0	0	0	2	0	1	2	1	0	0	1	0	0

Позначення:

- m – номер початкової вершини.
- n – номер кінцевої вершини.
- R_{mn} – кількість ребер між вершинами m і n.

Таблиця 3

1	$\begin{pmatrix} \infty & 12 & 6 & 20 & 14 \\ 12 & \infty & 2 & 4 & 6 \\ 6 & 2 & \infty & 10 & 12 \\ 20 & 4 & 10 & \infty & 6 \\ 14 & 6 & 12 & 6 & \infty \end{pmatrix}$	**16**	$\begin{pmatrix} \infty & 1 & \infty & 4 & 5 \\ 1 & \infty & 2 & \infty & 1 \\ \infty & 2 & \infty & 1 & 1 \\ 4 & \infty & 1 & \infty & 3 \\ 5 & 1 & 1 & 3 & \infty \end{pmatrix}$
2	$\begin{pmatrix} \infty & 6 & 3 & 10 & 7 \\ 6 & \infty & 1 & 2 & 3 \\ 3 & 1 & \infty & 5 & 6 \\ 10 & 2 & 5 & \infty & 3 \\ 7 & 3 & 6 & 3 & \infty \end{pmatrix}$	**17**	$\begin{pmatrix} \infty & 7 & 2 & 11 & 7 \\ 7 & \infty & 3 & \infty & 4 \\ 2 & 3 & \infty & 1 & 5 \\ 11 & \infty & 1 & \infty & 3 \\ 7 & 4 & 5 & 3 & \infty \end{pmatrix}$
3	$\begin{pmatrix} \infty & 2 & \infty & 5 & 5 \\ 2 & \infty & 8 & \infty & 7 \\ \infty & 8 & \infty & 10 & 1 \\ 5 & \infty & 10 & \infty & 13 \\ 5 & 7 & 1 & 13 & \infty \end{pmatrix}$	**18**	$\begin{pmatrix} \infty & 1 & 5 & 4 & 5 \\ 1 & \infty & 2 & 6 & 1 \\ 5 & 2 & \infty & 1 & 7 \\ 4 & 6 & 1 & \infty & 4 \\ 5 & 1 & 7 & 4 & \infty \end{pmatrix}$
4	$\begin{pmatrix} \infty & 4 & 3 & 5 & 6 \\ 4 & \infty & 2 & \infty & 1 \\ 3 & 2 & \infty & 1 & 1 \\ 5 & \infty & 1 & \infty & 3 \\ 6 & 1 & 1 & 3 & \infty \end{pmatrix}$	**19**	$\begin{pmatrix} \infty & 1 & 3 & 4 & 5 \\ 1 & \infty & 2 & 9 & 1 \\ 3 & 2 & \infty & 1 & 1 \\ 4 & 9 & 1 & \infty & 3 \\ 5 & 1 & 1 & 3 & \infty \end{pmatrix}$

5	$\begin{pmatrix} \infty & 5 & 3 & 10 & 7 \\ 5 & \infty & 1 & 2 & 4 \\ 3 & 1 & \infty & 5 & 6 \\ 10 & 2 & 5 & \infty & 3 \\ 7 & 4 & 6 & 3 & \infty \end{pmatrix}$	**20**	$\begin{pmatrix} \infty & 7 & 2 & 11 & 7 \\ 7 & \infty & 3 & \infty & 4 \\ 2 & 3 & \infty & 1 & 5 \\ 11 & \infty & 1 & \infty & 3 \\ 7 & 4 & 5 & 3 & \infty \end{pmatrix}$
6	$\begin{pmatrix} \infty & 1 & \infty & 4 & 5 \\ 1 & \infty & 8 & \infty & 7 \\ \infty & 8 & \infty & 10 & 1 \\ 4 & \infty & 10 & \infty & 13 \\ 5 & 7 & 1 & 13 & \infty \end{pmatrix}$	**21**	$\begin{pmatrix} \infty & 1 & 5 & 4 & 6 \\ 1 & \infty & 2 & 6 & 3 \\ 5 & 2 & \infty & 1 & 7 \\ 4 & 6 & 1 & \infty & 4 \\ 6 & 3 & 7 & 4 & \infty \end{pmatrix}$
7	$\begin{pmatrix} \infty & 12 & 6 & 10 & 4 \\ 12 & \infty & 2 & 5 & 6 \\ 6 & 2 & \infty & 10 & 12 \\ 10 & 5 & 10 & \infty & 6 \\ 4 & 6 & 12 & 6 & \infty \end{pmatrix}$	**22**	$\begin{pmatrix} \infty & 3 & 2 & 4 & 5 \\ 3 & \infty & 2 & \infty & 1 \\ 2 & 2 & \infty & 1 & 1 \\ 4 & \infty & 1 & \infty & 3 \\ 5 & 1 & 1 & 3 & \infty \end{pmatrix}$
8	$\begin{pmatrix} \infty & 12 & 6 & 10 & 4 \\ 12 & \infty & 2 & 5 & 6 \\ 6 & 2 & \infty & 10 & 12 \\ 10 & 5 & 10 & \infty & 6 \\ 4 & 6 & 12 & 6 & \infty \end{pmatrix}$	**23**	$\begin{pmatrix} \infty & 3 & 2 & 4 & 5 \\ 3 & \infty & 2 & \infty & 1 \\ 2 & 2 & \infty & 1 & 1 \\ 4 & \infty & 1 & \infty & 3 \\ 5 & 1 & 1 & 3 & \infty \end{pmatrix}$
9	$\begin{pmatrix} \infty & 12 & 6 & 20 & 14 \\ 12 & \infty & 2 & 4 & 6 \\ 6 & 2 & \infty & 10 & 12 \\ 20 & 4 & 10 & \infty & 6 \\ 14 & 6 & 12 & 6 & \infty \end{pmatrix}$	**24**	$\begin{pmatrix} \infty & 1 & \infty & 4 & 5 \\ 1 & \infty & 2 & \infty & 1 \\ \infty & 2 & \infty & 1 & 1 \\ 4 & \infty & 1 & \infty & 3 \\ 5 & 1 & 1 & 3 & \infty \end{pmatrix}$

10	$\begin{pmatrix} \infty & 6 & 3 & 10 & 7 \\ 6 & \infty & 1 & 2 & 3 \\ 3 & 1 & \infty & 5 & 6 \\ 10 & 2 & 5 & \infty & 3 \\ 7 & 3 & 6 & 3 & \infty \end{pmatrix}$	**25**	$\begin{pmatrix} \infty & 7 & 2 & 11 & 7 \\ 7 & \infty & 3 & \infty & 4 \\ 2 & 3 & \infty & 1 & 5 \\ 11 & \infty & 1 & \infty & 3 \\ 7 & 4 & 5 & 3 & \infty \end{pmatrix}$
11	$\begin{pmatrix} \infty & 2 & \infty & 5 & 5 \\ 2 & \infty & 8 & \infty & 7 \\ \infty & 8 & \infty & 10 & 1 \\ 5 & \infty & 10 & \infty & 13 \\ 5 & 7 & 1 & 13 & \infty \end{pmatrix}$	**26**	$\begin{pmatrix} \infty & 1 & 5 & 4 & 5 \\ 1 & \infty & 2 & 6 & 1 \\ 5 & 2 & \infty & 1 & 7 \\ 4 & 6 & 1 & \infty & 4 \\ 5 & 1 & 7 & 4 & \infty \end{pmatrix}$
12	$\begin{pmatrix} \infty & 4 & 3 & 5 & 6 \\ 4 & \infty & 2 & \infty & 1 \\ 3 & 2 & \infty & 1 & 1 \\ 5 & \infty & 1 & \infty & 3 \\ 6 & 1 & 1 & 3 & \infty \end{pmatrix}$	**27**	$\begin{pmatrix} \infty & 1 & 3 & 4 & 5 \\ 1 & \infty & 2 & 9 & 1 \\ 3 & 2 & \infty & 1 & 1 \\ 4 & 9 & 1 & \infty & 3 \\ 5 & 1 & 1 & 3 & \infty \end{pmatrix}$
13	$\begin{pmatrix} \infty & 5 & 3 & 10 & 7 \\ 5 & \infty & 1 & 2 & 4 \\ 3 & 1 & \infty & 5 & 6 \\ 10 & 2 & 5 & \infty & 3 \\ 7 & 4 & 6 & 3 & \infty \end{pmatrix}$	**28**	$\begin{pmatrix} \infty & 7 & 2 & 11 & 7 \\ 7 & \infty & 3 & \infty & 4 \\ 2 & 3 & \infty & 1 & 5 \\ 11 & \infty & 1 & \infty & 3 \\ 7 & 4 & 5 & 3 & \infty \end{pmatrix}$

14	$\begin{pmatrix} \infty & 2 & \infty & 4 & 5 \\ 1 & \infty & 8 & \infty & 7 \\ \infty & 8 & \infty & 10 & 1 \\ 4 & \infty & 10 & \infty & 13 \\ 5 & 7 & 1 & 13 & \infty \end{pmatrix}$	**29**	$\begin{pmatrix} \infty & 1 & 5 & 4 & 6 \\ 1 & \infty & 2 & 6 & 3 \\ 5 & 2 & \infty & 1 & 7 \\ 4 & 6 & 1 & \infty & 4 \\ 6 & 3 & 7 & 4 & \infty \end{pmatrix}$
15	$\begin{pmatrix} \infty & 12 & 6 & 10 & 4 \\ 12 & \infty & 2 & 5 & 6 \\ 6 & 2 & \infty & 10 & 12 \\ 10 & 5 & 1 & \infty & 6 \\ 4 & 6 & 12 & 6 & \infty \end{pmatrix}$	**30**	$\begin{pmatrix} \infty & 1 & 2 & 4 & 5 \\ 3 & \infty & 2 & \infty & 1 \\ 2 & 2 & \infty & 1 & 1 \\ 4 & \infty & 1 & \infty & 3 \\ 5 & 1 & 1 & 3 & \infty \end{pmatrix}$