МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ УКРАИНЫ
ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ ИНФОРМАТИКИ И ИСКУССТВЕННОГО ИНТЕЛЛЕКТА

УЧЕБНО-МЕТОДИЧЕСКОЕ ПОСОБИЕ

ПО ДИСЦИПЛИНЕ

«ТЕОРИЯ ГРАФОВ»

Донецк 2010

МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ УКРАИНЫ

ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ ИНФОРМАТИКИ И ИСКУССТВЕННОГО ИНТЕЛЛЕКТА

УЧЕБНО-МЕТОДИЧЕСКОЕ ПОСОБИЕ
ПО ДИСЦИПЛИНЕ
«**ТЕОРИЯ ГРАФОВ**»

для студентов специальностей
«Программное обеспечение автоматизированных систем»,
«Интеллектуальные системы принятия решений»
дневной формы обучения

Утверждено:
на заседании кафедры программного обеспечения интеллектуальных систем
(протокол № 9 от 21.05.2010г.)

на заседании Ученого совета ГУИиИИ
(протокол № 9 от 31.05.2010г.)

Донецк-2010

Учебно-методическое пособие по дисциплине «Теория графов» для студентов специальностей «Программное обеспечение автоматизированных систем» и «Интеллектуальные системы принятия решений» дневной формы обучения /Сост.: Е.В. Бычкова, Т.В. Ермоленко, Донецк: ГУИиИИ, 2010.- 108 с.

Изложены теоретические основы по следующим разделам теории графов: подграфы, изоморфизм, маршруты и связность, деревья и остовы, основы цикломатики, орграфы. Содержатся рекомендации по изучению теоретического материала, контрольные вопросы, рекомендуемая литература, задания для лабораторных работ и примеры их выполнения.

Составители:
ст. преп. Е.В. Бычкова,
асс. Т.В. Ермоленко

Рецензент: с.н.с., к.ф.-м.н., И.С. Грунский

1 ВВЕДЕНИЕ В ТЕОРИЮ НЕОРИЕНТИРОВАННЫХ ГРАФОВ

1.1 СВЕДЕНИЯ ИЗ ИСТОРИИ

1736 г. считается годом возникновения **теории графов**: **Л.Эйлер** опубликовал результаты решения задачи о Кенигсбергских мостах и сформулировал критерий существования в графах специального маршрута (эйлерова цикла). Более чем 100 лет эти результаты оставались единственными и лишь **в середине XIX в.** инженер-электрик **Г. Кирхгоф** разработал **теорию деревьев** для исследования электрических цепей. Математик **А. Кэли** решил перечислительные задачи для определенных типов деревьев при исследовании строения углеводородов. К этому же периоду относится появление знаменитой проблемы 4 красок.

За последние десятилетия теория графов превратилась в один из некоторых бурно развивающихся разделов математики. В теоретико-графовых терминах формулируется большое число задач, связанных с дискретными объектами.

Термин "граф" был введен в 1936 г. венгерским математиком Д. Кенигом.

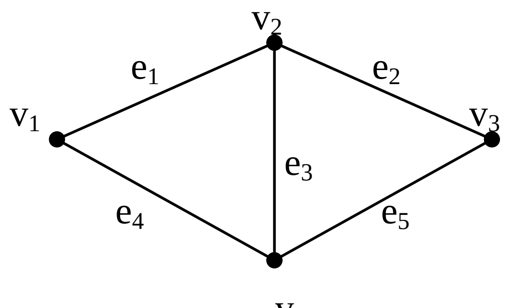
1.2 ОСНОВНЫЕ ОПРЕДЕЛЕНИЯ

Пусть имеется некоторое множество $V \neq \varnothing$ и пусть $V^{(2)}$ – неупорядоченное множество всех его двухэлементных подмножеств ($V^{(2)} = \{(u,v)$: $u,v \in V$, неупорядоченная пара$\}$). Тогда **неориентированным графом** G называется пара множеств (V,E), где $E \subseteq V^{(2)}$. Обозначается: G=(V,E).

Множество V называется **множеством вершин графа**, а множество E – **множеством ребер графа**.

Число |V| вершин графа G называется его **порядком**. Если $|V| = p$, а $|E| = q$, то граф G=(V,E) называют **p-графом,** или **(p,q)-графом**, или $G_{p,q}$.

Пример: 4-граф, или (4,5)-граф, или $G_{4,5}$.



Две **вершины** графа называются **смежными**, если они соединены ребром.

Два **ребра** графа называются **смежными**, если они имеют общую вершину.

Обозначим ребро графа: e=(u,v), где u и v –концевые вершины ребра. Ребро e **инцидентно** вершине v, если вершина v является одной из концевых вершин ребра e.

Заметим, что **смежность** есть отношение между однородными элементами графа, тогда как **инцидентность** является отношением между разнородными элементами.

Если множество V конечно, то граф называют **конечным**. Далее будем говорить только о конечных графах.

1.3 СПОСОБЫ ЗАДАНИЯ ГРАФОВ

Существуют следующие основные способы задания графов.

1. **Перечисление** множеств V (вершин) и E (ребер), задающих граф.G=(V, E).

2. **Графический**: множество вершин V – это множество точек плоскости, множество ребер E – множество отрезков прямой(кривой) на плоскости.

3. **Матричные** способы описания.

   Пусть G=(V,E), $|V| = p$, а $|E| = q$, тогда:

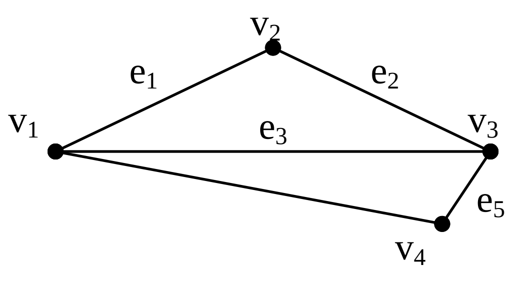
   a) матрица смежности – квадратная матрица $A = \|a_{ij}\|, \quad i, j = \overline{1, p}$, где

$$a_{ij} = \begin{cases} 1, \exists(i, j) \in E & \text{(вершины i и j смежны)}, \\ 0, \text{иначе;} \end{cases}$$

   b) матрица инцидентности – прямоугольная матрица $B = \|b_{ij}\|, \quad i = \overline{1, p}$, $j = \overline{1, q}$, где

$$b_{ij} = \begin{cases} 1, \text{если вершина i инцидентна ребру j}, \\ 0, \text{иначе.} \end{cases}$$

Пример: Задан граф G=(V, E), где V={v_1,v_2, v_3, v_4}, E=**{**v_1v_2, v_2v_3, v_1v_3, v_1v_4, v_3v_4**}**={e_1, e_2, e_3, e_4, e_5}.



A – матрица смежности:

A	v_1	v_2	v_3	v_4
v_1	0	1	1	1
v_2	1	0	1	0
v_3	1	1	0	1
v_4	1	0	1	0

B – матрица инцидентности:

B	e_1	e_2	e_3	e_4	e_5
v_1	1	0	1	1	0
v_2	1	1	0	0	0
v_3	0	1	1	0	1
v_4	0	0	0	1	1

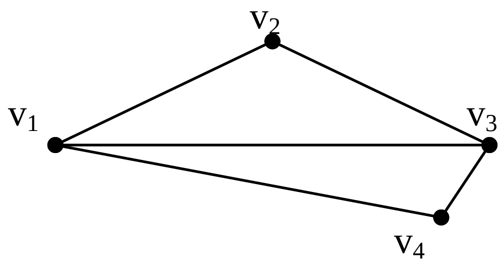
1.4 СТЕПЕНИ ВЕРШИН ГРАФА

Окружением вершины v называется множество всех вершин графа G, смежных с ней; обозначается: N(v).

Степенью или **валентностью** вершины v неориентированного графа G называется число ребер, инцидентных данной вершине (число вершин в ее окружении). Обозначается: deg(v) $(\deg(v) = \|(u,v) : u, v \in V, (u,v) \in E\|)$.

Число $\Delta G = \max \deg(v)$, $\forall v \in V$ называется **максимальной степенью** графа G, а число $\delta(G) = \min \deg(v)$, $\forall v \in V$ – **минимальной степенью** графа G.

Пример:



$\deg(v_1) = \deg(v_3) = 3$; $\deg(v_2) = \deg(v_4) = 2$.

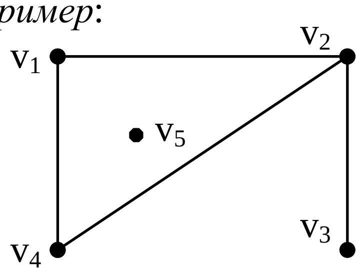
$\Delta G = 3$, $\delta(G) = 2$.

Вершина v графа G называется **изолированной**, если ее степень равна нулю (deg(v)=0).

Вершина v графа G называется **висячей** или **концевой,** если степень этой вершины равна единице (deg (v)=1).

Вершина v графа G(p,q) называется **доминирующей**, если ее степень равна p-1 (deg (v)=p-1).

Пример:



доминирующей вершины нет;
висячая – v_3;
изолированная – v_5.

Лемма о рукопожатии. Сумма степеней всех вершин графа четна и равна удвоенному числу ребер:

$$\sum_{i=1}^{p} \deg v_i = 2q = 2|E|.$$

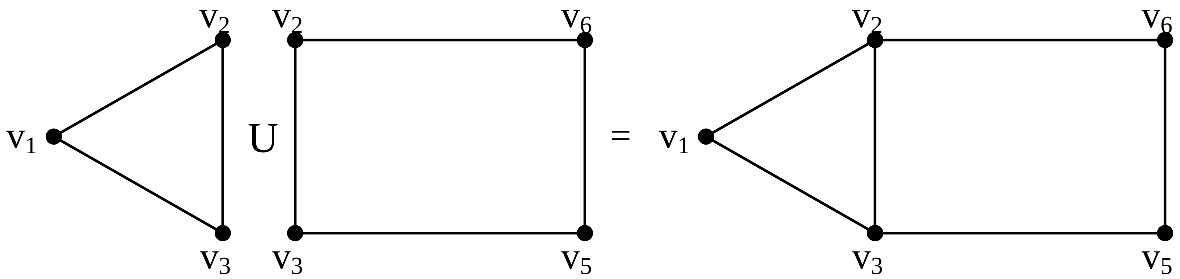
Следствие. В любом графе число вершин с нечетной степенью четно.

1.5 ОПЕРАЦИИ НАД ГРАФАМИ

Рассмотрим графы $G_1 = (V_1, E_1)$ и $G_2 = (V_2, E_2)$.

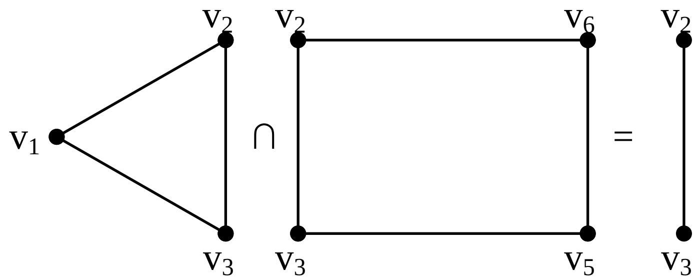
Объединением двух графов G_1 и G_2 называется граф G, множество вершин которого $V = V_1 \cup V_2$ и множество ребер $E = E_1 \cup E_2$.

Пример:



Пересечение двух графов G_1 и G_2 – это граф G, у которого множество вершин $V = V_1 \cap V_2$ и множество ребер $E = E_1 \cap E_2$.

Пример:

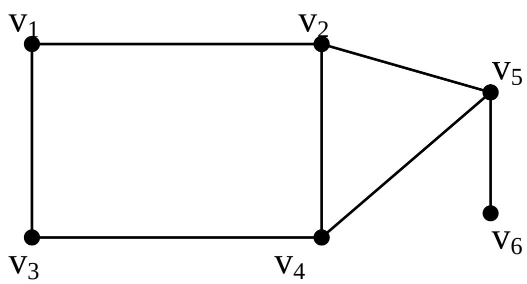


Непересекающимися графами называются графы G_1 и G_2 такие что $E_1 \cap E_2 = \varnothing$ и $V_1 \cap V_2 = \varnothing$ (нет общих ребер и нет общих вершин).

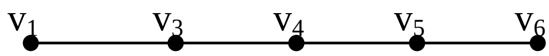
Пусть задан граф G=(V,E) и $v \in V$, тогда **удаление вершины** – это унарная операция над графом G, заключающаяся в построении порожденного подграфа G\{v} графа G на множестве вершин V\{v} (вершина удаляется из графа вместе с инцидентными ей ребрами).

Пример: v=v_2

G:



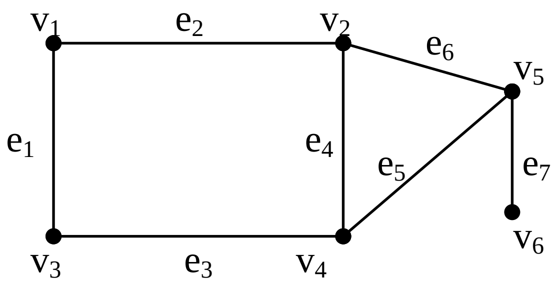
G\{v_2}:



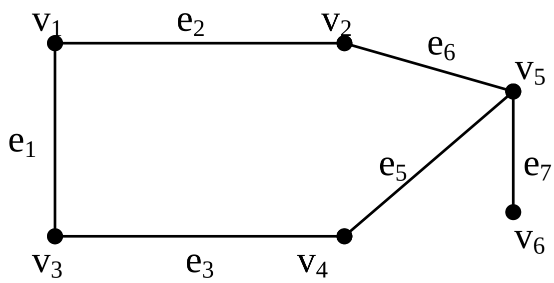
Пусть $e \in E$, тогда **удаление ребра** - это унарная операция над графом G, заключающаяся в построении порожденного подграфа G\{e} графа G с тем же множеством вершин и множеством ребер $E' = E \setminus \{e\}$.

Пример: e=e_4

G:

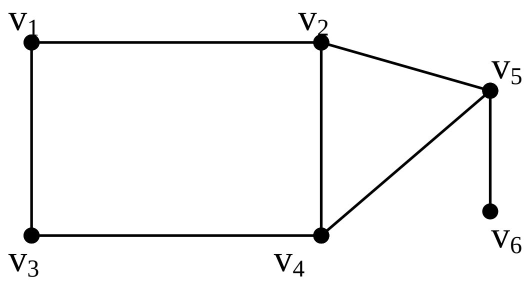


G\{e_4}:

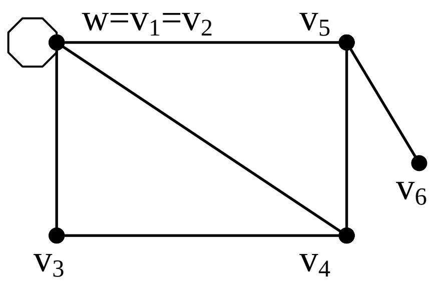


Говорят, что **пара вершин** v_i и v_j в графе G **замыкается (отождествляет-ся)**, если они заменяются такой новой вершиной, что все ребра графа G, инцидентные вершинам v_i и v_j, становятся инцидентными этой новой вершине.

Пример:

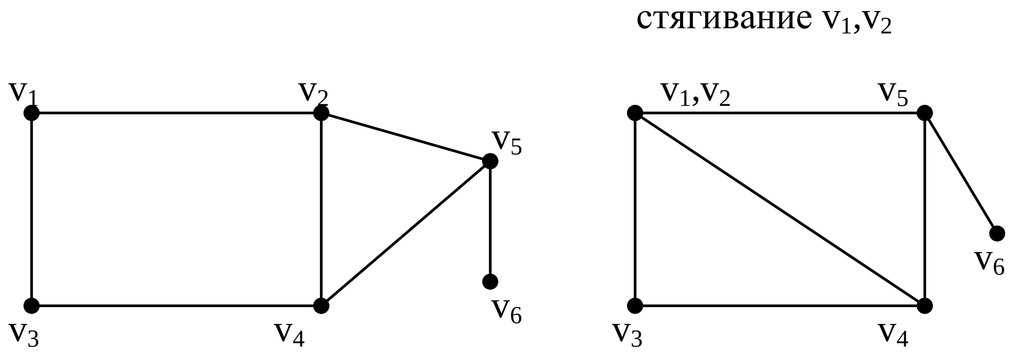


замыкание v_1,v_2



Стягивание заключается в удалении ребра e и отождествлении его концевых вершин.

Пример:



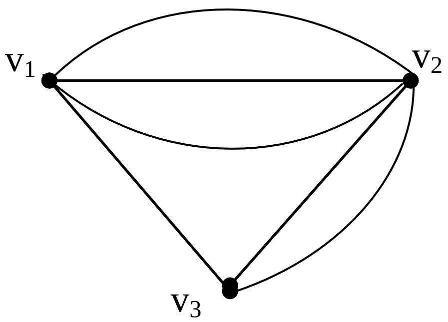
1.6 СПЕЦИАЛЬНЫЕ ГРАФЫ

В общем случае во множестве E допускается более чем одно ребро с одинаковыми концевыми вершинами. Такие ребра называются **параллельными,** или **кратными**.

Ребро, соединяющее вершину v саму с собой, называется **петлей**.

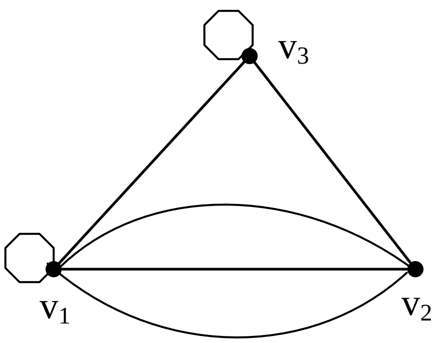
Граф, содержащий кратные ребра, но не содержащий петель, называется **мультиграфом**.

Пример:



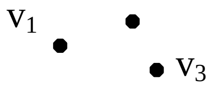
Граф, содержащий и петли, и кратные ребра, называется **псевдографом**.

Пример:



Граф, не содержащий ни одного ребра, называется **пустым графом** и обозначается O_n, где n – количество вершин графа.

Пример: O_3

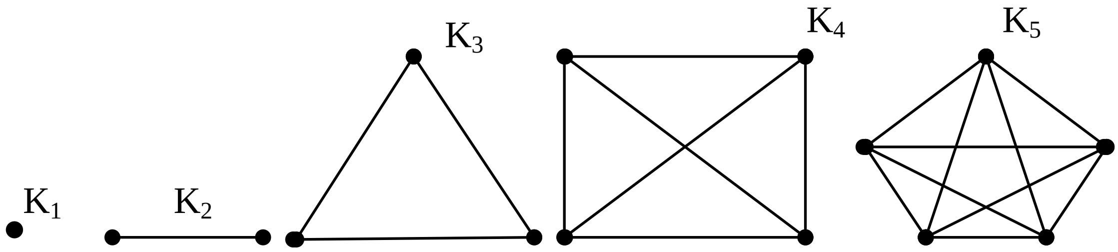


Граф, не содержащий ни одного ребра и ни одной вершины, называется **0-графом** (нуль-графом).

Тривиальный граф – это граф (1,0).

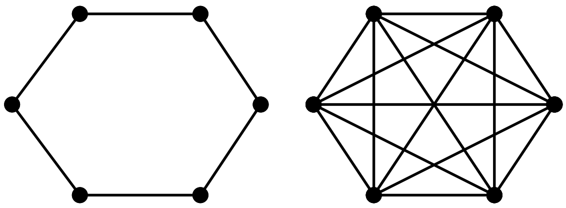
Граф, G называется **полным**, если у него все вершины смежные между собой. Каждая вершина полного графа является доминирующей. Обозначается: K_p, где p – количество вершин.

Пример:



Однородным, или **регулярным,** называется граф, у которого степени всех вершин равны. Все полные графы являются однородными.

Пример:

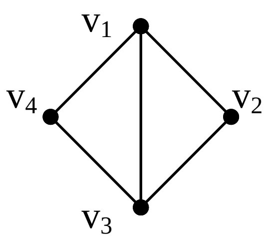


1.7 ПОДГРАФЫ

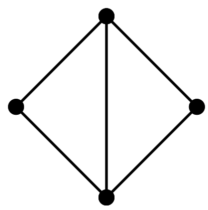
Неориентированный граф G=(V,E) называется **помеченным,** или **перенумерованным**, если каждой вершине графа поставлена в соответствие уникальная метка (число, символ); в противном случае граф называется **абстрактным**.

Пример:

помеченный граф



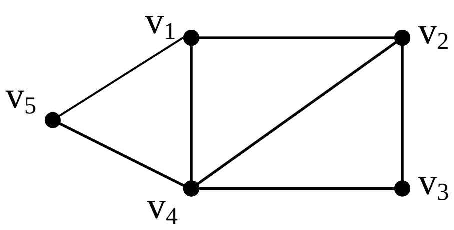
абстрактный граф



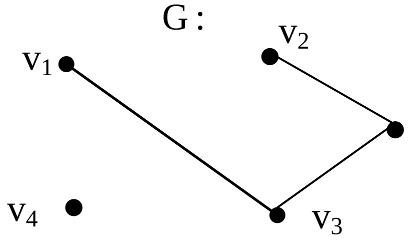
Граф $\overline{G} = (V', E')$ называется **дополнением** графа G, если множества вершин графов $\overline{G}$ и G совпадают, т. е. $V = V'$, а множество ребер $E' = V^{(2)} \setminus E$. Следовательно, любые две вершины, смежные в графе G, не смежны в его дополнении $\overline{G}$.

Пример:

G:



$\overline{G}$:

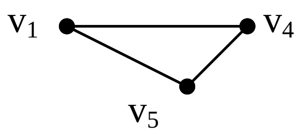


Подграфом графа G называется такой граф $G_1 = (V_1, E_1)$, у которого $V_1 \subseteq V$, $E_1 \subseteq E$.

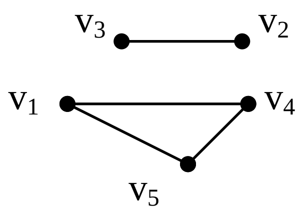
Пример: подграфы графа G

G_1:

G_2:



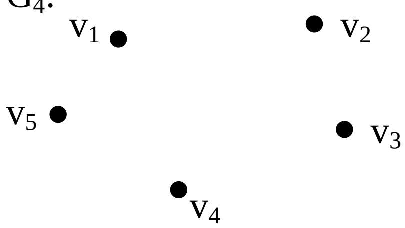
G_3:



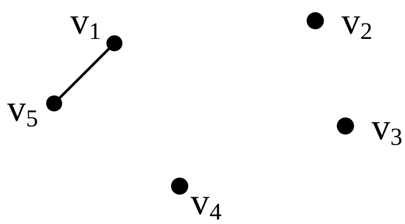
Остовным подграфом графа G называется такой подграф, у которого множество вершин равно множеству вершин исходного графа, а множество ребер может составлять часть множества ребер исходного графа ($V_1 = V$, $E_1 \subseteq E$).

Пример: остовные подграфы графа G

G_4:



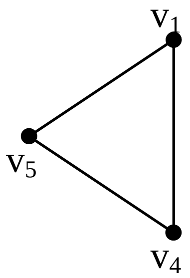
G_5:



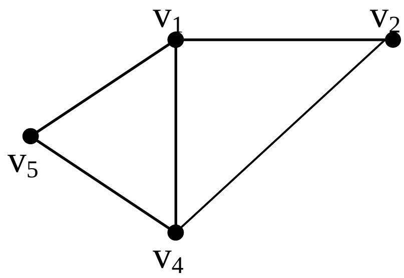
Порожденным подграфам графа G (подграфом, порожденным множеством вершин $V_1 \subseteq V$,) называется подграф $G_1 = (V_1, E_1)$ такой что содержит все ребра, инцидентные вершинам из множества V_1(т.е. те вершины, которые соединены в исходном графе ребром, будут соединены ребром и в порожденном подграфе).

Пример: подграфы графа G, порожденные вершинами

V_1={1,4,5}



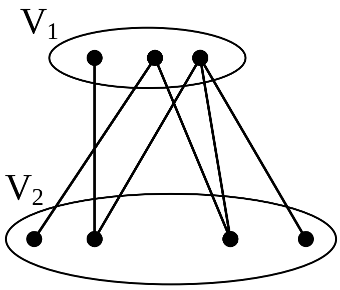
V_2={1,2,4,5}



1.8 ДВУДОЛЬНЫЕ ГРАФЫ

Граф, G=(V,E) называется **двудольным,** или **биграфом**, если множество его вершин V можно **разбить** на два подмножества V_1 и V_2, такие что любое ребро графа имеет одну концевую вершину во множестве V_1 и другую во множестве V_2. Таким образом, $V_1 \subseteq V$, $V_2 \subseteq V$, $V_1 \cap V_2 = \varnothing$, $V_1 \cup V_2 = V$.

Пример: двудольный граф

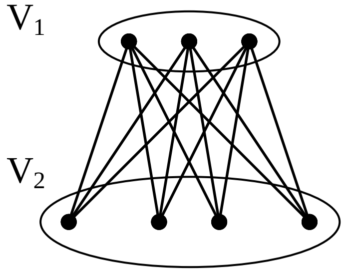


Если в двудольном графе G с разбиением (V_1,V_2) $\forall\, v_i \in V_1, \forall\, v_j \in V_2$, $\exists\,(v_i, v_j) \in E$, то такой двудольный граф называется **полным двудольным графом**.

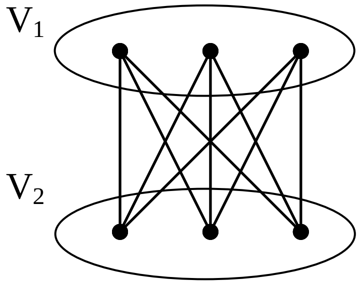
Обозначается: $K_{p,q}$, где $|V_1| = p$, $|V_2| = q$.

Пример:

$K_{3,4}$:



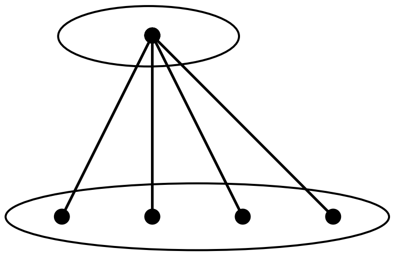
$K_{3,3}$:



Полный двудольный граф $K_{1,n}$ называется **звездой**.

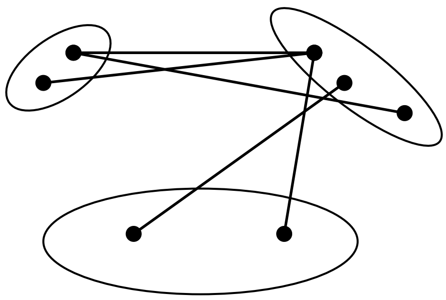
Пример:

$K_{1,4}$:



Граф G=(V,E) называется **k-дольным**, если множество вершин V можно разбить на k непересекающихся подмножеств таких что каждое ребро графа, имеющее концевую вершину $v_i \in V_i$, имеет другую концевую вершину $v_j \in V_j$ (i≠j), i,j=$\overline{1,k}$, $\bigcap_{i=1}^{k} V_i = \varnothing$, $\bigcup_{i=1}^{k} V_i = V$.

Пример:



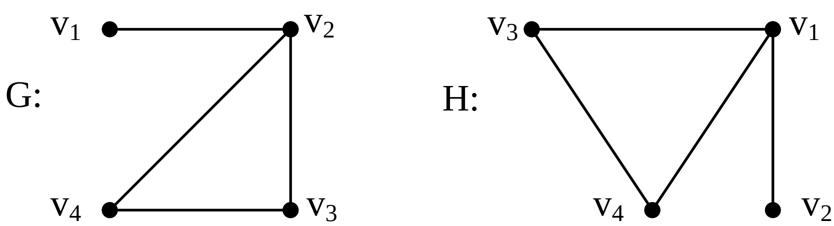
1.9 ИЗОМОРФИЗМ ГРАФОВ

Два графа G и H называют **изоморфными (**$G \cong H$), если существует взаимооднозначное соответствие (биекция) между множествами их вершин, сохраняющее отношение смежности.

Пусть G=(V,E), $H = (V_1, E_1)$. Тогда $G \cong H$, если

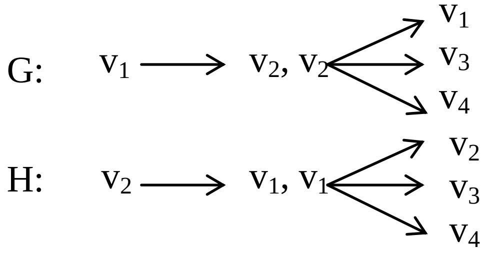
$\exists \varphi : V \rightarrow V_1 \quad \forall u, v \in V \Rightarrow \exists(\varphi(u), \varphi(v)) \in V_1$ и наоборот.

Пример:



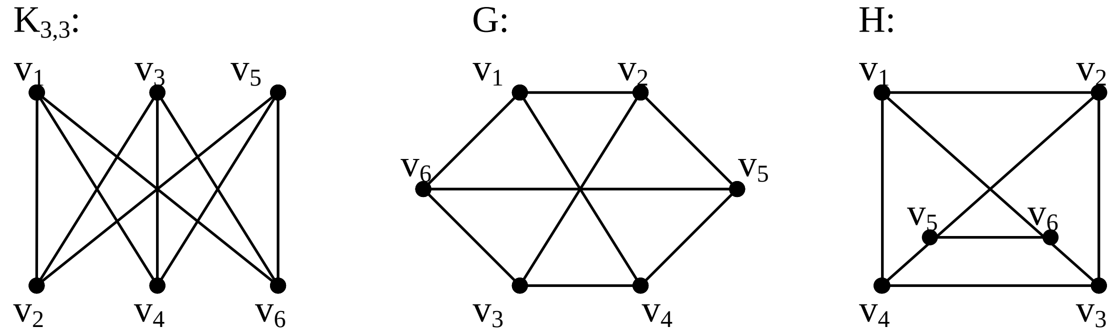
$G = (V, E), \quad H = (V', E')$

V	v_1	v_2	v_3	v_4
$\varphi(v)$	v_2	v_1	v_3	v_4



$G \cong H$

Пример изоморфных графов:

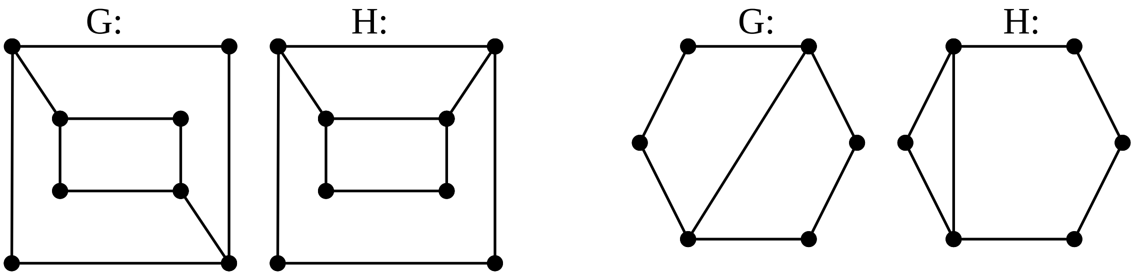


Условия изоморфизма графов:

- совпадение количества вершин и ребер;
- соответствие распределения степеней вершин;
- биекция, сохраняющая отношение смежности между вершинами графов.

Примеры неизоморфных графов:

а) б)



1.10 НЕЗАВИСИМОЕ МНОЖЕСТВО ВЕРШИН

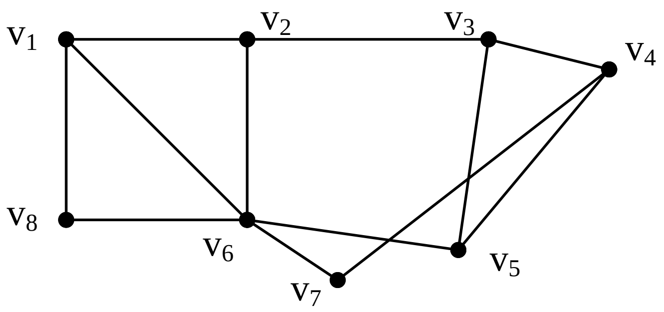
Независимым множеством вершин графа G, или **внутренним устойчивым** множеством, называется такое множество вершин S графа G, что любые две вершины этого множества не смежны.

Подграф, порожденный независимым множеством, является **пустым**.

Независимое множество называется **максимальным (Q)**, если оно не является собственным подмножеством другого независимого множества.

Независимое множество называется **наибольшим**, если оно наибольшее по мощности среди всех независимых множеств. Мощность наибольшего независимого множества называется **числом независимости**; обозначается: $\alpha(G)$.

Пример:



независимые множества:

$S_1=\{v_1,v_3\}$	$S_6=\{v_2,v_8\}$	$S_{11}=\{v_1,v_4\}$	$S_{16}=\{v_7,v_8\}$	$S_{21}=\{v_3,v_7,v_8\}$
$S_2=\{v_3,v_6\}$	$S_7=\{v_2,v_7\}$	$S_{12}=\{v_4,v_6\}$	$S_{17}=\{v_2,v_4,v_8\}$	$S_{22}=\{v_5,v_7,v_8\}$
$S_3=\{v_3,v_8\}$	$S_8=\{v_3,v_7\}$	$S_{13}=\{v_1,v_5\}$	$S_{18}=\{v_1,v_3,v_7\}$	$S_{23}=\{v_2,v_7,v_5\}$
$S_4=\{v_5,v_8\}$	$S_9=\{v_4,v_8\}$	$S_{14}=\{v_1,v_7\}$	$S_{19}=\{v_1,v_5,v_7\}$	$S_{24}=\{v_2,v_7,v_8\}$
$S_5=\{v_2,v_4\}$	$S_{10}=\{v_2,v_5\}$	$S_{15}=\{v_5,v_7\}$	$S_{20}=\{v_2,v_8,v_5\}$	$S_{25}=\{v_2,v_5,v_7,v_8\}$

$Q=\{S_2, S_{11}, S_{12}, S_{17}, S_{18}, S_{19}, S_{21,} S_{25}\}$, наибольшее – S_{25}, $\alpha(G)=4$.

1.11 КЛИКА

Кликой графа G называется такое подмножество вершин исходного графа, что любые две его вершины смежны.

Подграф, порожденный кликой, является полным.

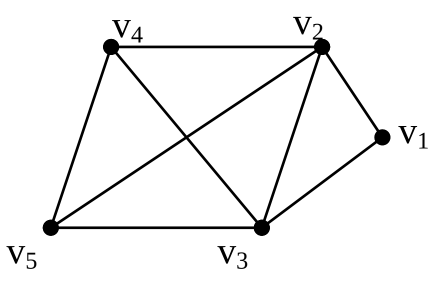
Максимальной кликой графа G называется клика, не являющаяся собственным подмножеством другой клики с большим числом вершин.

Клика графа G называется **наибольшей**, если число вершин в ней является максимальным среди всех других клик.

Мощность наибольшей клики называется **кликовым числом,** или **плотностью графа**; обозначается: $\varphi(G)$.

Пример:

G:



клики:
$S_1=\{v_1,v_2\}$, $S_2=\{v_1,v_3\}$, $S_3=\{v_2,v_3\}$,
$S_4=\{v_2,v_4\}$, $S_5=\{v_2,v_5\}$, $S_6=\{v_3,v_4\}$,
$S_7=\{v_3,v_5\}$, $S_8=\{v_4,v_5\}$, $S_9=\{v_1,v_2,v_3\}$,
$S_{10}=\{v_2,v_3,v_5\}$, $S_{11}=\{v_2,v_4,v_5\}$,
$S_{12}=\{v_3,v_4,v_5\}$,$S_{13}=\{v_2,v_3,v_4\}$,
$S_{14}=\{v_2,v_3,v_4,v_5\}$,
$Q=\{S_9, S_{14}\}$, наибольшая – S_{14}, $\varphi(G)=4$.

1.12 ДОМИНИРУЮЩИЕ МНОЖЕСТВА

Доминирующим, или **внешним устойчивым множеством** графа G, называется подмножество вершин исходного графа S такое что:

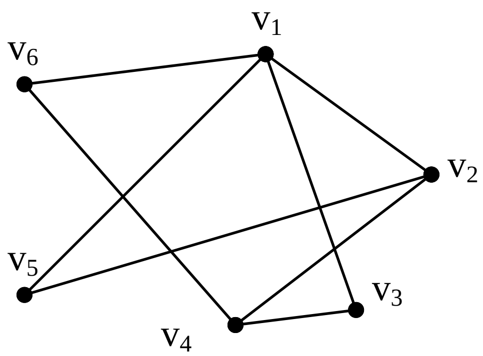
$S \subseteq V: \ \forall v \in V \setminus S \quad \exists \ w \in S$ такая что $\exists \ (v, w) \in E$.

Доминирующее множество называется **минимальным**, если нет другого доминирующего множества, содержащегося в нем.

Наименьшее доминирующее множество - доминирующее множество с наименьшей мощностью.

Мощность наименьшего доминирующего множества называется **числом доминирования**; обозначается: $\beta(G)$.

Пример:

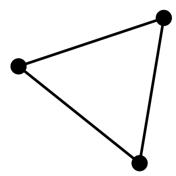


$S_1=\{v_3,v_5,v_6\}$,
$S_2=\{v_1,v_4,v_6\}$,
$S_3=\{v_1,v_4\}$,
$S_4=\{v_2,v_4\}$,
S_3, S_4 - минимальные, наименьшие множества,
$\beta(G) = 2$.

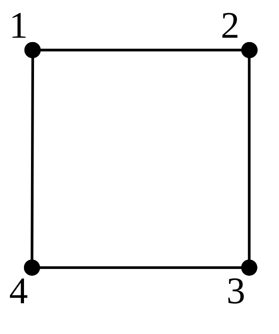
Ядром графа называется множество вершин графа, которое является одновременно и независимым, и доминирующим множеством.

Не каждый граф содержит ядро, например:

K_3:



Граф $K_{2,2}$ имеет два ядра:



$\{1, 4\}$,
$\{2, 3\}$

1.13 КОНТРОЛЬНЫЕ ВОПРОСЫ

1. Определение неорграфа, подграфа, остовного и порожденного подграфов.
2. Дополнение к графу.
3. Операции над графами.
4. Изоморфизм графов.
5. Помеченные и абстрактные графы.
6. Клика. Максимальная и наибольшая клика. Кликовое число, или плотность графа.
7. Независимое множество. Максимальное и наибольшее независимое множество. Число независимости.
8. Полный граф. Число ребер в полном графе.
9. Доминирующее множество. Минимальное и наименьшее доминирующие множество. Число доминирования.
10. Ядро графа.

2 МАРШРУТЫ И СВЯЗНОСТЬ НЕОРИЕНТИРОВАННЫХ ГРАФОВ

2.1 ОСНОВНЫЕ ОПРЕДЕЛЕНИЯ

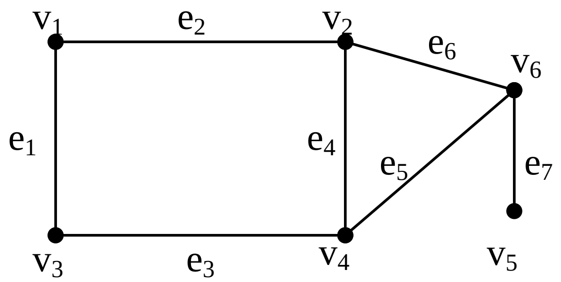
Пусть дан граф G=(V,E).

Маршрутом в графе G называется чередующаяся конечная последовательность вершин и ребер такая, что:

- начинается и заканчивается вершиной;
- каждое ребро соединяет две вершины (предыдущую и последующую), т.е. $(v_0,e_1,v_1, \ldots, e_n ,v_n)$, где $e_i= (v_{i-1},v_i)$, $i=\overline{1,n}$.

Обозначается: $M=(v_0,v_1, \ldots, v_n)$, или $(v_0 - v_n)$ – маршрут.

Пример:



$M= (v_1,e_2,v_2,e_6,v_6,e_5,v_4)= (v_1,v_2,v_6,v_4)$.

Вершины v_0 и v_n в маршруте M называются **конечными** или **терминальные**. Все остальные вершины маршрута называются **внутренними**.

Маршрут M называется **замкнутым**, если $v_0 = v_n$ (т.е. начальная и конечная вершины совпадают), в противном случае маршрут M называется **открытым**.

Маршрут называется **цепью**, если все его ребра различны, и **простой цепью**, если его ребра и вершины различны.

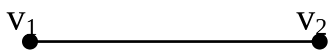
Замкнутая цепь называется **циклом** (замкнутый маршрут, в котором не повторяется ни одно ребро).

Простым циклом называется замкнутая простая цепь (замкнутый маршрут, в котором не повторяются ни ребра, ни вершины), причем количество вершин $n \geq 3$.

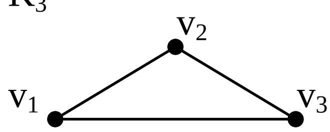
Пример:

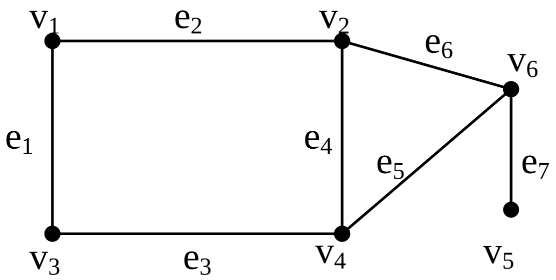
маршрут (v_1,v_2,v_1) – замкнутый,
но не цикл;

K_2



цикл:
K_3





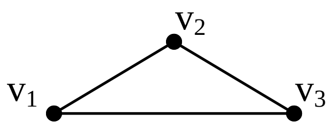
$M_1=(v_1,v_2,v_4,v_3,v_1,v_2,v_6,v_5)$ - маршрут;
$M_2=(v_1,v_3,v_4,v_2,v_6,v_4, v_3,v_1)$ - замкнутый маршрут, не цикл;
$M_3=(v_1,v_3,v_4,v_2,v_1)$ - простой цикл;
$M_4=(v_5,v_6,v_4,v_2, v_6)$ - цепь;
$M_5=(v_1,v_3,v_4,v_2)$ - простая цепь.

Граф, состоящий из одного простого цикла, обозначается C_n, где n – количество вершин в цикле.

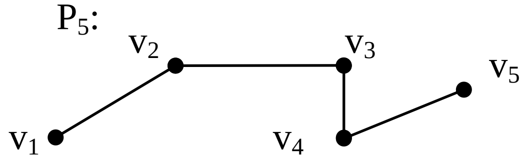
Граф, состоящий из одной простой цепи, обозначается P_n.

Пример:

C_3:



P_5:



2.2 СВЯЗНОСТЬ, КОМПОНЕНТЫ СВЯЗНОСТИ

Граф G называется **связным**, если любая пара его вершин соединена маршрутом.

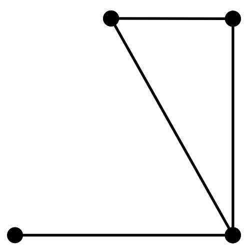
Две вершины v_i и v_j называются **связанными**, если в графе G существует маршрут между этими двумя вершинами. Вершина считается связанной сама с собой.

Максимальный связный подграф графа G называется **компонентой связности** графа G, или **компонентой**.

Связный граф состоит из одной компоненты связности.

Пример:

G_1:

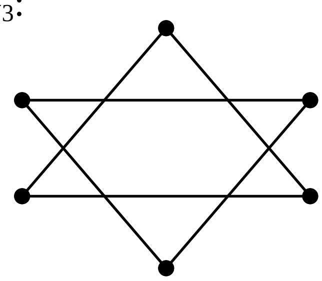


связный

G_2:

не связный
(4 компоненты)

G_3:



не связный
(2 компоненты)

Утверждение 1. Любой (u-v) – маршрут в неориентированном графе G содержит (u-v) – простую цепь.

Утверждение 2. Любой цикл неориентированного графа содержит в себе простой цикл.

В силу утверждений все определения связности можно давать в терминах простой цепи, например: две вершины называются **связанными**, если они соединены простой цепью.

Теорема о связности. Любой неориентированный граф является объединением своих компонент связности.

Теорема о связности дополнения. Любой неориентированный граф или его дополнение связны.

2.3 МЕТРИЧЕСКИЕ ХАРАКТЕРИСТИКИ ГРАФА

Длиной маршрута ($v_0, v_1, \ldots, v_n$) называется число ребер, содержащихся в маршруте, причем каждое ребро учитывается столько раз, сколько оно встречается в маршруте.

Пример:

Длина маршрута M=($v_0, v_1, \ldots, v_n$) равна n.

Любому ребру приписывается единичная длина, если граф не является взвешенным.

Взвешенным (нагруженным) называется граф, у которого каждому ребру поставлено в соответствии некоторое число, называемое **весом** или длиной ребра.

Достаточно задать матрицу весов:

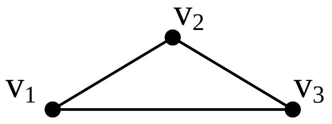
$$\mathrm{C} = \left\| \mathrm{C}_{ij} \right\| \qquad i, j = \overline{1, p}, \text{ где}$$

$$\mathrm{C}_{ij} = \begin{cases} \text{const, если } (i, j) \in \mathrm{E}; \\ 0, \text{ иначе.} \end{cases}$$

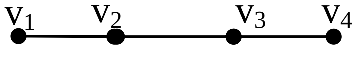
Обхватом графа G называется длина наименьшего простого цикла графа G, если таковой имеется; обозначается: g(G).

Окружением графа G называется длина наибольшего простого цикла графа G, если таковой имеется; обозначается: C(G).

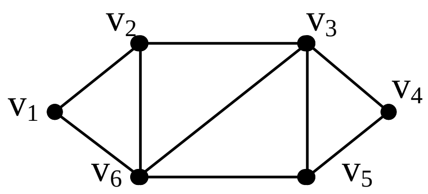
Пример:



g(G) = C(G) =3



g(G) = C(G) =0



g(G) = 3; C(G) =6

Расстоянием между двумя вершинами u и v графа G называется длина кратчайшей простой цепи, соединяющей эти вершины; обозначается: $d(u,v)$.

Если вершины u и v графа G не связаны, расстояние считается равным бесконечности: $d(u,v)=\infty$.

В связном графе расстояние является метрикой, т.е. удовлетворяет следующим аксиомам.

$\forall$ u,v,w $\in$V:

1) $d(u,v)\geq 0$ ($d(u,v)=0 \Leftrightarrow u=v$);
2) $d(u,v)=d(v,u)$;
3) $d(u,v)+d(v,w)= d(u,w)$.

Кратчайшая простая цепь, соединяющая вершины u и v, называется **геодезической** и обозначается: $\sigma(u,v)$.

<u>Алгоритм нахождения кратчайших маршрутов (волновой)</u>.

Пусть задан неориентированный граф G=(V,E). Необходимо найти расстояние от одной заданной вершины до другой, или от одной заданной ко всем остальным вершинам графа G.

Идея алгоритма:

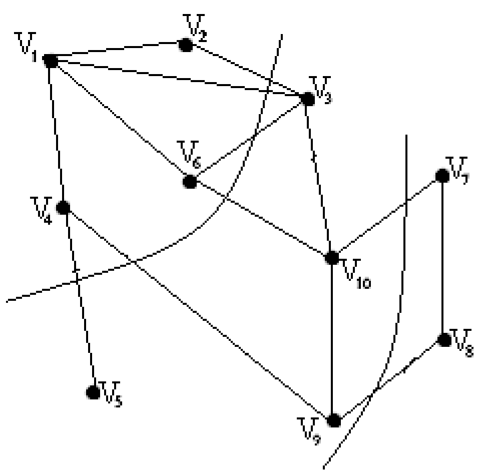
– выбирается некоторая вершина графа G и определяются все “соседи” выбранной вершины (“соседями” называются вершины, смежные с данной);
– эти вершины помечаются;
– определяются “соседи соседей” и также помечаются и т. д.

Алгоритм заканчивает работу, если:

– все вершины помечены (граф связен);
– построены все геодезические, связывающие данную вершину со всеми остальными;
– найдены расстояния от данной вершины ко всем остальным.

Если не все вершины в результате работы алгоритма помечены, это означает что граф G не связен.

Пример:



v_1	v_2	v_4	v_6	v_3	v_5	v_9	v_{10}	v_7	v_8
0	1	1	1	2	2	2	2	3	3

Эксцентриситетом, или **отклоненностью,** вершины v графа G называется длина максимальной геодезической, исходящей из этой вершины; обозначается: $e(v) = \max d(v, u)$, $v \in V$, $u \in V$.

Диаметром графа G называется максимальный среди всех эксцентриситетов вершин графа; обозначается: $d(G) = \max e(u)$, $\qquad u \in V$.

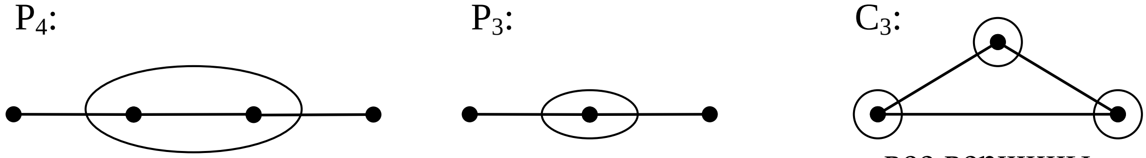
Периферией графа G называется множество вершин графа G, для которых эксцентриситет равен диаметру, т.е.: $e(v) = d(G)$.

Радиусом графа G называется минимальный из эксцентриситетов вершин графа G; обозначается: r(G)=min e(u), u∈V.

Центром графа G называется множество вершин, для которых эксцентриситет равен радиусу графа. т.е.: e(v) = r(G).

Примеры:

P_4: $\qquad$ P_3: $\qquad$ C_3:



центр $\qquad$ центр $\qquad$ все вершины - центральные

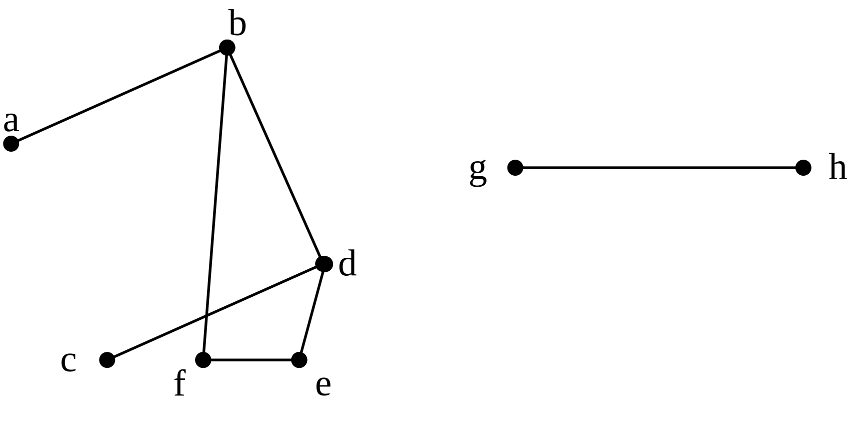
Определим **матрицу расстояний**:

$$D = \|d_{ij}\| \qquad i, j = \overline{1, p}, \quad \text{где p - количество вершин,}$$

$$d_{ij} = \begin{cases} d(i, j), & i \text{ и } j \text{ связны;} \\ 0, & i = j; \\ \infty, & i \text{ и } j \text{ не связны.} \end{cases}$$

Матрица расстояний несвязного графа может состоять из нескольких матриц (для каждой компоненты) или иметь блочный вид.

Пример: найти e(v) в заданном графе



Используем волновой алгоритм.

Первая компонента связности графа G:

v	j	u	σ(v,u)	d(u,v)
a	0	a	a	0
	1	b	ab	1
	2	d	abd	2
	2	f	abf	2
	3	c	abdc	3
		e	abfe	3

Вторая компонента связности графа G:

v	j	u	$\sigma(v,u)$	d(u,v)
g	0	g	g	0
	1	h	gh	1

Построим матрицу расстояний:

$$D =$$

	a	b	c	d	e	f	g	h	$e(v)$
a	0	1	3	2	3	2	∞	∞	3
b	1	0	2	1	2	1	∞	∞	2
c	3	2	0	1	2	3	∞	∞	3
d	2	1	1	0	1	2	∞	∞	2
e	3	2	2	1	0	1	∞	∞	3
f	2	1	3	2	1	0	∞	∞	3
g	∞	∞	∞	∞	∞	∞	0	1	1
h	∞	∞	∞	∞	∞	∞	1	0	1

d(G)=3; r(G)=2; периферия: {a,c,e,f}; центр: {b,d} - для первой компоненты связности.
d(G)=1; r(G)=1; периферия: {g,h}; центр: {g,h} - для первой компоненты связности.

2.4 ВЕРШИННАЯ И РЕБЕРНАЯ СВЯЗНОСТЬ

Вершинной связностью графа G называется наименьшее число вершин, удаление которых приводит к несвязному, или тривиальному, графу; обозначается: χ(G).

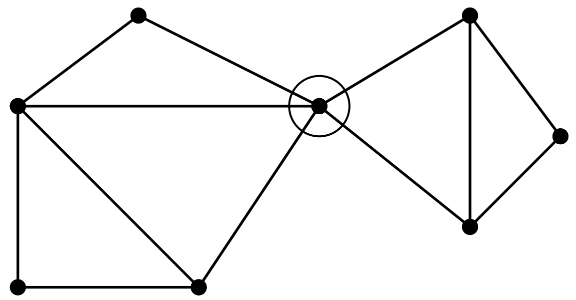
$$\chi(K_n)=n-1;\ \chi(P_n)=1;\ \chi(C_n)=2.$$

Реберной связностью графа G называется наименьшее число ребер, удаление которых приводит к несвязному графу; обозначается: $\lambda(G)$.

$$\lambda(K_n)=n-1;\ \lambda(P_n)=1;\ \lambda(C_n)=2.$$

Для тривиального графа реберную связность полагают равной 0.

Пример:



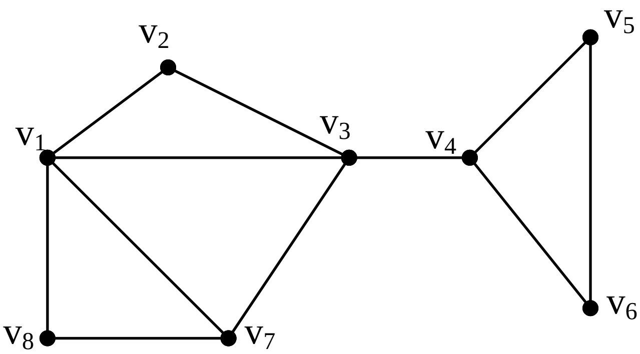
$\chi(G)=1$,
$\lambda(G)=2$.

Вершина v графа G называется **точкой сочленения,** или **разделяющей вершиной**, если граф $G\backslash\{v\}$ имеет больше компонент связности, чем граф G.

Ребро e графа G называется **мостом,** если его удаление приводит к нарушению связности графа.

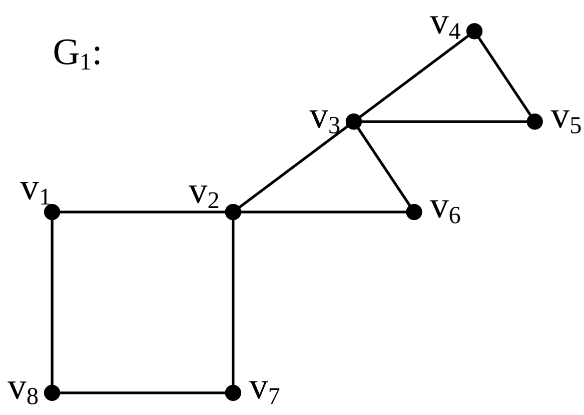
Пример:

G:



$\chi(G)=1$;
$\lambda(G)=1$;
точки сочленения: v_3, v_4;
мост: (v_3,v_4).

G_1:



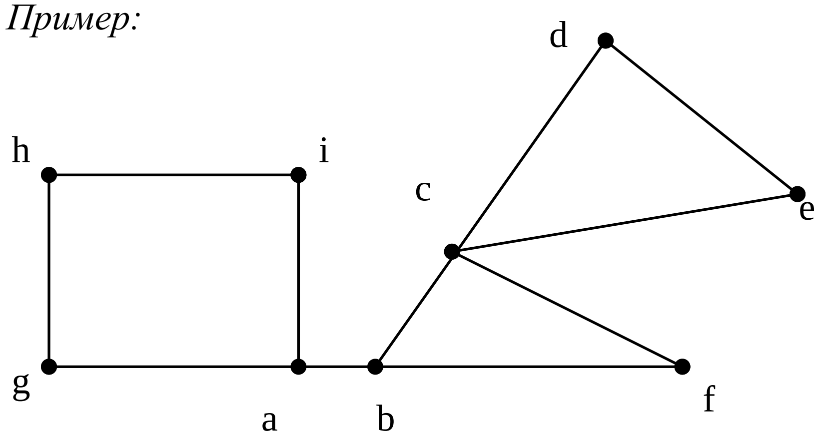
$\chi(G)=1$;
$\lambda(G)=2$;
точки сочленения: v_2, v_3.

Связный непустой граф называется **неразделимым**, если в нем отсутствуют точки сочленения.

Блок – это максимальный неразделимый подграф исходного графа.

Теорема Уитни. Пусть $\sigma(G)$ - минимальная степень вершин графа G, тогда справедливо следующее неравенство: $\chi(G) \le \lambda(G) \le \sigma(G)$.

Пример:



Граф не является неразделимым;
блоки графа**:** {c,d,e},{c,b,f},{a,g,h,i},{a,b}.

2.5 КРАТЧАЙШИЕ МАРШРУТЫ В ГРАФАХ

Пусть задан граф G=(V,E)**,** ребрам которого приписаны веса (стоимости), задаваемые матрицей $C = \| C_{ij} \| \quad i, j = \overline{1, p}$. Задача состоит в нахождении кратчайшего пути от заданной начальной вершины s∈V до заданной конечной вершины t∈V при условии, что такой путь существует. Элементы матрицы весов могут быть положительными, отрицательными или нулевыми. Единственное ограничение состоит в том, чтобы граф не содержал циклов с суммарным отрицательным весом.

Задачи данного типа имеют следующие модификации:

- для заданной начальной вершины **s** найти кратчайшие пути от нее ко всем другим вершинам графа;
- найти кратчайшие пути между всеми парами вершин графа.

2.5.1 АЛГОРИТМ ДЕЙКСТРЫ

<u>Постановка задачи.</u> Имеется произвольный взвешенный (n,m)-граф (в матрице весов нет отрицательных чисел), т.е.:

$$C_{i,j} = \begin{cases} \infty, & \text{если } (i, j) \notin E, \quad i \ne j; \\ 0, & \text{если } i = j; \\ const > 0, & \text{если } (i, j) \in E, \quad i \ne j. \end{cases}$$

Требуется найти кратчайший маршрут от вершины s ко всем остальным вершинам графа.

Алгоритм основан на приписывании вершинам временных пометок, причем пометка вершины дает верхнюю границу длины пути от s к данной вершине. Эти пометки постепенно уменьшаются с помощью итерационной процедуры, и на каждом шаге итерации одна из временных пометок становится постоянной. Последнее указывает на то, что пометка уже не является верхней границей, а дает точную длину кратчайшего пути от s к рассматриваемой вершине.

Алгоритм.

Пусть $l(x_i)$ – пометка вершины x_i, $\Gamma(p)$ – множество вершин графа, смежных с вершиной p.

Шаг 1. **Присвоение начальных значений.**

Положить $l(s)=0$ и считать эту пометку постоянной. Положить $l(x_i)=\infty$ для всех $x_i \neq s$ и считать эти пометки временными. Положить $p=s$.

Шаг 2. **Обновление пометок.**

Для всех $x_i \in \Gamma(p)$, пометки которых временные, заменить пометки в соответствии со следующим выражением:

$$l(x_i)=\min[l(x_i),l(p)+c(p,x_i)] \,. \qquad (1)$$

Шаг 3. **Превращение пометки в постоянную.**

Среди всех вершин с временными пометками найти такую, для которой $l(x_i^*)=\min[l(x_i)]$.

Шаг 4. Считать пометку вершины x_i^* постоянной; положить $p= x_i^*$.

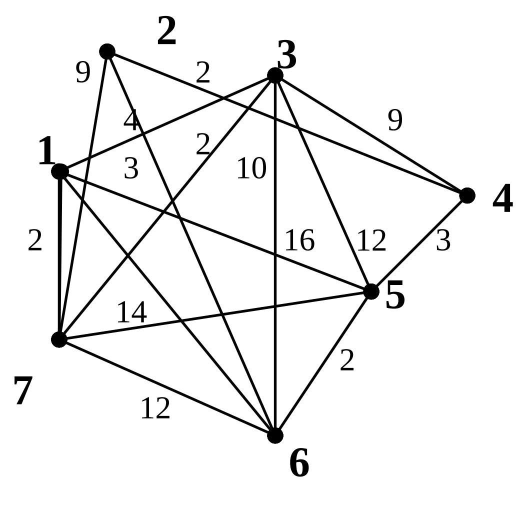
Шаг 5. Если p=t, то l(p) является длиной кратчайшего пути. Останов. Если $p\neq t$, то перейти к шагу 2.

Замечание. Как только длины кратчайших путей будут найдены, сами пути можно получить при помощи рекурсивной процедуры с использованием соотношения

$$L(x_i')+c(x_i', x_i)=L(x_i);$$

т.к. вершина $\mathbf{x_i'}$ непосредственно предшествует вершине $\mathbf{x_i}$ в кратчайшем пути от **s** к $\mathbf{x_i}$, то для любой вершины $\mathbf{x_i}$ соответствующую вершину $\mathbf{x_i'}$ можно найти как одну из оставшихся вершин.

Пример:



Матрица весов для графа G:

	1	2	3	4	5	6	7
1	0	∞	4	∞	16	14	2
2	∞	0	∞	2	∞	3	9
3	4	∞	0	9	12	10	2
4	∞	2	9	0	3	∞	∞
5	16	∞	12	3	0	2	14
6	14	3	10	∞	2	0	12
7	2	9	2	∞	14	12	0

Найти кратчайшие расстояния от 1 вершины ко всем остальным.

1) $l(1) = 0$;
   $l(2) = l(3) = \ldots = l(7) = \infty$;
   $p = 1$.
2) $\Gamma(1) = \{3,5,6,7\}$;
   $l(3) = \min(l(3), l(1)+C(1,3)) = \min(\infty, 0+4)=4$;
   $l(5) = \min(l(5), l(1)+C(1,5)) = 16$;
   $l(6) = \min(l(2), l(1)+C(1,2)) = 14$;
   $l(7) = \min(l(2), l(1)+C(1,2)) = 2$.
3) $\min(4,16,14,2) = 2$;
   $\min = l(7) = 2$;
   $p = 7$.
4) $\Gamma(7) = \{1,2,3,5,6\}$;
   $l(2) = \min(l(2), l(7)+C(2,7)) = \min(\infty, 2+9)=11$;
   $l(3) = \min(l(3), l(7)+C(3,7)) = 4$;
   $l(5) = \min(l(5), l(7)+C(5,7)) =16$;
   $l(6) = \min(l(6), l(7)+C(6,7)) = 14$.
5) $\min(11,4,16,14) = 4$;
   $\min = l(3) = 4$;
   $p = 3$.
6) $\Gamma(3) = \{1,4,5,6,7\}$;
   $l(4) = \min(l(4), l(3)+C(4,3)) = 13$;
   $l(5) = \min(l(5), l(3)+C(5,3)) = 16$;
   $l(6) = \min(l(6), l(3)+C(6,3)) = 14$.
7) $\min(13,16, 14) = 13$;
   $\min = l(4) = 13$;
   $p=4$.
8) $\Gamma(4) = \{2,3,5\}$;
   $l(5) = \min(l(5), l(4)+C(5,4)) = 16$;
   $l(2) = \min(l(6), l(3)+C(6,3)) = 11$.
9) $\min(11,16) = 11$;
   $\min = l(2) =11$;

$p = 2$.

10) $Г(2) = \{4,6,7\}$;

$l(6) = \min(l(6), l(2)+C(2,6)) = 14$.

11) $\min = l(6) =14$;

$p = 6$.

12) $l(5) = \min(l(5), l(6)+C(5,6)) = 16$;

$p=5$.

Таким образом, найдены следующие кратчайшие маршруты от вершины 1:
(1-2): (1,7,2)=11;
(1-3): (1,3)=4;
(1-4): (1,3,4)=(1,7,3,4)=(1,7,2,4)=13;
(1-5): (1,5)=16;
(1-6): (1,6)=14;
(1-7): (1,7)=2.

2.5.2 АЛГОРИТМ ФОРДА

Алгоритм Форда – обобщение алгоритма Дейкстры для случая, когда некоторые ребра имеют отрицательные веса.

На шаге 2 алгоритма пересчет $l(x)$ с помощью соотношения (1) производится для любых вершин, а не только для непомеченных. Значения $l(x)$ могут уменьшаться как для помеченных, так и для непомеченных вершин, т.к. существуют ребра с отрицательным весом.

Если для некоторой помеченной вершины x происходит уменьшение величины $l(x)$, то с этой вершины пометка снимается.

Процедура заканчивается, когда все вершины помечены, и после выполнения шага 2 ни одна из величин $l(x)$ не изменяется.

Алгоритм.

Пусть $l^k(x_i)$ – пометка вершины x_i в конце (k+1)-й итерации, $Г(s)$ – множество вершин, смежных с вершиной s.

Шаг 1. **Присвоение начальных значений.**

Положить $S= Г(s)$, $k=1$, $l^1(s)=0$, $l^1(x_i)=c(s, x_i)$ для всех $x_i \in Г(s)$. Положить $l^1(x_i)=\infty$ для всех остальных x_i.

Шаг 2. **Обновление пометок.**

Для каждой вершины $x_i \in Г(s)$ $(x_i \neq s)$ изменить ее пометку следующим образом:

$$l^{k+1}(x_i)=\min[l^k(x_i),\min\{ l^k(x_i)+c(x_j, x_i)\}] \text{ для } x_j \in T_i, \text{ где } T_i=Г(x_i)\cap S.$$

Шаг 3. **Проверка на окончание.**

Если $k \le n-1$ и $l^{k+1}(x_i)= l^k(x_i)$ для всех x_i, то пометки равны длинам кратчайших путей. Останов.

Если $k<n-1$ и $l^{k+1}(x_i)\neq l^k(x_i)$ для некоторой вершины x_i, то перейти к шагу 4.

Если $k=n-1$ и $l^{k+1}(x_i) \neq l^k(x_i)$ для некоторой вершины x_i, то в графе существует цикл отрицательного веса и задача не имеет решения. Останов.

Шаг 4. **Подготовка к следующей итерации.**

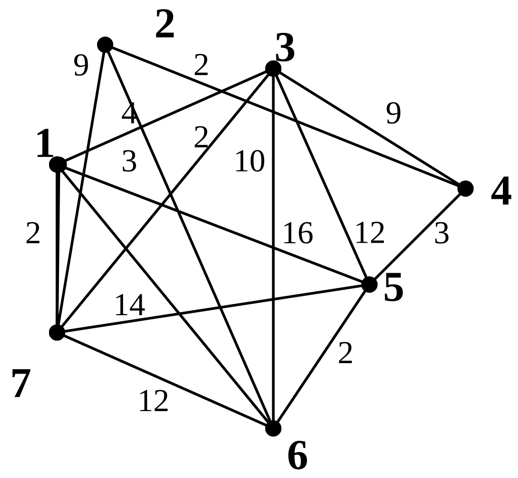
Обновить множество следующим образом:

$$S=\{x_i \mid l^{k+1}(x_i) \neq l^k(x_i)\}.$$

Шаг 5. Положить $k=k+1$ и перейти к шагу 2.

Замечание. Алгоритм Форда не решает задачу при наличии цикла отрицательной длины. Для его обнаружения в процессе работы алгоритма просчитывается, сколько раз помечается каждая вершина: если вершина помечается ровно n раз, где n – число вершин графа, - процедура заканчивает работу: существует цикл отрицательной длины.

Пример:



Матрица весов для графа G:

	1	2	3	4	5	6	7
1	0	∞	4	∞	16	14	2
2	∞	0	∞	2	∞	3	9
3	4	∞	0	9	12	10	2
4	∞	2	9	0	3	∞	∞
5	16	∞	12	3	0	2	14
6	14	3	10	∞	2	0	12
7	2	9	2	∞	14	12	0

1) $s=1$; вершины, смежные вершине 1: $S= \Gamma(s)=\{3,5,6,7\}$,

$k=1$; $l^1(1)=0$,

$l^1(3)=4$, $l^1(5)=16$, $l^1(6)=14$, $l^1(7)=2$, $l^1(2)= l^1(4)=\infty$;

2) $\Gamma(s)=\{2,3,4,5,6,7\}$ - вершины, смежные вершинам 3,5,6,7;

для 2: $T_2=\{7,6,4\}\cap\{3,5,6,7\}=\{7,6\}$, где $\{7,6,4\}$ - вершины, смежные вершине 2;

$l^2(2)=\min(\infty, \min(2+9,14+3))=11$;

для 3: T_3={1,4,5,6,7}∩{3,5,6,7}={5,6,7}, где {1,4,5,6,7} - вершины, смежные вершине 3;

$l^2(3)$=min(4, min(16+12,14+10,2+2))=4;

для 4: T_4={2,3,5}∩{3,5,6,7}={3,5}, где {2,3,5} - вершины, смежные вершине 4;

$l^2(4)$=13;

для 5: T_5={1,3,4,6,7}∩{3,5,6,7}={3,6,7}, где {1,3,4,6,7} - вершины, смежные вершине 5;

$l^2(5)$= 16;

для 6: T_2={1,2,3,5,7}∩{3,5,6,7}={3,5,7}, где {1,2,3,5,7} - вершины, смежные вершине 6;

$l^2(6)$= 14;

для 7: T_2={1,2,3,5,6}∩{3,5,6,7}={3,5,6}, где {1,2,3,5,6} - вершины, смежные вершине 7;

$l^2(7)$=2;

3) переходим к шагу 4, обновляем множество S;

4) S={2,4};

5) k=2; переходим к шагу 2;

6) Г(S)={2,3,4,5,6,7} - вершины, смежные вершинам 2,4;

для 2: T_2={7,6,4}∩{2,4}={4};

$l^3(2)$=min(11, 13+2))=11;

для 3: T_3={1,4,5,6,7}∩{2,4}={4};

$l^3(3)$=min(4, 13+9)=4;

для 4: T_4={2,3,5}∩{2,4}={2};

$l^3(4)$= min(13, 11+2)=13;

для 5: T_5={1,3,4,6,7}∩{2,4}={4};

$l^3(5)$= 16;

для 6: T_2={1,2,3,5,7}∩{2,4}={2};

$l^3(6)$= 14;

для 7: T_2={1,2,3,5,6}∩{2,4}={2}

$l^3(7)$=2.

7) $k \le n-1$ и $l^{k+1}(x_i)= l^k(x_i)$ для всех x_i, следовательно, пометки равны длинам кратчайших путей.

Кратчайшие маршруты от вершины 1:

(1-2):(1,7,2)=11;

(1-3): (1,3)=4;

(1-4): (1,3,4)=(1,7,3,4)=(1,7,2,4)=13;

(1-5): (1,5)=16;

(1-6): (1,6)=14;

(1-7): (1,7)=2.

2.5.3 АЛГОРИТМ ФЛОЙДА

Алгоритм базируется на использовании последовательности из n преобразований (итераций) начальной матрицы весов. При этом на k-той итерации матрица представляет длины кратчайших путей между каждой парой вершин с тем ограничением, что путь между x_i и x_j (для любых x_i и x_j) содержит в качестве промежуточных только вершины из множества $\{ x_1 , x_2 ,\dots, x_n\}$.

Все циклы в графе G имеют неотрицательный суммарный вес.

Алгоритм.

Шаг 1. **Присвоение начальных значений.**

Положить k=0.

Шаг 2.

k=k+1.

Шаг 3.

Для всех $i \neq k$ таких что $c_{ik}\neq\infty$, и для всех $j \neq k$ таких что $c_{kj} \neq\infty$ введем операцию:

$$c_{ij}=\min[c_{ij}, (c_{ik}+c_{kj})]. \qquad (2)$$

Шаг 4. **Проверка на окончание.**

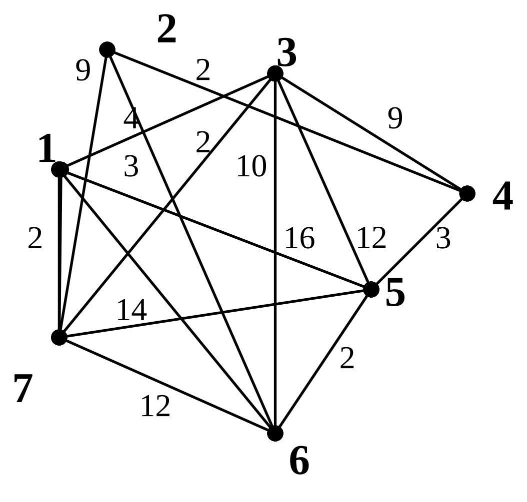
Если $c_{ij} <0$, то в графе существует цикл отрицательного веса, содержащий вершину x_i; решения нет. Останов.

Если все $c_{ij} \geq 0$ и k=n, то получено решение и матрица дает длины всех кратчайших путей. Останов.

Если все $c_{ij} \geq 0$, но k<n, то вернуться к шагу 2.

Пример:

Пусть задан граф G следующего вида:



Матрица весов: Матрица путей:

k=0;

k=1;

	1	2	3	4	5	6	7
1	0	∞	4	∞	16	14	2
2	∞	0	∞	2	∞	3	9
3	4	∞	0	9	12	10	2
4	∞	2	9	0	3	∞	∞
5	16	∞	12	3	0	2	14
6	14	3	10	∞	2	0	12
7	2	9	2	∞	14	12	0

	1	2	3	4	5	6	7
1	1	1	1	1	1	1	1
2	2	2	2	2	2	2	2
3	3	3	3	3	3	3	3
4	4	4	4	4	4	4	4
5	5	5	5	5	5	5	5
6	6	6	6	6	6	6	6
7	7	7	7	7	7	7	7

k=2;

	1	2	3	4	5	6	7
1	0	∞	4	∞	16	14	2
2	∞	0	∞	2	∞	3	9
3	4	∞	0	9	12	10	2
4	∞	2	9	0	3	**5**	**11**
5	16	∞	12	3	0	2	14
6	14	3	10	**5**	2	0	12
7	2	9	2	**11**	14	12	0

	1	2	3	4	5	6	7
1	1	1	1	1	1	1	1
2	2	2	2	2	2	2	2
3	3	3	3	3	3	3	3
4	4	4	4	4	4	2	2
5	5	5	5	5	5	5	5
6	6	6	6	2	6	6	6
7	7	7	7	2	7	7	7

k=3;

	1	2	3	4	5	6	7
1	0	∞	4	**13**	16	14	2
2	∞	0	∞	2	∞	3	9
3	4	∞	0	9	12	10	2
4	**13**	2	9	0	3	5	11
5	16	∞	12	3	0	2	14
6	14	3	10	5	2	0	12
7	2	9	2	11	14	12	0

	1	2	3	4	5	6	7
1	1	1	1	3	1	1	1
2	2	2	2	2	2	2	2
3	3	3	3	3	3	3	3
4	3	4	4	4	4	2	2
5	5	5	5	5	5	5	5
6	6	6	6	2	6	6	6
7	7	7	7	2	7	7	7

k=4;

	1	2	3	4	5	6	7
1	0	**15**	4	13	16	14	2
2	**15**	0	**11**	2	**5**	3	9
3	4	**11**	0	9	12	10	2
4	13	2	9	0	3	5	11
5	16	**5**	12	3	0	2	14
6	14	3	10	5	2	0	12
7	2	9	2	11	14	12	0

	1	2	3	4	5	6	7
1	1	4	1	3	1	1	1
2	4	2	4	2	4	2	2
3	3	4	3	3	3	3	3
4	3	4	4	4	4	2	2
5	5	4	5	5	5	5	5
6	6	6	6	2	6	6	6
7	7	7	7	2	7	7	7

k=5;

	1	2	3	4	5	6	7
1	0	15	4	13	16	14	2
2	15	0	11	2	5	3	9
3	4	11	0	9	12	10	2
4	13	2	9	0	3	5	11
5	16	5	12	3	0	2	14
6	14	3	10	5	2	0	12
7	2	9	2	11	14	12	0

	1	2	3	4	5	6	7
1	1	4	1	3	1	1	1
2	4	2	4	2	4	2	2
3	3	4	3	3	3	3	3
4	3	4	4	4	4	2	2
5	5	4	5	5	5	5	5
6	6	6	6	2	6	6	6
7	7	7	7	2	7	7	7

k=6;

	1	2	3	4	5	6	7
1	0	15	4	13	16	14	2
2	15	0	11	2	5	3	9
3	4	11	0	9	12	10	2
4	13	2	9	0	3	5	11
5	16	5	12	3	0	2	14
6	14	3	10	5	2	0	12
7	2	9	2	11	14	12	0

	1	2	3	4	5	6	7
1	1	4	1	3	1	1	1
2	4	2	4	2	4	2	2
3	3	4	3	3	3	3	3
4	3	4	4	4	4	2	2
5	5	4	5	5	5	5	5
6	6	6	6	2	6	6	6
7	7	7	7	2	7	7	7

k=7;

	1	2	3	4	5	6	7
1	0	**11**	4	13	16	14	2
2	**11**	0	11	2	5	3	9
3	4	11	0	9	12	10	2
4	13	2	9	0	3	5	11
5	16	5	12	3	0	2	14
6	14	3	10	5	2	0	12
7	2	9	2	11	14	12	0

	1	2	3	4	5	6	7
1	1	7	1	3	1	1	1
2	7	2	4	2	4	2	2
3	3	4	3	3	3	3	3
4	3	4	4	4	4	2	2
5	5	4	5	5	5	5	5
6	6	6	6	2	6	6	6
7	7	7	7	2	7	7	7

Матрица $\theta=[\theta_{ij}]$ – матрица, в которой элемент θ_{ij} указывает вершину, непосредственно предшествующую вершине x_j в кратчайшем пути от x_i к x_j. Матрице θ присваиваются начальные значения $\theta_{ij}= x_i$ для всех x_i и x_j. На шаге 3 алгоритма обновление матрицы происходит следующим образом:

$$\theta_{kj} = \begin{cases} \theta_{kj}, \text{если } (c_{ik} + c_{kj}) < c_{ij} \text{ в квадратных скобках в выражении (2);} \\ \\ \text{не изменяется, если} \quad c_{ij} \le (c_{ik} + c_{kj}). \end{cases}$$

В конце алгоритма кратчайшие пути получаются непосредственно из заключительной матрицы θ.

Таким образом, матрица всех кратчайших маршрутов:

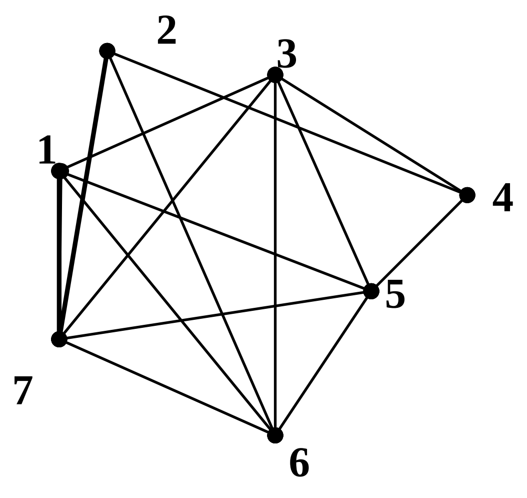
	1	2	3	4	5	6	7
1	0	11	4	13	16	14	2
2	11	0	11	2	5	3	9
3	4	11	0	9	12	10	2
4	13	2	9	0	3	5	11
5	16	5	12	3	0	2	14
6	14	3	10	5	2	0	12
7	2	9	2	11	14	12	0

Матрица путей:

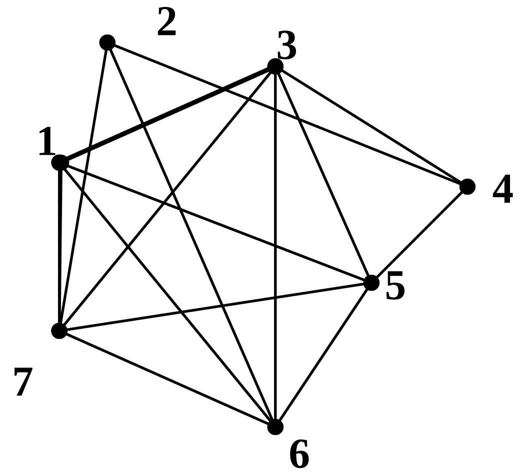
	1	2	3	4	5	6	7
1	0	1,7,2	1,3	1,3,4	1,5	1,6	1,7
2	2,7,1	0	2,4,3	2,4	2,4,5	2,6	2,7
3	3,1	3,4,2	0	3,4	3,5	3,6	3,7
4	4,3,1	4,2	4,3	0	4,5	4,2,6	4,2,7
5	5,1	5,4,2	5,3	5,4	0	5,6	5,7
6	6,1	6,2	6,3	6,2,4	6,5	0	6,7
7	7,1	7,2	7,3	7,2,4	7,5	7,6	0

Кратчайшие маршруты от вершины 1:

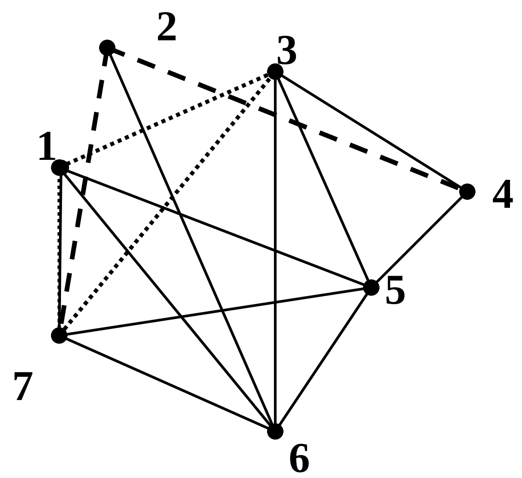
(1-2): (1,7,2)=11;



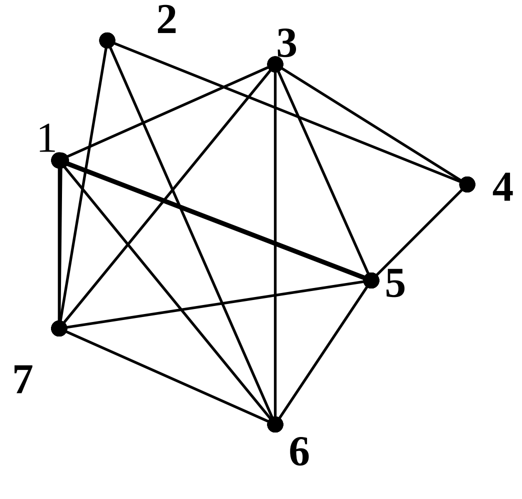
(1-3): (1,3)=4;



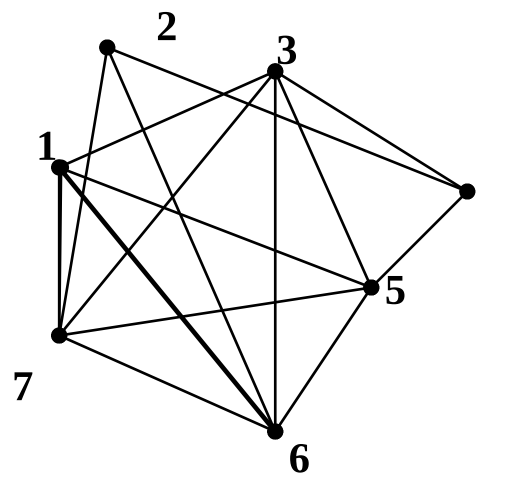
(1-4): (1,3,4)=(1,7,3,4)=(1,7,2,4)=13;



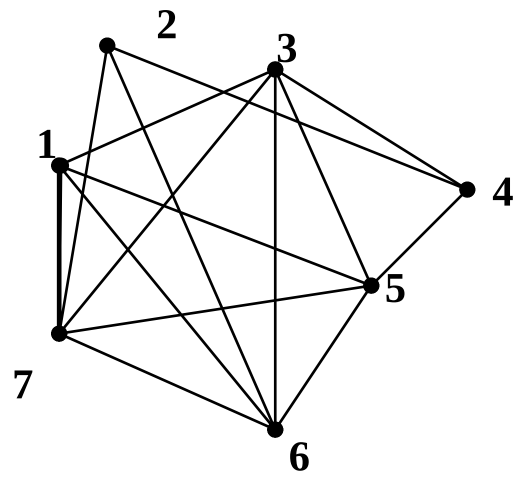
(1-5): (1,5)=16;



(1-6): (1,6)=14;



(1-7): (1,7)=2.



2.6 КОНТРОЛЬНЫЕ ВОПРОСЫ

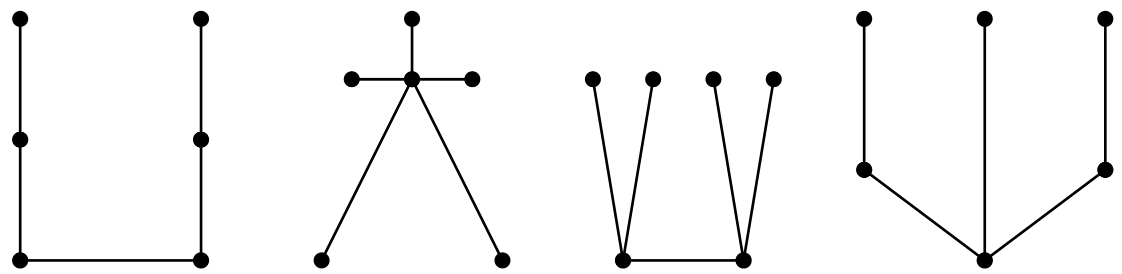
1. Определение маршрута, цепи, цикла.
2. Какой граф называется связным? Что такое компонента связности?
3. Привести примеры графов, которые имеют все периферийные и все центральные вершины.
4. Что такое эксцентриситет?
5. Чем диаметр графа отличается от его радиуса (дайте их определение)?
6. Что такое число вершинной и реберной связности?
7. Дайте определения моста и точки сочленения.
8. Для произвольного графа постройте матрицу расстояний (вес любого ребра равен 1).
9. Привести пример графа, удовлетворяющего строгому неравенству теоремы Уитни.
10. Сформулируйте задачу нахождения кратчайших маршрутов в графе.

3 ДЕРЕВЬЯ И ОСТОВЫ

3.1 ОСНОВНОЕ ОПРЕДЕЛЕНИЕ ДЕРЕВА

Дерево - это специфический вид графов. **Деревом** называется связный граф, не содержащий циклов. Любой граф, не содержащий циклов, называется **ациклическим,** или **лесом**. Компонентами леса являются деревья.

Пример:



3.2 ДРУГИЕ ОПРЕДЕЛЕНИЯ ДЕРЕВА

Теорема**,** дающая 5 различных определений дерева (**теорема эквивалентности).**

Для любого (p,q) - графа следующие утверждения эквивалентны:

1) граф G – дерево: связный ацикличный граф;
2) любые две несовпадающие вершины графа G соединены единственной простой цепью;
3) граф G связен и q=p-1(количество ребер в нем на 1 меньше, чем количество вершин);
4) граф G ацикличен и q=p-1;
5) граф G ацикличен и, если любую пару его несовпадающих несмежных вершин соединить ребром e, то полученный граф G+e будет содержать в точности один цикл.

Теорема о висячих вершинах дерева.

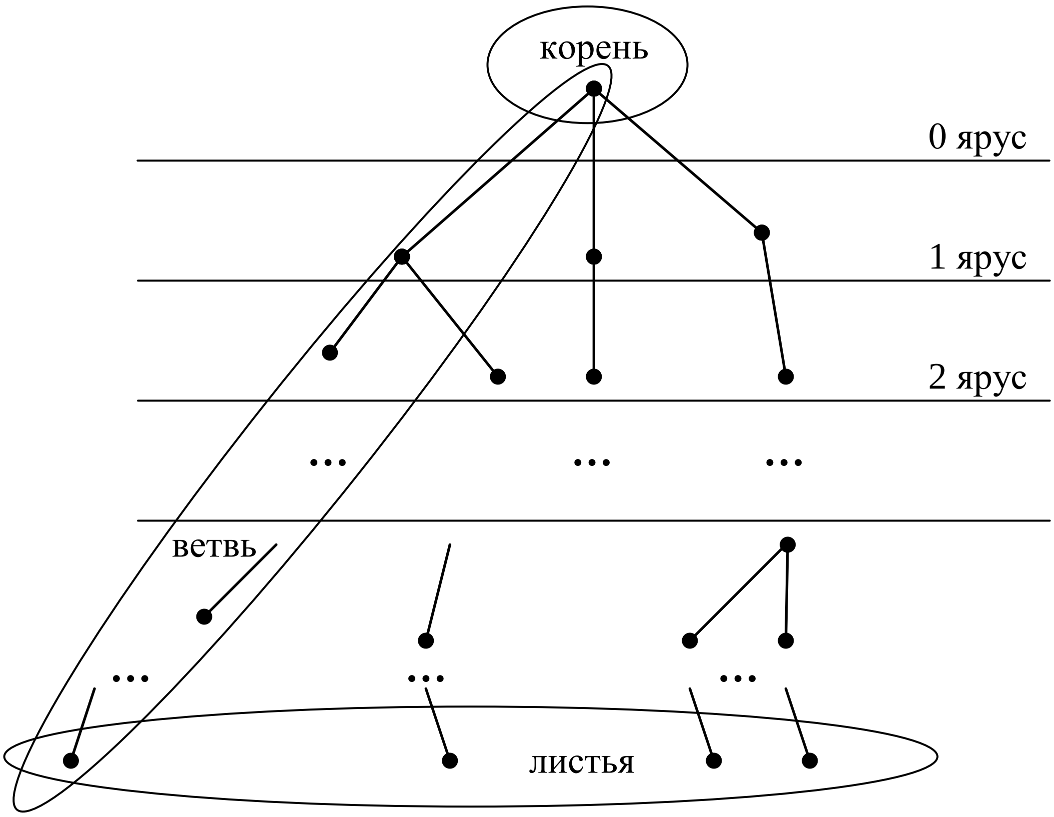
В любом нетривиальном ($p \geq 2$) дереве имеются, по крайней мере, две висячие вершины.

Теорема о центральных вершинах дерева.

Центр любого дерева состоит либо из одной, либо из двух (смежных) вершин.

3.3 ЯРУСНАЯ ФОРМА ПРЕДСТАВЛЕНИЯ ДЕРЕВЬЕВ

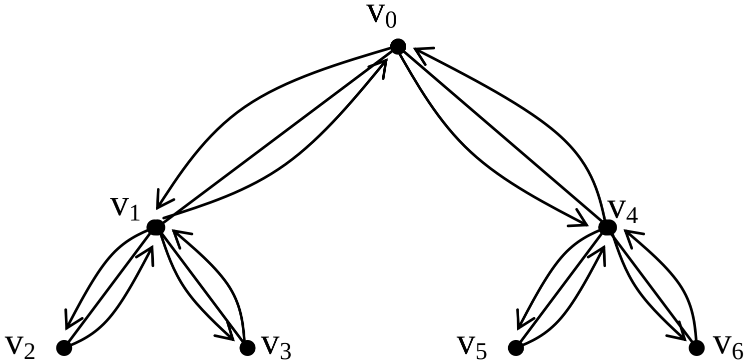
Выберем некоторую произвольную фиксированную вершину дерева и назовем ее **корнем**. Будем полагать, что корень дерева расположен на 0 ярусе. Все остальные вершины сориентируем относительно корня следующим образом: на i-й ярус поместим вершины с расстоянием от корня, равным i. Концевые висячие вершины будем называть **листами**, а геодезические от корня к листу - **ветвями**.



3.4 СПОСОБЫ ОБХОДА ДЕРЕВЬЕВ

Существует два основных способа.

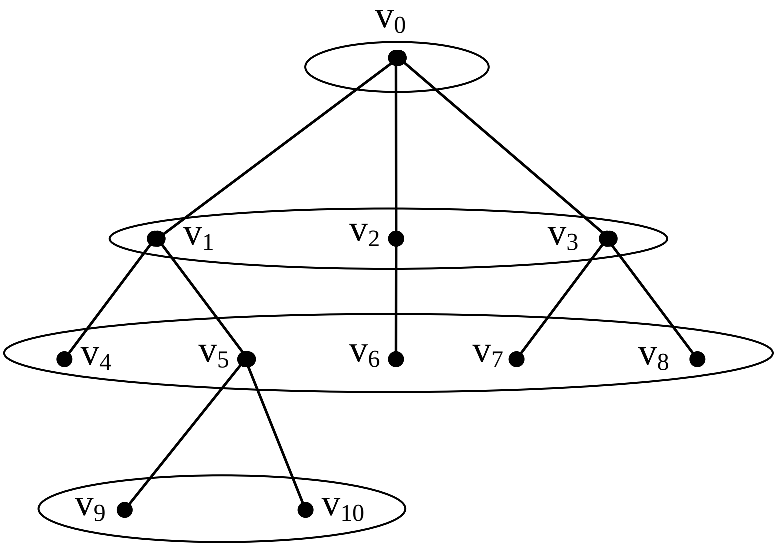
1. **Способ обхода (поиска) в глубину** подразумевает просмотр ветвей в ярусном представлении дерева.



Начинаем поиск с некоторой фиксированной вершины v_0. Находим вершину u, смежную с v_0, и повторяем процесс, начиная с вершины u:

- если существует не просмотренная вершина, смежная с вершиной u, рассматриваем ее и, начиная с нее, продолжаем поиск;
- если не существует ни одной новой вершины, смежной с u, то говорят, что вершина u использована, и делается возврат в вершину, из которой мы попали в вершину u; продолжаем процесс;
- если на каком-то шаге u= v_0, и нет ни одной не просмотренной вершины, смежной v_0, то алгоритм заканчивает работу.

2. **Способ обхода (поиска) в ширину** подразумевает просмотр вершин по ярусам с перебором вершин одного яруса.



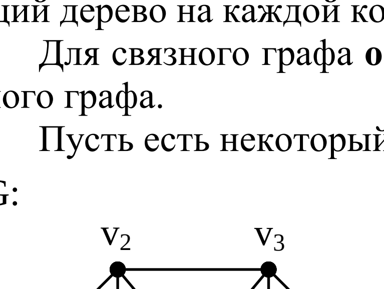
3.5 ОСТОВЫ

Остов (каркас, скелет) графа G – это остовный подграф графа G, задающий дерево на каждой компоненте связности графа G.

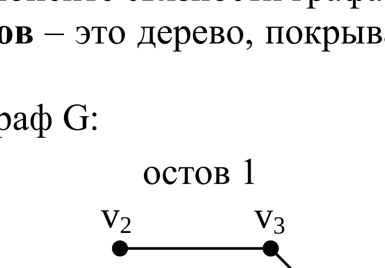
Для связного графа **остов** – это дерево, покрывающее все вершины исходного графа.

Пусть есть некоторый граф G:

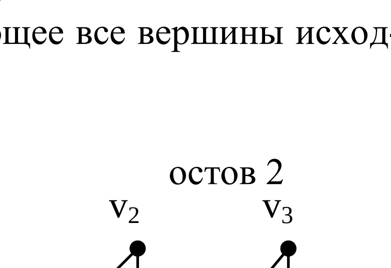
G:



остов 1



остов 2



Очевидно, что в каждом графе существует остов, в общем случае, не один. Его можно получить, разрушая циклы в каждой компоненте связности.

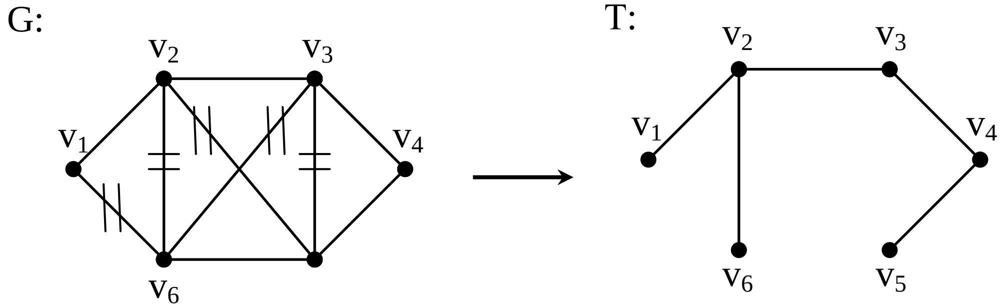
Ребра остова T некоторого графа G называются **ветвями**, а ребра графа G, не вошедшие в остов, называются **хордами**.

В первом остове графа G: (v_2,v_3) , (v_3,v_4) , (v_4,v_5) , (v_5,v_6) , (v_1,v_6) - ветви;
(v_1,v_2) , (v_2,v_6) , (v_2,v_5) , (v_3,v_6) , (v_3,v_5) - хорды.

3.6 АЛГОРИТМЫ ПОСТРОЕНИЯ ОСТОВА

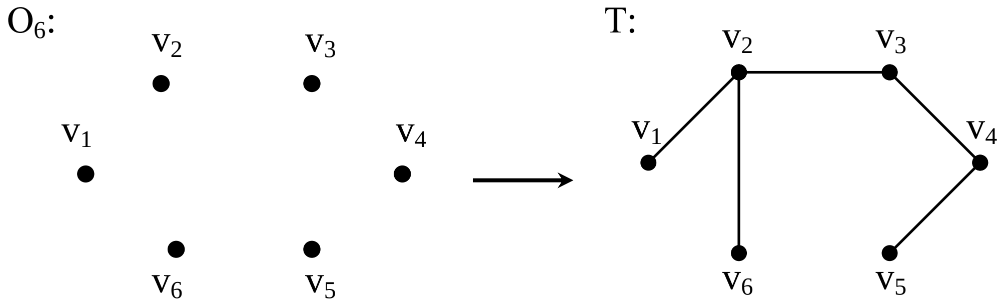
1. **Удаление ребер исходного графа**; причем очередное удаляемое ребро не должно приводить к нарушению связности, но делать граф ацикличным.

Пример:



2. **Добавление ребра пустому графу (**количество вершин в нем должно совпадать с количеством вершин исходного графа**)**; причем очередное добавляемое ребро не должно приводить к образованию цикла.

Пример:



3. **Остовы кратчайших маршрутов**: из произвольной вершины алгоритм строит не просто дерево, а геодезические до каждой вершины графа.

3.7 МАТРИЧНАЯ ТЕОРЕМА КИРХГОФА

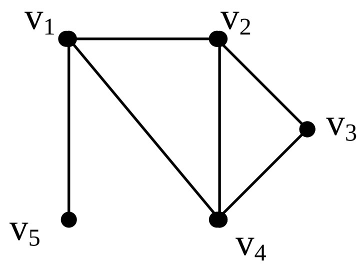
Матрица Кирхгофа – это квадратная матрица

$$M=\|m_{ij}\|,\quad i,j=\overline{1,p},\quad m_{ij}=\begin{cases}\deg v_i, & i=j;\\ -a_{ij}, & i\neq j,\end{cases}$$

где a_{ij} – соответствующий элемент матрицы смежности.

Пример:

G:



M=

	1	2	3	4	5
1	3	-1	0	-1	-1
2	-1	3	-1	-1	0
3	0	-1	2	-1	0
4	-1	-1	-1	3	0
5	-1	0	0	0	1

Замечание. Матрица Кирхгофа обладает следующим свойством: сумма элементов любой строки матрицы равна сумме элементов любого столбца, равна сумме всех элементов матрицы и равна нулю.

Теорема Кирхгофа. Число остовных деревьев в связном графе G порядка n ($n \geq 2$) равно алгебраическому дополнению любого элемента матрицы Кирхгофа.

В связном помеченном графе все алгебраические дополнения матрицы Кирхгофа равны между собой и определяют общее число помеченных остовов графа.

Следствие из теоремы Кирхгофа. При $n \geq 2$ (где n – количество вершин графа) число остовов полного графа K_n равно n^{n-2}.

Теорема А. Кэли (1897 г.). Число помеченных деревьев порядка n равно n^{n-2}.

3.8 АЛГОРИТМЫ ПОИСКА ОСТОВОВ КРАТЧАЙШИХ МАРШРУТОВ

Постановка задачи. Имеется связный граф G = (V,E), заданный матрицей весов $C = \|C_{ij}\|$, $i, j = \overline{1, n}$. Требуется найти остов с минимальным суммарным весом ребер.

3.8.1 Алгоритм Краскала

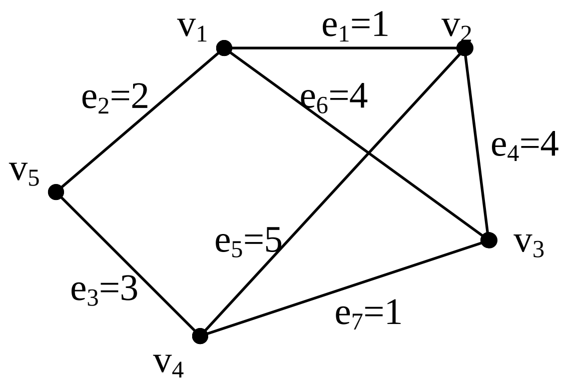
Ребра графа упорядочиваются в порядке не убывания их весов. На каждом шаге к пустому графу O_p, где p – количество вершин исходного графа, добавляется ребро с минимальным весом из списка. Добавляемое ребро не должно приводить к образованию цикла. Алгоритм заканчивает работу, если количество ребер в формируемом графе станет равным p – 1.

3.8.2 Алгоритм Прима

Отличается от алгоритма Краскала тем, что на каждом шаге строится не просто граф без циклов, но дерево.

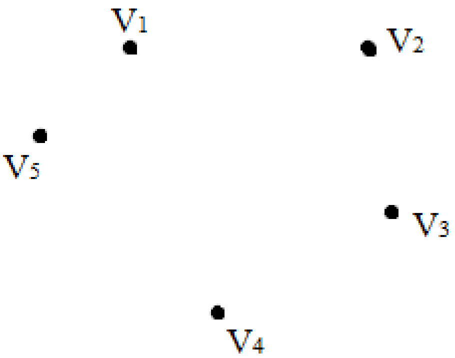
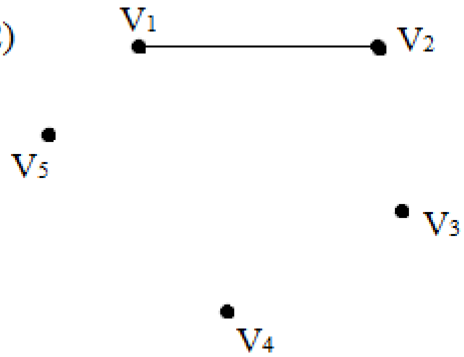
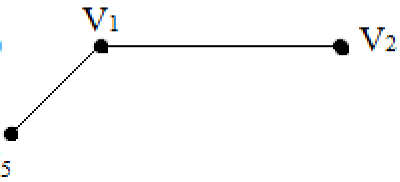
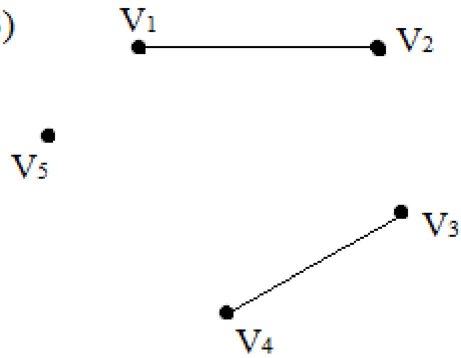
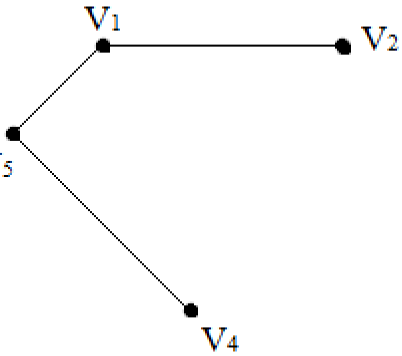
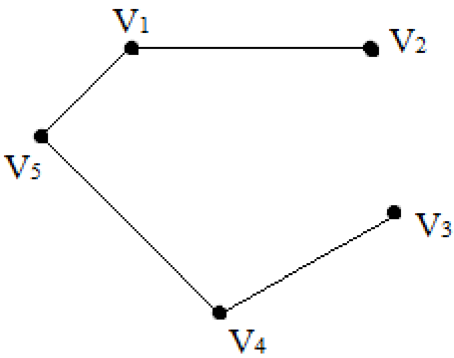
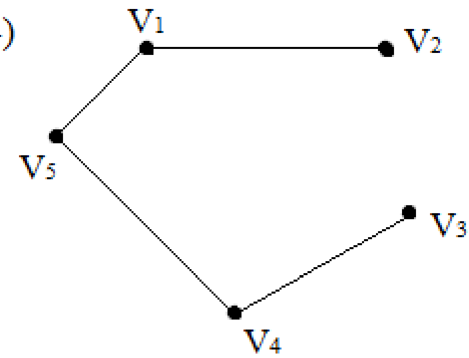
Упорядочиваются ребра исходного графа в порядке не убывания их весов. Строится пустой граф O_p, где p – количество вершин исходного графа. На каждом шаге в граф O_p добавляется ребро из списка ребер таким образом, чтобы не образовался цикл и не нарушалась связность. Для этого список ребер каждый раз просматривается сначала. Алгоритм строит остов кратчайших маршрутов посредством разрастания поддерева T за счет присоединения очередного ребра (x_i, x_j) с наименьшим весом.

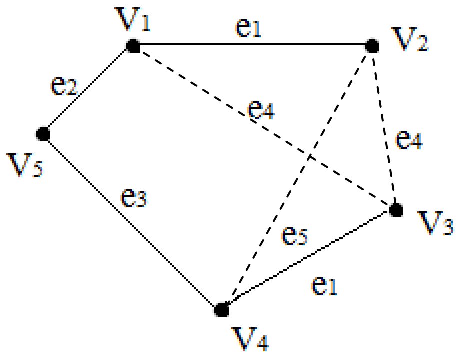
Пример:



Для заданного графа G упорядочим ребра в порядке не убывания их весов.

Ребра:	Вес:
$e_1 = (v_1,v_2)$,	– 1
$e_7 = (v_3,v_4)$	– 1
$e_2 = (v_1,v_5)$	– 2
$e_3 = (v_4,v_5)$	– 3
$e_6 = (v_1,v_3)$	– 4
$e_4 = (v_2,v_3)$	– 4
$e_5 = (v_2,v_4)$	– 5

алгоритм Краскала	алгоритм Прима
1) 	1) 
2) 	2) 
3) 	3) 
4) 	4) 



(v_1,v_3), (v_2,v_3), (v_2,v_4) – хорды;

(v_1,v_2), (v_1,v_5), (v_4,v_5), (v_3,v_4) – ветви.

Общий суммарный вес $\sum = 7$.

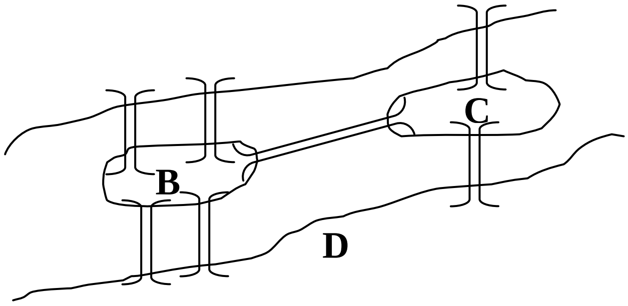
3.9 КОНТРОЛЬНЫЕ ВОПРОСЫ

1. Определения дерева, леса.
2. Рисунки деревьев заданного порядка с указанием центральных вершин.
3. Сгенерировать все различные абстрактные, неизоморфные друг другу деревья порядка (4,7). Разделить множество деревьев на 2 подмножества с одной и двумя центральными вершинами.
4. Теорема о висячих вершинах дерева.
5. Ярусное представление деревьев.
6. Перечислить способы обходы деревьев. Найти обходы для заданного дерева.
7. Определение остова, ветвей, хорд. Алгоритмы построения остовных деревьев.
8. Остовы кратчайших маршрутов. Отличие алгоритмов Прима и Краскала.
9. Матричная теорема Кирхгофа и следствие из нее.
10. Теорема Кэли.

4 ЦИКЛОМАТИКА

4.1 ЭЙЛЕРОВЫ ГРАФЫ

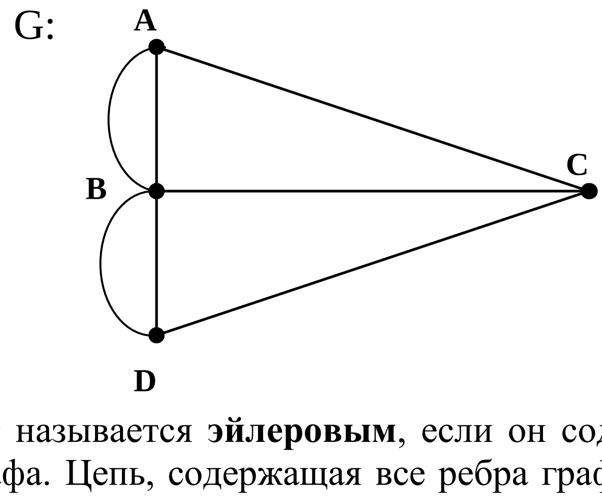
Начало теории графов, как отмечалось ранее, связывают с так называемой задачей о кенигсбергских мостах. Эта знаменитая в свое время задача состоит в следующем. Семь мостов г. Кенигсберга (ныне Калининград) были расположены на реке Прегель так, как изображено на рисунке. Возможно ли, выйдя из дома, вернуться об-



ратно, пройдя в точности один раз по каждому мосту?

Сопоставим рисунку граф G, вершины которого соответствуют четырем разделяемым рекой участкам суши A, B, C и D, а ребра - мостам.

Таким образом, данную задачу на языке теории графов можно сформулировать следующим образом: существует ли в мультиграфе G цикл, содержащий все его ребра ровно раз? В 1767 г. Л.Эйлер доказал, что данная задача не имеет решения. Он сформулировал и решил общую проблему теории графов: при каких условиях связный граф содердит цикл, проходящий через каждое его ребро?

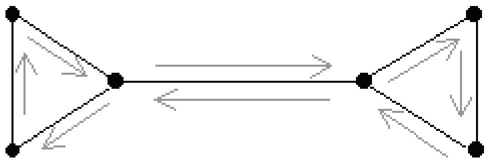


Цикл в графе G называется **эйлеровым**, если он содержит все ребра исходного графа. Цепь, содержащая все ребра графа, называется **эйлеровой цепью**.

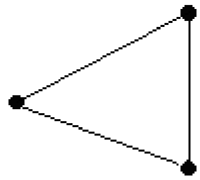
Связный **граф** называется **эйлеровым**, если он содержит эйлеров цикл.

Пример: не эйлеров граф эйлеров граф

G_1:



G_2:



Теорема о существовании эйлерового цикла в графе. Связный граф G является эйлеровым тогда и только тогда, когда степени всех его вершин четны.

Следствие №1. Если граф G эйлеров, то множество его ребер можно разбить на простые циклы.

Следствие №2. Для того чтобы связный граф G покрывался единственной эйлеровой цепью, необходимо и достаточно, чтобы он содержал ровно две вершины с нечетной степенью. Тогда цепь начинается в одной из этих вершин и заканчивается в другой.

4.2 АЛГОРИТМ ПОСТРОЕНИЯ ЭЙЛЕРОВОГО ЦИКЛА, ИЛИ АЛГОРИТМ ФЛЁРИ

Алгоритм сводится к нумерации ребер исходного графа от 1 до q так, чтобы номер ребра указывал, каким по счету это ребро войдет в эйлеров цикл.

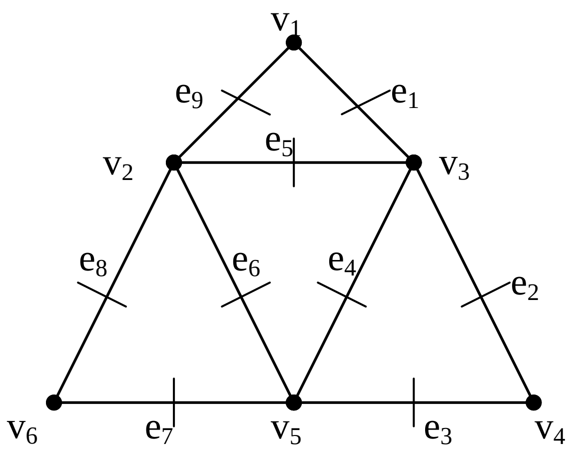
Алгоритм.

Начиная с любой вершины v, присваиваем ребру (v,u) номер 1. Вычеркиваем это ребро из списка ребер и переходим к вершине u.

Пусть w – вершина, в которую мы попали в результате выполнения предыдущего шага и k – номер, присвоенный очередному ребру на этом шаге. Выбираем произвольное ребро, инцидентное вершине w, причем мост и ребро (w,v) выбираем только в крайнем случае, если других возможностей выбора ребра не существует. Присваиваем этому ребру номер k + 1 и вычеркиваем его из списка ребер.

Алгоритм заканчивает работу, если мы попали в вершину v, и нет ни одного не пройденного ребра, инцидентного этой вершине.

Пример: эйлеров цикл: $(v_1,v_3,v_4,v_5,v_3,v_2,v_5,v_6,v_2,v_1)$.

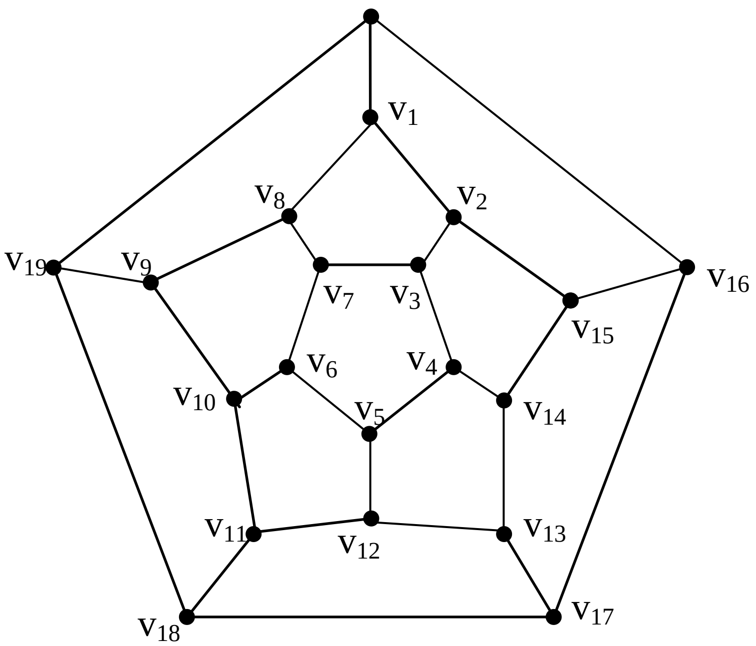


4.3 ГАМИЛЬТОНОВЫ ГРАФЫ

Простой **цикл** называется **гамильтоновым**, если он содержит каждую вершину графа. **Гамильтонова цепь** – простая цепь, содержащая каждую вершину исходного графа.

Граф называется **гамильтоновым**, если он содержит гамильтонов цикл.

В 1859 г. ирландский математик У. Гамильтон предложил игру, которая называется «Кругосветное путешествие». Каждой из двадцати вершин додекаэдра приписано название одного крупного города мира. Требовалось, переходя от одного города к другому, посетить каждый город ровно один раз и вернуться в исходный город.



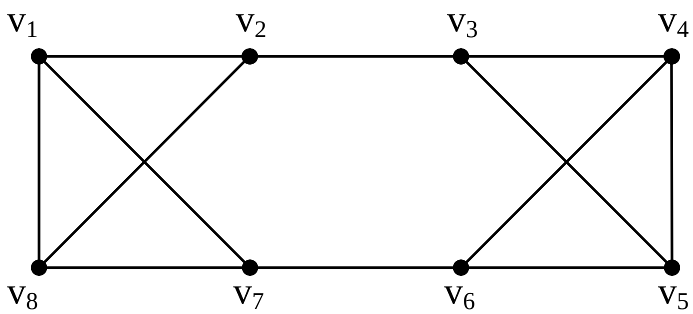
До сих пор не найдено эффективного критерия, отвечающего на вопрос, есть ли в графе гамильтонов цикл. Решается задача способом полного перебора (в ярусном представлении дерева). Поиск эффективных алгоритмов для решения данной задачи – одно из популярных направлений современной теории графов.

4.4 ДОСТАТОЧНЫЕ УСЛОВИЯ СУЩЕСТВОВАНИЯ ГАМИЛЬТОНОВА ЦИКЛА В ГРАФЕ

1. **Теорема Г. Дирака (1952 г.).** Если число вершин графа $p \geq 3$ и для любой вершины v выполняется условие: $\deg v \geq \frac{p}{2}$, то граф G - гамильтонов.

2. **Теорема О. Оре (1960 г.).** Если число вершин графа $p \geq 3$ и для любых двух несмежных вершин u и v выполняется неравенство: $\deg u + \deg v \geq p$, то граф G - гамильтонов.

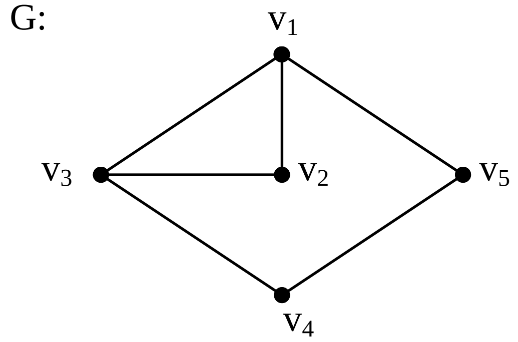
Пример:



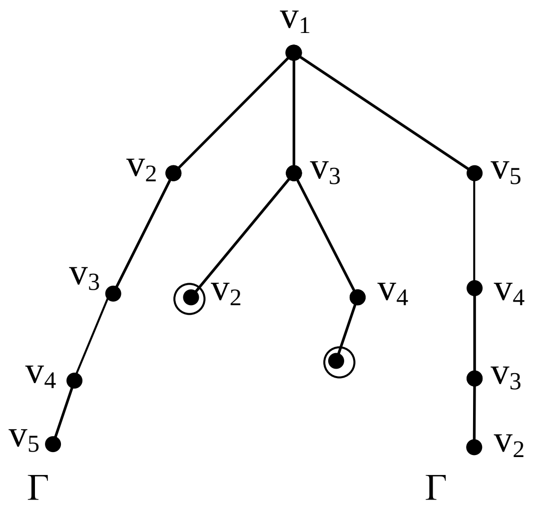
$p = 8$;
$\deg v_i = 3$;
$3 \not\geq \frac{8}{2} = 4$.

Данный граф гамильтонов, в нем существует гамильтонов цикл $M=(v_1,v_2,v_3,v_4,v_5,v_6,v_7,v_8,v_1)$, но граф не удовлетворяет условиям Оре-Дирака.

Пример:



Построим дерево полного перебора:



$(v_1,v_2,v_3,v_4,v_5,v_1)$,
$(v_1,v_5,v_4,v_3,v_2,v_1)$ –
гамильтоновы циклы

4.5 АЛГОРИТМ ПЕРЕБОРА РОБЕРТСА - ФЛОРЕСА

(алгоритм поиска гамильтонова цикла)**.**

Выбирается некоторая начальная вершина графа v_1. Эта вершина образует первый элемент множества S: $S = \{v_1\}$. Множество S на каждом шаге будет хранить уже найденные вершины гамильтоновой цепи. К S добавляется первая вершина (пусть это вершина a) в столбце v_1. Затем к S добавляется первая возможная вершина в столбце a. Пусть это вершина b, затем к S добавляется первая возможная вершина (например, c) в столбце b и т.д.

$S = \{ v_1, a, b, c, \ldots\}$

Под «возможной» понимается вершина, которой нет в S.

Пусть множество S на шаге r имеет вид: $S = \{ v_1, a, b, c, \ldots, v_{r-1}, v_r\}$.

Существует **две причины**, препятствующих включению некоторой вершины во множество S.

1. В столбце v_r нет возможной вершины (множество S нельзя расширить).
2. Цепь, определеляемая последовательностью вершин в множестве S, имеет длину p-1, где p – количество вершин графа, т. е. она является гамильтоновой цепью.

Во втором случае тоже 2 варианта:

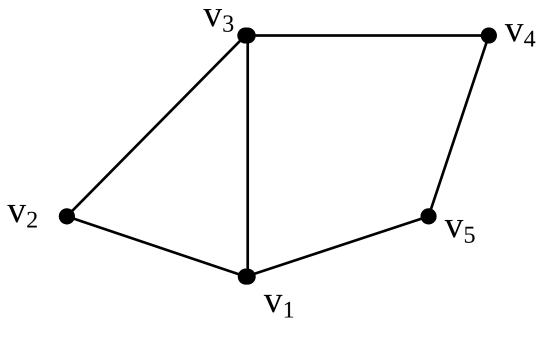
а) в графе G существует ребро (v_r,v_1), следовательно, найден гамильтонов цикл;

б) ребро (v_r,v_1) не существует, следовательно, гамильтонов цикл не может быть получен.

В случаях 1) и 2б) следует прибегнуть к возвращению, а в случае 2а) прекратить поиск (если необходимо найти только один гамильтонов цикл) или (если нужны все гамильтоновы циклы) прибегнуть к возвращению. Возвращение состоит в удалении последней включенной вершины из S, после чего множество S примет вид: S = {v_1, a, …, v_{r-1}} и добавлении к S первой возможной вершины, следующей за v_r в столбце v_{r-1}. Если не существует никакой возможной вершины, то делается следующий шаг возвращения и т. д.

Поиск заканчивается, когда множество S состоит из одной вершины v_1 и не существует никакой возможной вершины, которую можно добавить в S, так что шаг возвращения делает S пустым. Это значит, что все гамильтоновы циклы найдены. Алгоритм заканчивает работу.

Пример:



		∨			
	1	2	3	4	5
1	0	1	1	0	1
2	1	0	1	0	0
3	1	1	0	1	0
4	0	0	1	0	1
5	1	0	0	1	0

“2”

1) S = {1}
2) S = {1, 2}
3) S = {1, 2, 3}
4) S = {1, 2, 3, 4}
5) S = {1, 2, 3, 4, 5} – Г
6) S = {1, 2, 3, 4}
7) S = {1, 2, 3}
8) S = {1, 2}
9) S = {1}

$4 \to 3$
$4 \to 5$

$3 \to 1$
$3 \to 2$
$3 \to 4$

$2 \to 3$

“3”

10) S = {1,3}
11) S = {1,3,2}
12) S = {1,3}
13) S = {1,3,4}
14) S = {1,3,4,5}
15) S = {1,3,4}
16) S = {1,3}
17) S = {1}

$3 \to 1$
$3 \to 2$

$2 \to 1$
$2 \to 3$

$3 \to 4$

$4 \to 5$

$5 \to$ нет

“5”

18) S = {1, 5}
19) S = {1, 5, 4}
20) S = {1, 5, 4, 3}
21) S = {1, 5, 4, 3, 2} – Г
22) S = {1, 5, 4, 3}
23) S = {1, 5, 4}
24) S = {1, 5}
25) S = {1}
26) S = ∅

4.6 ЗАДАЧА КОММИВОЯЖЕРА И ЗАДАЧА КИТАЙСКОГО ПОЧТАЛЬОНА

Задача коммивояжера – это обобщение задачи о поиске гамильтонова цикла в графе.

Постановка задачи. В полном взвешенном графе G построить маршрут (цикл, цепь) минимального веса, проходящий по крайней мере один раз через каждую вершину исходного графа.

Если такой маршрут существует, то задача сводится к задаче поиска гамильтонова цикла.

Задача китайского почтальона - это обобщение задачи о поиске эйлерового цикла в графе.

Постановка задачи. Построить маршрут (цикл, цепь) с минимальным суммарным весом, проходящий по каждому ребру графа.

4.7 ОСНОВЫ ЦИКЛОМАТИКИ

Рассмотрим (p,q)-граф, имеющий k компонент связности.

Число ν(G) = q – p + k называется **цикломатическим числом,** или **циклическим рангом**; число ν^*(G) = p – k называется **коциклическим рангом.**

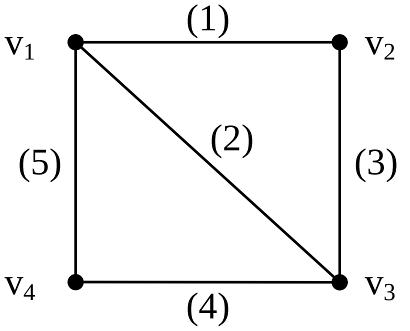
Теорема. Количество ребер остова (ветвей) произвольного графа G равно его коциклическому рангу, а количество ребер, которые необходимо удалить для построения остова (хорд), равно его циклическому рангу.

Каждому циклу графа ставится в соответствии вектор длиной q, где q – количество ребер графа G.

$z_c = (z_1, z_2, \ldots, z_q)$, где

$$z_i = \begin{cases} 1, \text{если i - е ребро} \in \text{циклу;} \\ 0, \text{иначе.} \end{cases}$$

Пример:



C1 = (v_1, v_2, v_3, v_4, v_1);

z_{C1} = (1, 0, 1, 1, 1);

C2 = (v_1, v_2, v_3, v_1);

z_{C2} = (1, 1, 1, 0, 0);

и т. д.

Множество всех векторов, каждый из которых соответствует одному циклу графа G, образует векторное пространство, которое называется **пространством циклов графа G.**

Пусть z, $z^{(1)}$, …, $z^{(k)}$ – некоторые вектора пространства, тогда если

$z = \sum_{i=1}^{k} \varepsilon_i * z^{(i)}$, где $\varepsilon_i \in \{0, 1\}$, а $\sum$ – есть операция сложения по mod 2,

то вектор z - **линейная комбинация векторов** $z^{(1)}, \ldots, z^{(k)}$.

Векторы z_1, z_2, …, z_k называются **линейно независимыми**, если никакой вектор этого множества не может быть представлен в виде линейной комбинации остальных.

Векторы z_1, z_2, …, z_k образуют **базис** в векторном пространстве, если:

1) они линейно независимы;
2) любой вектор этого пространства можно представить в виде линейной комбинации остальных векторов.

Циклы, соответствующие базисным векторам, образуют **базис цикла** графа G и называются **базисными,** или **фундаментальными**.

Теорема Эйлера. Число базисных векторов (циклов) постоянно и равно его цикломатическому числу (циклическому рангу):

V(G) = q – p + k.

Следствие 1. Если V(G) = 0, то граф G является ациклическим.

Следствие 2. Если V(G) = 1, то граф содержит один единственный простой цикл.

Следствие 3. V(G) ≥ 0.

Базисной системой циклов G называется множество всех циклов графа, каждый из которых содержит ровно одну хорду для произвольного заданного остова T. Эта система векторов образует базис, поскольку каждый из этих циклов содержит ребро (хорду), не принадлежащее ни одному из остальных циклов, следовательно, они линейно независимы, и все остальные циклы могут быть выражены в виде линейной комбинации этих циклов.

Алгоритм нахождения базисных циклов.

1. Строится произвольный остов T графа G.
2. Выписывается множество хорд относительно остова T.
3. Добавляется одна из хорд к остову; в новом получается графе единственный простой цикл, образованный этой хордой и ребрами остова (по определению дерева).
4. Повторяется пункт 3 до тех пор, пока множество хорд не будет исчерпано.

4.8 МАТРИЦА ЦИКЛОВ

Матрицей циклов графа G называется прямоугольная матрица $C = \|C_{ij}\|$,

$i = \overline{1,k}$, $j = \overline{1,q}$:

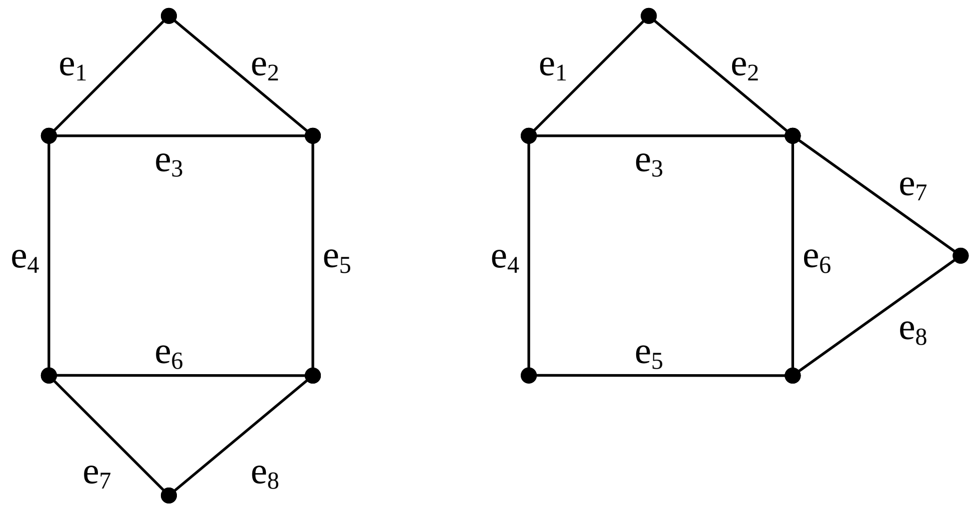
$$C_{ij} = \begin{cases} 1, \text{если } i \text{ - й цикл содержит } j \text{ - е ребро;} \\ 0, \text{иначе,} \end{cases}$$

где q – количество ребер, k – количество простых циклов в графе G.

<u>Замечание.</u> В отличие о матрицы смежности и матрицы инцидентности матрица циклов не задает граф с точностью до изоморфизма.

Пример:

G_1: G_2:



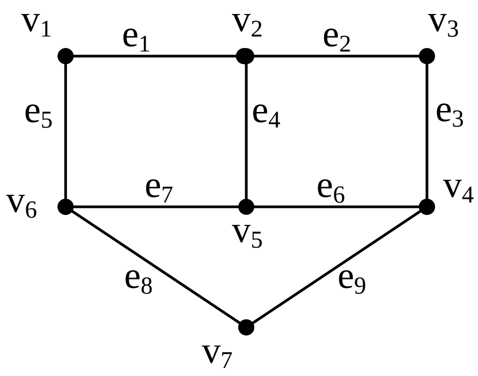
$G_1 \neq G_2$ $C(G_1) = C(G_2)$

4.9 МАТРИЦА БАЗИСНЫХ ЦИКЛОВ

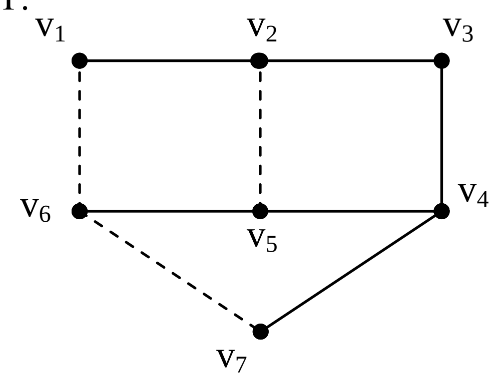
Матрица базисных(фундаментальных) циклов – это подматрица матрицы циклов:

$C_{\nu(G)} = \|C_{ij}\|$, $i = \overline{1,\nu(G)}$, $j = \overline{1,q}$.

Пример:



T:



p=7
q=9
k=1

$\nu(G) = q - p + k = 9 - 7 + 1 = 3$ – число хорд остова T (базисных, или фундаментальных, циклов графа G);

$\nu^*(G) = p - k = 7 - 1 = 6$ – число ребер остова T.

Матрица базисных циклов графа G:

	e_1	e_2	e_3	e_4	e_5	e_6	e_7	e_8	e_9
Z_{C_1}	0	1	1	1	0	1	0	0	0
Z_{C_2}	1	1	1	0	1	1	1	0	0
Z_{C_3}	0	0	0	0	0	1	1	1	1

Хорды :
(v_2, v_5);
(v_1, v_6);
(v_6, v_7).

Матрица циклов графа G:

	e_1	e_2	e_3	e_4	e_5	e_6	e_7	e_8	e_9	
Z_{C_1}	0	1	1	1	0	1	0	0	0	$C_1 = (v_2, v_3, v_4, v_5, v_2)$;
Z_{C_2}	1	1	1	0	1	1	1	0	0	$C_2 = (v_1, v_2, v_3, v_4, v_5, v_6, v_1)$;
Z_{C_3}	0	0	0	0	0	1	1	1	1	$C_3 = (v_4, v_5, v_6, v_7, v_4)$;
$Z_{C_4} = Z_{C_1} \oplus Z_{C_2}$	1	0	0	1	1	0	1	0	0	$C_4 = (v_1, v_2, v_5, v_6, v_1)$;
$Z_{C_5} = Z_{C_1} \oplus Z_{C_3}$	0	1	1	1	0	0	1	1	1	$C_5 = (v_2, v_3, v_4, v_7, v_6, v_5, v_2)$;
$Z_{C_6} = Z_{C_2} \oplus Z_{C_3}$	1	1	1	0	1	0	0	1	1	$C_6 = (v_1, v_2, v_3, v_4, v_7, v_6, v_1)$;
$Z_{C_7} = Z_{C_1} \oplus Z_{C_2} \oplus Z_{C_3}$	1	0	0	1	1	1	0	1	1	$C_7 = (v_1, v_2, v_5, v_4, v_7, v_6, v_1)$.

Замечание. Следует отметить, что хотя число базисных циклов постоянно и равно ν(G), сами циклы определены неоднозначно и зависят от певоначально выбранного остова.

4.10 КОНТРОЛЬНЫЕ ВОПРОСЫ

1. Определение эйлерова цикла. Привести пример эйлерова графа.
2. Алгоритм Флери построения эйлерова цикла.
3. Определение гамильтонова цикла. Привести пример графа, удовлетворяющего достаточным условиям гамильтоновости.
4. Идея алгоритма Робертса-Флореса поиска гамильтонова цикла.
5. Задача коммивояжера и задача китайского почтальона.
6. Определение циклического и коциклического рангов графа.
7. Количество базисных циклов графа.
8. Матрица циклов и матрица базисных (фундаментальных) циклов. Определения и примеры построения для произвольного графа.

5 ОРИЕНТИРОВАННЫЕ ГРАФЫ

5.1 ОПРЕДЕЛЕНИЕ И СПОСОБЫ ЗАДАНИЯ

Пусть имеется непустое множество V ($V \neq \varnothing$) и пусть V^2 – его декартов квадрат, тогда **орграфом** называется пара множеств $G = (V, A)$, где $A \subseteq V^2$.

Элементы множества V называются **вершинами**, элементы множества A – **дугами** орграфа.

Дуга (***u,v***) – это упорядоченная пара вершин ***u*** *и* ***v***, где ***u*** – начало дуги, ***v*** – конец дуги:

a=(u,v)

Матричные способы задания орграфа.

Пусть задан орграф $G = (V, A)$, $|V| = p$, $|A| = q$.

1. **Матрица смежности** (непосредственной достижимости):

$A = \|a_{ij}\|$, $i,j = \overline{1,p}$, где

$$a_{ij} = \begin{cases} 1, (i, j) \in A (\exists (i, j) - \text{дуга}); \\ 0, \text{иначе.} \end{cases}$$

Матрица не является симметричной.

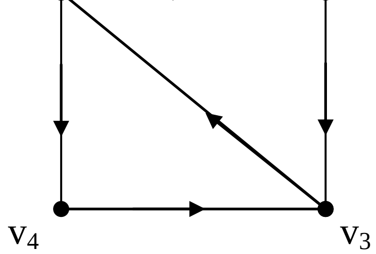
2. **Матрица инцидентности**:

$B = \|b_{ij}\|$, $i = \overline{1,p}$, $j = \overline{1,q}$, где

$$b_{ij} = \begin{cases} 1, \text{если вершина } i \text{ является началом дуги } j; \\ -1, \text{если вершина } i \text{ является концом дуги } j; \\ 0, \text{если вершина } i \text{ и дуга } j \text{ не инцидентны }. \end{cases}$$

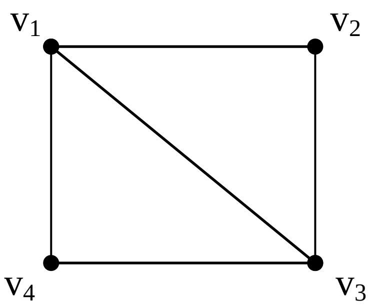
Неориентированный граф, полученный из орграфа G в результате снятия ориентации дуг орграфа, называется **основанием орграфа**.

Обратный граф $\mathbf{G^{-1} = (V, E^{-1})}$ – это граф, заданный на том же множестве вершин, у которого все дуги переориентированы, т. е.: $\exists (i,j) \in A \Rightarrow (j, i) \in A^{-1}$.

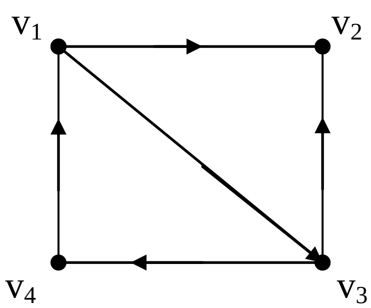


Пример:

основание:



обратный:



Орграф G1 называется **симметричным**, если для любой дуги $(i,j)\in A$ существует дуга $(j,i)\in A$.

Турниром называется орграф, основание которого – полный граф.

Полустепень исхода вершины v орграфа G есть число дуг, исходящих из этой вершины:

$\deg^{+} v = d^{+}v = |\{(u, v)|, v, u \in V, (v, u) \in A\}|$.

Полустепень захода вершины v орграфа G есть число дуг, входящих в данную вершину.

$\deg^{-} v = d^{-}v = |\{u, v)|v, u \in V, (u, v) \in A\}|$.

Степенью вершины v орграфа G называется:

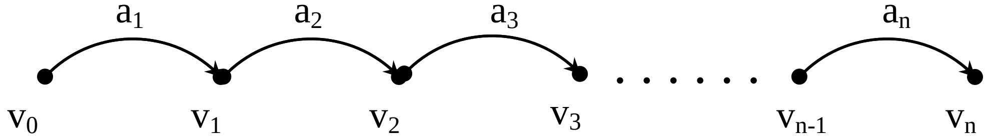
$\deg v = \deg^{+}v + \deg^{-}v$.

Аналог леммы о рукопожатии для орграфа: сумма полустепеней исхода всех вершины орграфа G равна сумме полустепеней захода всех его вершин и равна числу дуг орграфа G:

$$\sum_{v\in V}\deg^{+}v = \sum_{v\in V}\deg^{-}v = |A|.$$

5.2 МАРШРУТЫ И СВЯЗНОСТЬ

Ориентированным маршрутом называется конечная чередующаяся последовательность вершин и дуг орграфа G такая, что каждая дуга исходит из предыдущей вершины маршрута и заходит в по-



следующую:

Обозначается: $M = (v_0, a_1, v_1, a_2, \dots, a_n, v_n)$, где $a_i = (v_{i-1}, v_i)$, $i,j = \overline{1,n}$.

Цепь – это ориентированный маршрут без повторяющихся дуг.

Путь – это цепь без повторяющихся вершин (аналог простой цепи для неорграфа).

Циклический маршрут – это замкнутая цепь.

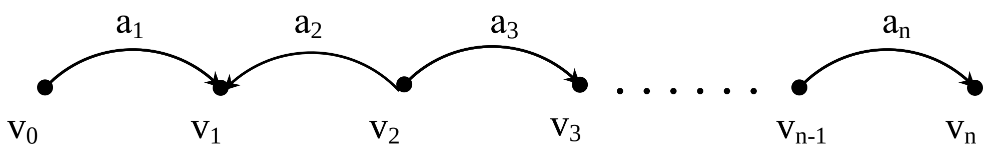
Контур – это замкнутый путь.

Длина маршрута – количество дуг, образующих данный маршрут, причем каждая дуга считается столько раз, сколько раз она входит в маршрут.

Если существует (u, v)-маршрут, то вершина v – **достижима** из вершины u, и вершина u контрдостижима для вершины v, если существует (v, u)–маршрут.

Если вершины u и v достижимы друг для друга, то они называются **взаимодостижимыми.**

Полумаршрутом в орграфе G называется маршрут его основания:



$a_i = (v_{i-1}, v_i)$, или $a_i = (v_i, v_{i-1})$ $i,j = \overline{1,n}$.

5.3 ТИПЫ СВЯЗНОСТИ ОРГРАФА

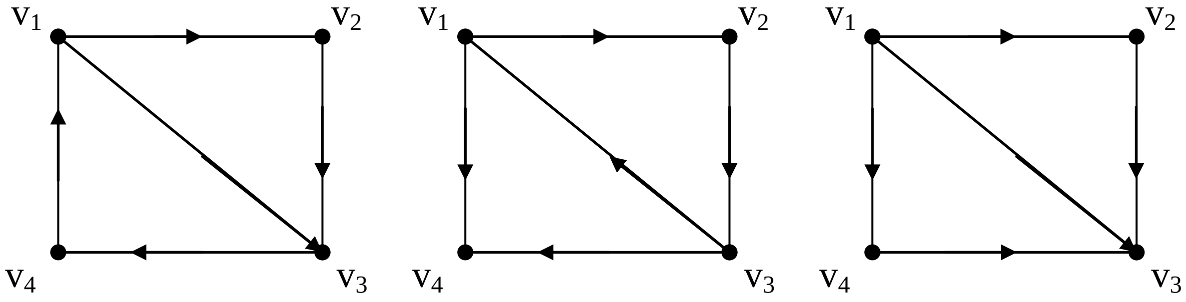
Орграф называется **сильным (сильно связанным)**, если любые две его вершины достижимы друг для друга.

Орграф называется **односторонним (односторонне связанным)**, если для каждой пары его вершин, по крайней мере, одна достижима из другой.

Орграф называется **слабым (слабо связанным)**, если любые две его вершины соединены полумаршрутом.

Пример:

сильный: односторонний: слабый:



Орграф называется **несвязным**, если его основание не связно.

Замечание. Определения подграфов и порожденных подграфов для ориентированных графов аналогичны соответствующим определениям для графов неориентированных.

5.4 ТЕОРЕМЫ О СВЯЗНОСТИ ОРГРАФА

Теорема 1. Орграф является **сильносвязным**, тогда и только тогда, когда в нем существует остовный циклический маршрут **(остовным маршрутом** называется маршрут, проходящий через все вершины графа).

Теорема 2. Орграф является односторонним тогда и только тогда, когда в нем существует остовный маршрут.

Теорема 3. Орграф является слабым тогда и только тогда, когда в нем существует остовный полумаршрут.

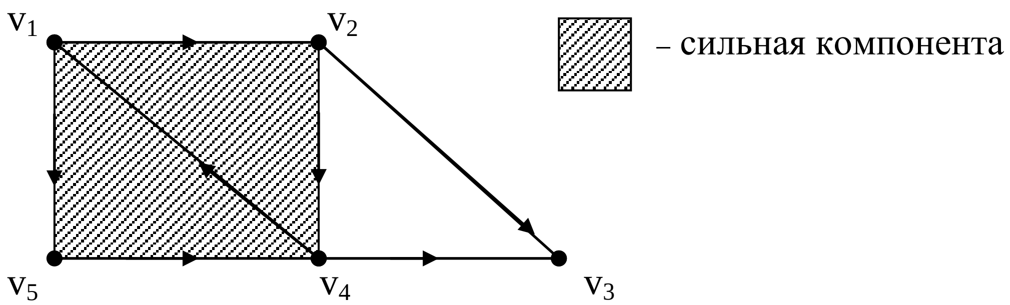
5.5 ТИПЫ КОМПОНЕНТ СВЯЗНОСТИ

Сильная компонента – это максимальный (по включению) сильно связанный подграф исходного графа.

Односторонняя компонента – это максимальный (по включению) односторонне связанный подграф исходного графа.

Слабая компонента - это максимальный (по включению) слабый подграф исходного графа.

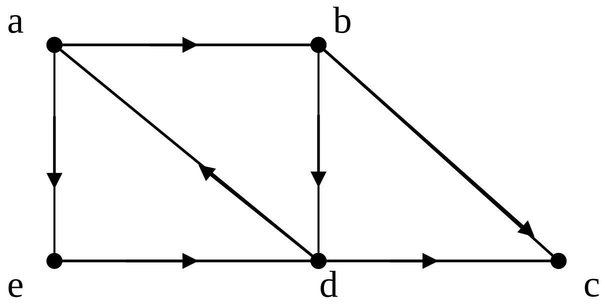
Пример:



Поскольку отношение взаимной достижимости двух вершин орграфа является отношением эквивалентности (рефлексивно, симметрично и транзитивно), то оно разбивает множество всех вершин на классы эквивалентности, т. е. такие подмножества вершин орграфа, что подграфы, порожденные этими подмножествами вершин, являются сильными компонентами.

Конденсацией называется такой орграф G*, вершины которого соответствуют сильным компонентам $S_1, S_2, \ldots, S_k$ орграфа G, и пара (S_i, S_j) в G* является дугой тогда и только тогда, когда в орграфе G существует хотя бы одна дуга, начало которой принадлежит компоненте S_i, конец - S_j.

Пример:



$S_1 = \{a, e, d, b\}$
$S_2 = \{c\}$

G*:

5.6 АЛГОРИТМ ПОСТРОЕНИЯ КОНДЕНСАЦИИ

1. Построить матрицу достижимости:

$R = \|r_{ij}\|$, $i, j = \overline{1, p}$, где

$$r_{ij} = \begin{cases} 1, \text{если существует} (j, i) \text{- маршрут} \in G; \\ 0, \text{иначе или, если } i \text{ достижима из } j. \end{cases}$$

2. Построить матрицу контрдостижимости:

$Q = \|q_{ij}\|$, $i, j = \overline{1, p}$, где

$$q_{ij} = \begin{cases} 1, \text{если существует} (i, j) - \text{маршрут} \in G; \\ 0, \text{иначе.} \end{cases}$$

Замечание. Матрица контрдостижимости Q может быть получена с помощью транспонирования матрицы R: $Q = R^T$.

3. Построить матрицу взаимной достижимости:

$S = \|s_{ij}\|$, i, j = $\overline{1, p}$, где

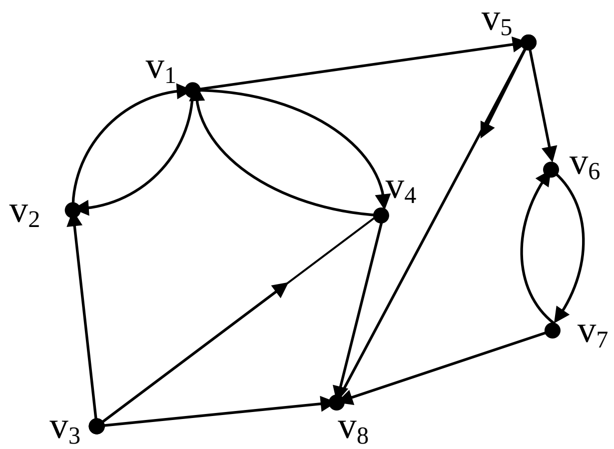
$$s_{ij} = \begin{cases} 1, \text{если существует} (i, j) \text{ и } (j, i) - \text{маршруты} \in G; \\ 0, \text{иначе.} \end{cases}$$

S = R * Q (где * – оператор поэлементного умножения матриц R и Q: $s_{ij} = r_{ij} * q_{ij}$).

4. Определить сильные компоненты: привести матрицу S к блочно-диагональному виду (перестановкой ее строк и столбцов). Вершины, составляющие каждый блок - сильные компоненты.

5. Каждой сильной компоненте поставить в соответствие одну вершину конденсации. Построить множество дуг (по определению конденсации). Полученный орграф – конденсация исходного графа.

Пример:



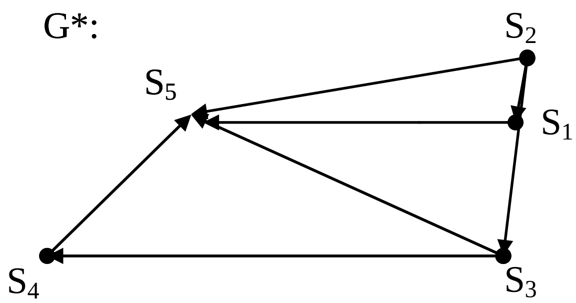
R	1	2	3	4	5	6	7	8
1	1	1	0	1	1	1	1	1
2	1	1	0	1	1	1	1	1
3	1	1	1	1	1	1	1	1
4	1	1	0	1	1	1	1	1
5	0	0	0	0	1	1	1	1
6	0	0	0	0	0	1	1	1
7	0	0	0	0	0	1	1	1
8	0	0	0	0	0	0	0	1

Q	1	2	3	4	5	6	7	8
1	1	1	1	1	0	0	0	0
2	1	1	1	1	0	0	0	0
3	0	0	1	0	0	0	0	0
4	1	1	1	1	0	0	0	0
5	1	1	1	1	1	0	0	0
6	1	1	1	1	1	1	1	0
7	1	1	1	1	1	1	1	0
8	1	1	1	1	1	1	1	1

S	1	2	3	4	5	6	7	8
1	1	1	0	1	0	0	0	0
2	1	1	0	1	0	0	0	0
3	0	0	1	0	0	0	0	0
4	1	1	0	1	0	0	0	0
5	0	0	0	0	1	0	0	0
6	0	0	0	0	0	1	1	0
7	0	0	0	0	0	1	1	0
8	0	0	0	0	0	0	0	1

Преобразуем матрицу S к блочно-диагональному виду:

S	1	2	4	3	5	6	7	8
1	1	1	1	0	0	0	0	0
2	1	1	1	0	0	0	0	0
4	1	1	1	0	0	0	0	0
3	0	0	0	1	0	0	0	0
5	0	0	0	0	1	0	0	0
6	0	0	0	0	0	1	1	0
7	0	0	0	0	0	1	1	0
8	0	0	0	0	0	0	0	1



$S_1 = \{1, 2, 4\}$
$S_2 = \{3\}$
$S_3 = \{5\}$
$S_4 = \{6,7\}$
$S_5 = \{8\}$

5.7 БАЗА И АНТИБАЗА

База орграфа G – наименьшее (относительно включения) подмножество вершин B ($B \subset V$), удовлетворяющее условию: любая вершина $v \in V \backslash B$ достижима из какой-либо вершины $w \in B$.

Антибаза орграфа G – наименьшее (относительно включения) подмножество вершин B' ($B' \subset V$) такое, что любая вершина $v \in B'$ достижима из какой- либо вершины $w \in V \backslash B'$.

5.8 АЛГОРИТМ ПОСТРОЕНИЯ БАЗЫ

В орграфе всегда существует база. Вершины, полустепени захода которых равны нулю, принадлежат базе.

База бесконтурного графа состоит только из вершин, полустепени захода которых равны нулю.

Сильные компоненты орграфа, в которые не входят дуги из других сильных компонент, соответствуют вершинам с нулевыми полустепенями захода в орграфе G* и являются **базовыми компонентами**.

Алгоритм.

1. Построить конденсацию G* орграфа G.
2. Выделить в ней базовые компоненты; им в конденсации соответствует вершины с нулевой полустепенью захода.
3. Из каждой базовой компоненты произвольно выбрать по одной вершине.

<u>Замечание.</u> База определяется не единственным образом, исключая ациклический граф.

В рассмотренном выше примере база: $B=\{v_3\}$.

5.9 АЛГОРИТМ ПОСТРОЕНИЯ АНТИБАЗЫ

1. Построить конденсацию G* орграфа G.
2. Выделить в конденсации вершины с нулевой полустепенью исхода.
3. Из соответствующих сильных компонент выбрать произвольно по одной вершине.

В рассмотренном выше примере антибаза: $B'=\{v_8\}$.

5.10 ОБХОДЫ ОРГРАФА

Определения эйлеровых и гамильтоновых циклов орграфа сходны с аналогичными определениями для неориентированного графа.

Орцепь, содержащая каждую дугу орграфа, называется **эйлеровой цепью.**

Связный орграф называется **эйлеровым графом**, если он содержит замкнутую эйлерову цепь.

Теорема. Для связного орграфа G следующие утверждения эквивалентны:

1) орграф G - эйлеров;
2) для любой вершины v орграфа G справедливо следующее равенство: $\deg^{+} v = \deg^{-} v$.
3) орграф G является объединением контуров, попарно не имеющих общих дуг.

Контур или замкнутый ормаршрут является **гамильтоновым**, если он содержит все вершины орграфа, и сам граф является гамильтоновым, если он содержит гамильтонов контур.

Теорема (достаточное условие гамильтоновости).

Пусть G – сильный орграф порядка $p > 1$ (без петель и кратных дуг). Тогда, если для любой пары его смежных вершин выполняется равенство

$\deg u + \deg v \leq 2p - 1$, то в графе G есть гамильтонов контур.

5.11 КОНТРОЛЬНЫЕ ВОПРОСЫ

1. Дать определение матрицы смежности и матрицы инцидентности орграфа.
2. Обратный орграф, основание, турнир.
3. Степени вершин орграфа.
4. Дать определение следующим понятиям: ормаршрут, цепь, путь, полумаршрут, полуцепь, полупуть, замкнутый маршрут, цикл, контур.
5. Матрицы достижимости, контрдостижимости и взаимодостижимости.
6. Типы связности орграфа, сильные компоненты.
7. Конденсация, база, антибаза. Алгоритмы построения.
8. Обходы орграфа.

6 ЗАДАНИЯ ДЛЯ ЛАБОРАТОРНЫХ РАБОТ

6.1 ЛАБОРАТОРНАЯ РАБОТА №1

Тема: «Подграфы. Изоморфизм».
Цель работы: изучение основных понятий теории графов, приобретение практических навыков определения изоморфизмов графа; построение подграфов, независимых множеств и клик.

Содержание работы:
1. Используя алгоритм генерации варианта GV, построить неориентированный граф G: GV(7,{2,3}).
2. Описать граф матрицей смежности, матрицей инцидентности. Изобразить графически граф G и его дополнение $\overline{G}$. Построить произвольный остовный подграф и подграф, порожденный множеством вершин {1,2,5,6,7}.
3. Построить все помеченные 5-графы, изоморфно вложимые в граф G. Среди них определить классы изоморфных графов, построив биекцию их вершин. Для каждого класса изоморфных графов привести рисунок абстрактного графа.
4. Найти все максимальные и наибольшие независимые множества исходного графа. Определить число независимости.
5. Найти все максимальные и наибольшие клики данного графа. Определить плотность графа G.
6. Найти полный двудольный подграф $K_{p,q}$, изоморфно вложимый в граф G с максимальным количеством вершин p+q ($p\neq1$).
7. Найти звезду $K_{1,n}$, изоморфно вложимую в граф G, с максимальным значением n.

6.2 ЛАБОРАТОРНАЯ РАБОТА №2

Тема: «Маршруты и связность».
Цель работы: изучение понятий связных графов и маршрутов, приобретение практических навыков построения матрицы расстояний и нахождения матрицы кратчайших маршрутов.

Содержание работы:
1. Используя алгоритм генерации варианта GV, построить неориентированный граф G1: GV(13,{6,7}) и граф G2: GV(7,{2,3}). Ребра графа G2 взвешены соответствующими элементами матрицы Y.
2. Определить, является ли граф G1 связным.
3. Для максимальной компоненты графа G1 выделить:
   a) открытый маршрут, не являющийся цепью;
   b) замкнутый маршрут, не являющийся циклом;

c) цепь, не являющуюся простой цепью;
d) простую цепь;
e) цикл, не являющийся простым циклом;
f) простой цикл;
g) определить обхват и окружение;
h) найти вершинную и реберную связность.

4. Для каждой компоненты графа G1:
   a) построить матрицу расстояний;
   b) определить эксцентриситеты вершин, радиус, диаметр, центр, периферию;
   c) выделить блоки;
   d) найти точки сочленения и мосты.
5. В графе G2:
   a) построить кратчайшие маршруты от произвольной вершины ко всем остальным при помощи алгоритма Дейкстры;
   b) построить кратчайшие маршруты от произвольной вершины ко всем остальным при помощи алгоритма Форда;
   c) построить кратчайшие маршруты при помощи алгоритма Флойда. При построении вести две матрицы – матрицу маршрутов и матрицу расстояний.

6.3 ЛАБОРАТОРНАЯ РАБОТА №3

Тема 1: «Деревья и остовы».
Цель работы*:* изучение понятий деревьев и остовов, приобретение практических навыков построения матрицы Кирхгофа и вычисления количества помеченных остовов.

Содержание работы:
1. Используя алгоритм генерации варианта GV, построить неориентированный граф G1: GV(5,{2,3}) и граф G2: GV(13,{6,7}). Ребра графа G2 взвешены соответствующими элементами матрицы Y.
2. Для графа G1 составить матрицу Кирхгофа и посчитать количество помеченных остовов.
3. Для графа G2:
   a) построить дерево обхода вершин графа в ширину и в глубину;
   b) решить задачу построения остовов кратчайших маршрутов, используя алгоритмы Прима и Краскала (в качестве весов ребер использовать элементы матрицы Y);
4. Сгенерировать все различные абстрактные не изоморфные друг другу деревья порядка 4 и 7. Разделить множество деревьев на два подмножества: с одной и двумя центральными вершинами.

Тема 2: «Циклы и обходы».
Цель работы: изучение понятий гамильтоновых и эйлеровых циклов, приобретение практических навыков построения матрицы циклов и матрицы фундаментальных циклов.

Содержание работы:

1. Используя алгоритм генерации варианта GV, построить неориентированный граф G1: GV(5,{2,3}) и граф G2: GV(13,{6,7}).
2. Для произвольного остова графа G1 построить матрицу фундаментальных циклов. Посчитать циклический и коциклический ранг, выразить 3 непростых цикла (если таковые имеются) через минимальную комбинацию базисных.
3. Определить, являются ли графы G1 и G2 эйлеровыми, построить эйлеровы циклы по алгоритму Флёри, эйлеровы цепи. Если граф не эйлеров, добавить минимальное число ребер, делающих его эйлеровым.
4. Определить, является ли граф G2 гамильтоновым, построить гамильтонов цикл, используя алгоритм Робертса-Флореса. Если граф не является гамильтоновым, то добавить минимальное число ребер, делающих его гамильтоновым.

6.4 ЛАБОРАТОРНАЯ РАБОТА №4

Тема: «Ориентированные графы».
Цель работы: изучение основных понятий ориентированных графов, приобретение практических навыков построения матрицы смежности, матрицы инцидентности; определение типов связности орграфа; построение конденсации орграфа.

Содержание работы:

1. Использую алгоритм генерации варианта GV1, построить ориентированный граф G: GV1(9,{6,7}).
2. Построить матрицу смежности и матрицу инцидентности заданного орграфа.
3. Построить основание и обратный граф. Определить, является ли граф симметричным.
4. Построить ормаршрут, цепь, путь, полумаршрут, полуцепь, полупуть, замкнутый маршрут, цикл и контур.
5. Построить матрицу достижимости, контрдостижимости, взаимной достижимости. Представить ограниченно достижимую матрицу для числа достижимости, равного 2.
6. Определить тип связности орграфа, выделить сильные компоненты.
7. Построить конденсацию. Определить базы и антибазы.

6.5 АЛГОРИТМЫ ГЕНЕРАЦИИ ВАРИАНТОВ

GV(p,x): A[1..p,1..p], где
p – количество вершин в графе;
x – параметр генерации варианта;
A – матрица смежности графа.

Алгоритм генерации варианта GV(p,x): A[1..p,1..p]:

Статический локальный параметр S = <фамилия> <имя> <отчество> .
Функция n(c) - номер буквы в алфавите (1..33).

1. Вычеркнуть из S все повторные вхождения букв.
2. Построить $Y = \|y_{ij}\|$, $i,j = \overline{1,p}$, где
   $y_{ij} = |n(S_i)-n(S_j)|$
3. Построить $A = \|a_{ij}\|$, $i,j = \overline{1,p}$:
   если i=j, то a_{ij}=0, иначе:
   $$a_{ij} = \begin{cases} 1, \text{если существует } x \in X,\ y_{ij} \bmod x = 0; \\ 0, \text{иначе.} \end{cases}$$
4. Для каждой изолированной (доминирующей) вершины добавить (удалить) одно ребро. Добавляемое (удаляемое) ребро связывает текущую вершину со следующей (по номеру). Для последней вершины следующей считать первую вершину.

Алгоритм генерации варианта GV1(p,x): A[1..p,1..p]:

Статический локальный параметр S = <фамилия> <имя> <отчество> .
Функция n(c) - номер буквы в алфавите (1..33).
1. Вычеркнуть из S все повторные вхождения букв.
2. Построить $Y = \|y_{ij}\|$, $i,j = \overline{1,p}$, где верхний треугольник матрицы заполняется по формуле $y_{ij} = |n(S_i)-n(S_j)|$, а нижний - $y_{ij} = n(S_i)+n(S_j)$.
3. Построить $A = \|a_{ij}\|$, $i,j = \overline{1,p}$, :
   если i=j, то a_{ij}=0, иначе:
   $$a_{ij} = \begin{cases} 1, \text{если существует } x \in X,\ y_{ij} \bmod x = 0; \\ 0, \text{иначе.} \end{cases}$$

7 ПРИМЕРЫ ВЫПОЛНЕНИЯ ЛАБОРАТОРНЫХ РАБОТ

7.1 ЛАБОРАТОРНАЯ РАБОТА №1

Тема: «Подграфы. Изоморфизм».
Цель работы: изучение основных понятий теории графов и приобретение практических навыков определения изоморфизмов графа; построение подграфов, независимых множеств и клик.

1. ***Используя алгоритм генерации варианта* GV*, построить неориентированный граф* G*:* GV(7,{2,3}) .**

S=<юфаеленаяковлевна>;
S=<ю,ф,а,е,л,н,я,к,о,в>;
$n(S_i)$={32,22,1,6,13,15,33}.

Y	32	22	1	6	13	15	33
32	0	10	31	26	29	17	1
22	10	0	21	16	9	7	11
1	31	21	0	5	12	14	32
6	26	16	5	0	7	9	27
13	29	9	12	7	0	2	20
15	17	7	14	9	2	0	18
33	1	11	32	27	20	18	0

A	1	2	3	4	5	6	7
1	0	1	0	1	0	0	0
2	1	0	1	1	1	0	0
3	0	1	0	0	1	1	1
4	1	1	0	0	0	1	1
5	0	1	1	0	0	1	1
6	0	0	1	1	1	0	1
7	0	0	1	1	1	1	0

Построим неориентированный граф G1, используя способ перечисления. G=(V,E):

V={1,2,3,4,5,6,7} – множество вершин графа;

E={(1,2),(1,4),(2,3),(2,4),(2,5),(3,5),(3,6),(3,7),(4,6),(4,7),(5,6), (5,7),(6,7)} - множество ребер графа.

2. ***Описать граф матрицей смежности, матрицей инцидентности. Изобразить графически граф G и его дополнение $\overline{G}$. Построить произвольный остовный подграф и подграф, порожденный вершинами {1,2,5,6,7}.***

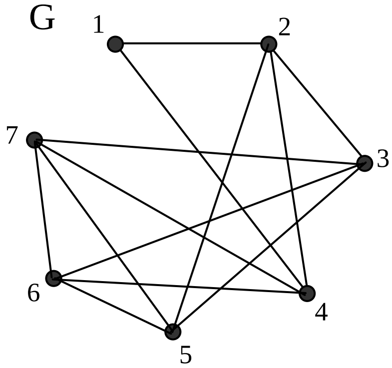
Матрица смежности

A	1	2	3	4	5	6	7
1	0	1	0	1	0	0	0
2	1	0	1	1	1	0	0
3	0	1	0	0	1	1	1
4	1	1	0	0	0	1	1
5	0	1	1	0	0	1	1
6	0	0	1	1	1	0	1
7	0	0	1	1	1	1	0

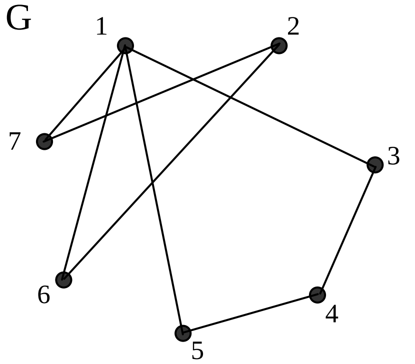
Матрица инцидентности

B	(1,2)	(1,4)	(2,3)	(2,4)	(2,5)	(3,5)	(3,6)	(3,7)	(4,6)	(4,7)	(5,6)	(5,7)	(6,7)
1	1	1	0	0	0	0	0	0	0	0	0	0	0
2	1	0	1	1	1	0	0	0	0	0	0	0	0
3	0	0	1	0	0	1	1	1	0	0	0	0	0
4	0	1	0	1	0	0	0	0	1	1	0	0	0
5	0	0	0	0	1	1	0	0	0	0	1	1	0
6	0	0	0	0	0	0	1	0	1	0	1	0	1
7	0	0	0	0	0	0	0	1	0	1	0	1	1

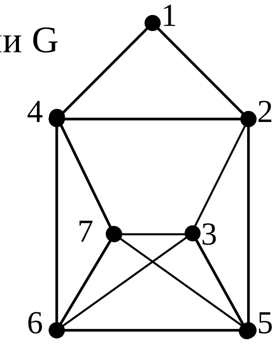
G



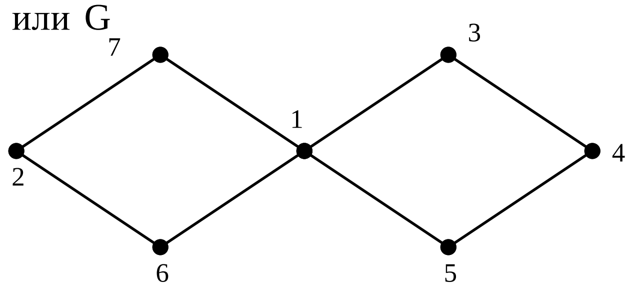
$\overline{G}$



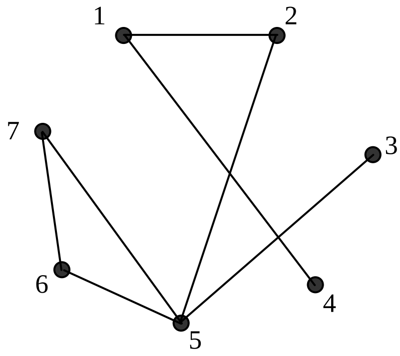
или G



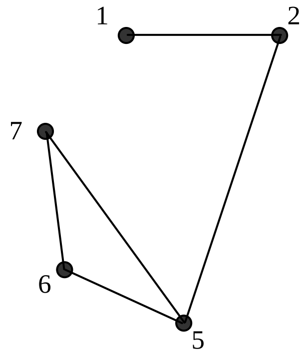
или $\overline{G}$



Остовный подграф G1=(V,E1), где E1={(1,2),(1,4),(2,5),(3,5),(5,6),(5,7),(6,7)}:



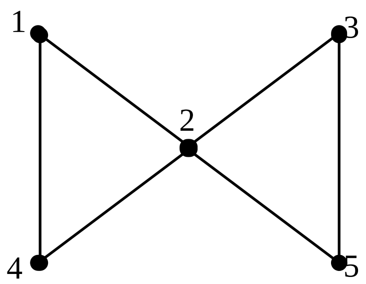
Подграф G2, порожденный вершинами {1,2,5,6,7}:



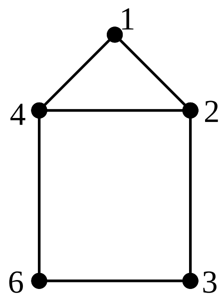
3. ***Построить все помеченные 5-графы, изоморфно вложимые в граф G. Среди них определить классы изоморфных графов, построив биекцию их вершин. Для каждого класса изоморфных графов привести рисунок абстрактного графа.***

5-графы, изоморфно вложимые в граф G:

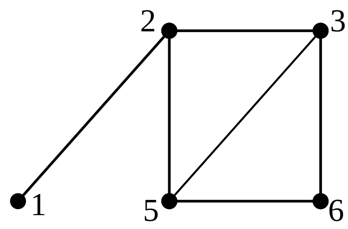
G_1:



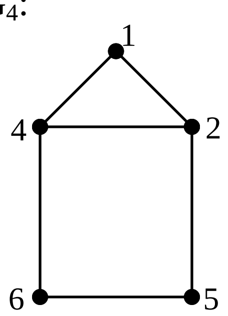
G_2:



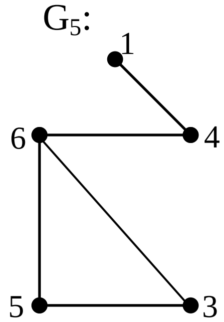
G_3:



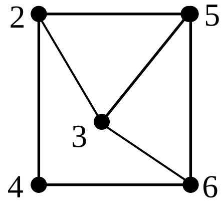
G_4:



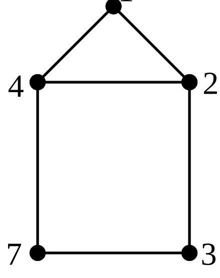
G_5:



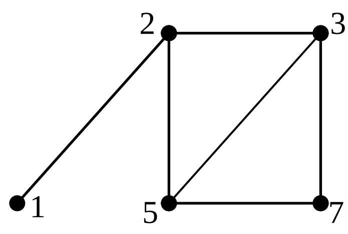
G_6:



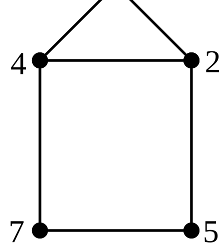
G_7:



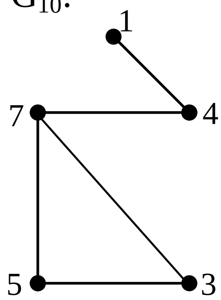
G_8:



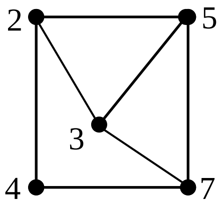
G_9:



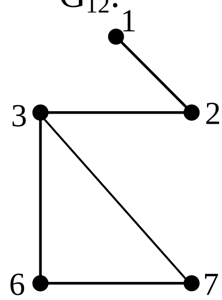
G_{10}:



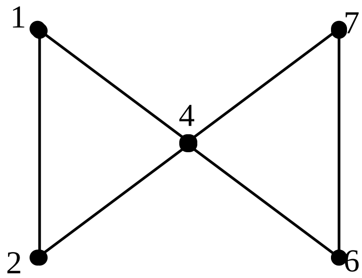
G_{11}:



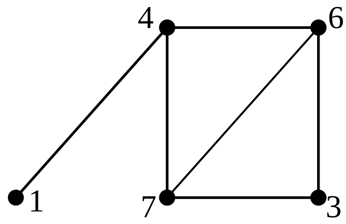
G_{12}:



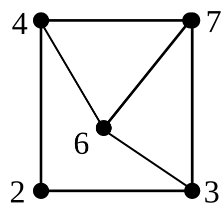
G_{13}:



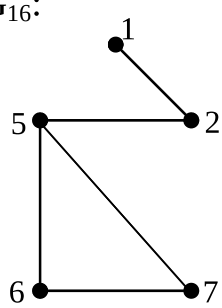
G_{14}:



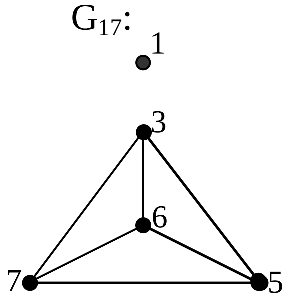
G_{15}:



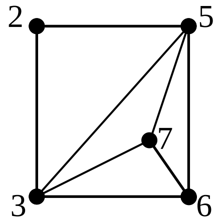
G_{16}:



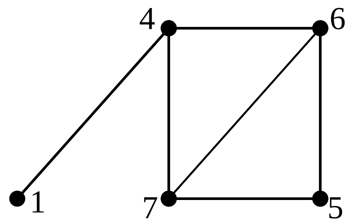
G_{17}:



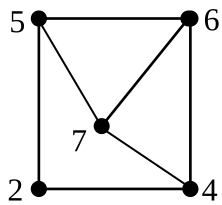
G_{18}:



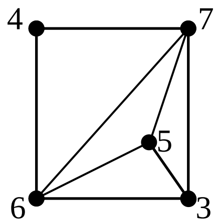
G_{19}:



G_{20}:



G_{21}:



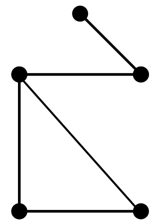
Классы изоморфных графов:

1) Графы с пятью ребрами: G_5, G_{10}, G_{12}, G_{16}.

Биекция вершин:

G_i	v1	v2	v3	v4	v5
G_5	1	4	6	3	5
G_{10}	1	4	7	3	5
G_{12}	1	2	3	7	6
G_{16}	1	2	5	7	6

Рисунок абстрактного графа:



2) Графы с шестью ребрами: G_1, G_2,G_3, G4, G7, G8, G9, G13, G14, G17, G19.

Биекция вершин: Рисунок абстрактного графа:

а)

G_i	v1	v2	v3	v4	v5
G_1	1	2	3	5	4
G_{13}	1	4	7	6	2

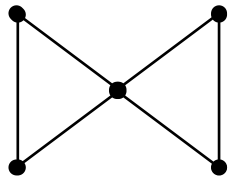


Рисунок абстрактного графа:

б)

G_i	v1	v2	v3	v4	v5
G_2	1	2	3	6	4
G_4	1	2	5	6	4
G_7	1	2	3	7	4
G_9	1	2	5	7	4

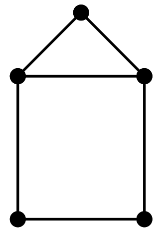


Рисунок абстрактного графа:

в)

G_i	v1	v2	v3	v4	v5
G_3	1	2	3	6	5
G_8	1	2	3	7	5
G_{14}	1	4	6	3	7
G_{19}	1	4	6	5	7

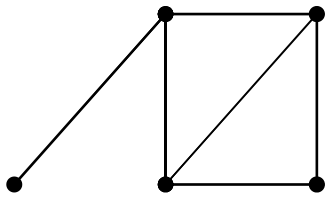
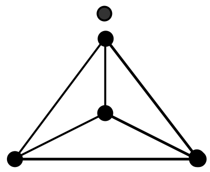


Рисунок абстрактного графа:

г)

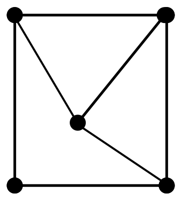
G_i	v1	v2	v3	v4	v5
G_{17}	1	3	5	6	7



3) Графы с семью ребрами: G_6, G_{11}, G_{15}, G_{20}.

Биекция вершин: Рисунок абстрактного графа:

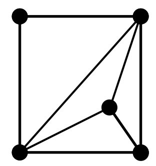
G_i	v1	v2	v3	v4	v5
G_6	2	3	5	6	4
G_{11}	2	3	5	7	4
G_{15}	4	6	7	3	2
G_{20}	5	7	6	4	2



4) Графы с восемью ребрами:

Биекция вершин: Рисунок абстрактного графа:

G_i	v1	v2	v3	v4	v5
G_{18}	2	5	6	7	3
G_{21}	4	7	3	5	6



4. ***Найти все максимальные и наибольшие независимые множества исходного графа. Определить число независимости.***

Выпишем все независимые множества данного графа G:
S1={1,3}; S2={1,5}; S3={1,6}; S4={1,7};
S5={2,6}; S6={2,7}; S7={3,4}; S8={4,5}.
Для данного графа максимальными независимыми множествами являются S1, S2, S3, S4, S5, S6 ,S7, S8.
Для данного графа наибольшими независимыми являются множества S1, S2, S3, S4, S5, S6 ,S7, S8.
Число независимости графа $\alpha(G)=|S1|=2$.

5. ***Найти все максимальные и наибольшие клики данного графа. Определить плотность графа* G.**

Выпишем все клики данного графа:

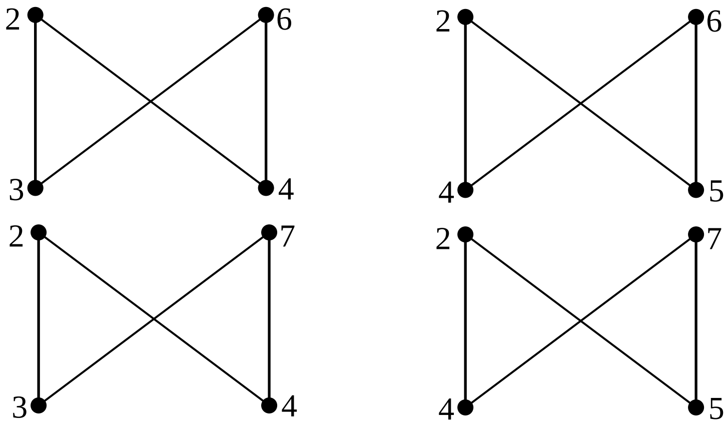
K1={1,2}; K8={3,7}; K15={2,3,5};
K2={1,4}; K9={4,6}; K16={3,5,6};
K3={2,3}; K10={4,7}; K17={3,6,7};
K4={2,4}; K11={5,6}; K18={4,6,7};
K5={2,5}; K12 ={5,7}; K19={5,6,7};
K6={3,5}; K13={6,7}; K20={3,5,7};
K7={3,6}; K14={1,2,4}; K21={3,5,6,7}.

Для данного графа максимальными кликами являются K14, K15, K18, K21; наибольшей кликой является K21.

Плотность графа $\beta(G)=|K21|=4$.

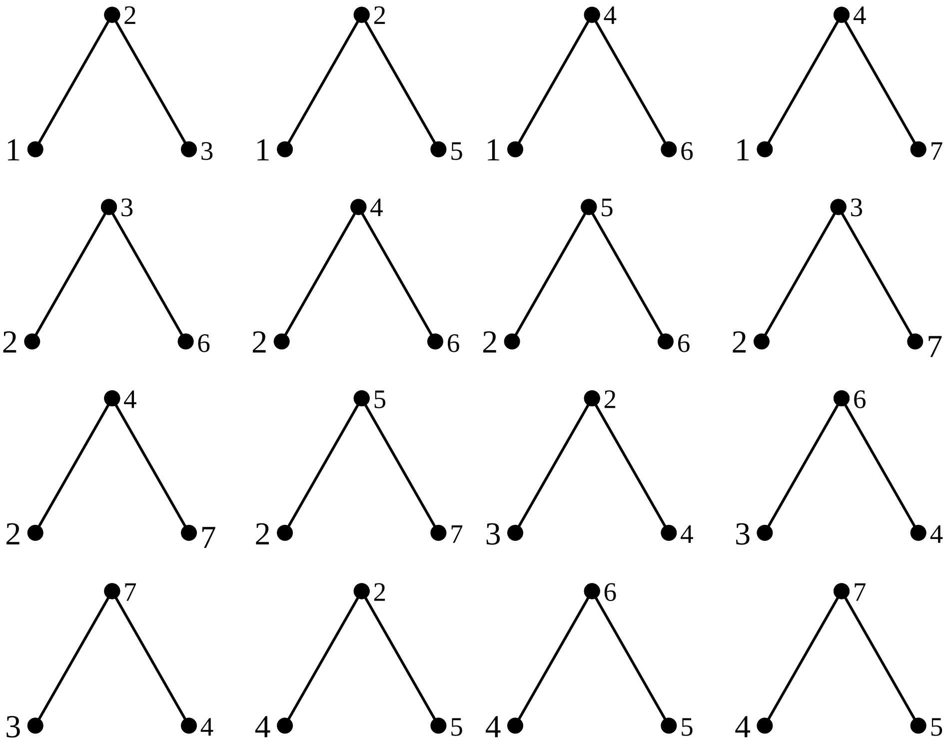
6. ***Найти полный двудольный подграф* $\mathbf{K_{p,q}}$*, изоморфно вложимый в граф* G, *с максимальным количеством вершин* p+q (p≠1).**

Полные двудольные подграфы, изоморфно вложимые в G:
$K_{p,q}=K_{2,2}$; p+q=2+2=4.



7. ***Найти звезду*** $\mathbf{K_{1,n}}$***, изоморфно вложимую в граф* G *с максимальным значением* n.**

Звезды $K_{1,n}$, изоморфно вложимые в G с максимальными n:
$K_{1,n}=K_{1,2}$; n=2.



Вывод: в данной лабораторной работе изучены основные понятия теории графов, приобретены практические навыки определения изоморфизмов графа, построения подграфов, независимых множеств и клик.

7.2 ЛАБОРАТОРНАЯ РАБОТА №2

Тема: «Маршруты и связность».

Цель работы: изучение понятий связных графов и маршрутов, приобретение практических навыков построения матрицы расстояний и нахождения матрицы кратчайших маршрутов.

1. ***Используя алгоритм генерации варианта* GV*, построить неориентированный граф* G1:GV(13,{6,7}) и граф G2:GV(7,{2,3}). *Ребра графа* G2 *взвешены соответствующими элементами матрицы* Y.**

Построим граф G1, используя алгоритм генерации варианта GV(13,{6,7}).
S=<юфаеленаяковлевна>;
S=<ю,ф,а,е,л,н,я,к,о,в>;
$n(S_i)$={32,22,1,6,13,15,33,12,16,3,29,18,14}.

Y1	32	22	1	6	13	15	33	12	16	3	29	18	14
32	0	10	31	26	19	17	1	20	16	29	3	14	18
22	10	0	21	16	9	7	11	10	6	19	7	4	8
1	31	21	0	5	12	14	32	11	15	2	28	17	13
6	26	16	5	0	7	9	27	6	10	3	23	12	8
13	19	9	12	7	0	2	20	1	3	10	16	5	1
15	17	7	14	9	2	0	18	2	1	12	14	3	1
33	1	11	32	27	20	18	0	21	17	30	4	15	19
12	20	10	11	6	1	2	21	0	4	9	17	6	2
16	16	6	15	10	3	1	17	4	0	13	13	2	2
3	29	19	2	3	10	12	30	9	13	0	26	15	11
29	3	7	28	23	16	14	4	17	13	26	0	11	15
18	14	4	17	12	5	3	15	6	2	15	11	0	4
14	18	8	13	8	1	1	19	2	2	11	15	4	0

Матрица смежности

A1	1	2	3	4	5	6	7	8	9	10	11	12	13
1	0	0	0	0	0	0	0	0	0	0	0	1	1
2	0	0	1	0	0	1	0	0	1	0	1	0	0
3	0	1	0	0	1	1	0	0	0	0	1	0	0
4	0	0	0	0	1	0	0	1	0	0	0	1	0
5	0	0	1	1	0	0	0	0	0	0	0	0	0
6	0	1	1	0	0	0	1	0	0	1	1	0	0
7	0	0	0	0	0	1	0	1	0	1	0	0	0
8	0	0	0	1	0	0	1	0	0	0	0	1	0
9	0	1	0	0	0	0	0	0	0	0	0	0	0
10	0	0	0	0	0	1	1	0	0	0	0	0	0
11	0	1	1	0	0	1	0	0	0	0	0	0	0
12	1	0	0	1	0	0	0	1	0	0	0	0	0
13	1	0	0	0	0	0	0	0	0	0	0	0	0

G1=(V1,E1): V1={1,2,3,4,5,6,7,8,9,10,11,12,13} – множество вершин графа;
E1={(1,12),(1,13),(2,3),(2,6),(2,9),(2,11),(3,5),(3,6),(3,11),(4,5),
(4,8),(4,12),(6,7),(6,10),(6,11),(7,8),(7,10),(8,12)} – множество ребер графа.

Построим граф G2, используя алгоритм генерации варианта GV(7,{2,3}).

S=<юфаеленаяковлевна>;

S=<ю,ф,а,е,л,н,я,к,о,в>;

n(S_i)={32,22,1,6,13,15,33}.

Y2	32	22	1	6	13	15	33
32	0	10	31	26	29	17	1
22	10	0	21	16	9	7	11
1	31	21	0	5	12	14	32
6	26	16	5	0	7	9	27
13	29	9	12	7	0	2	20
15	17	7	14	9	2	0	18
33	1	11	32	27	20	18	0

Матрица смежности

A2	1	2	3	4	5	6	7
1	0	1	0	1	0	0	0
2	1	0	1	1	1	0	0
3	0	1	0	0	1	1	1
4	1	1	0	0	0	1	1
5	0	1	1	0	0	1	1
6	0	0	1	1	1	0	1
7	0	0	1	1	1	1	0

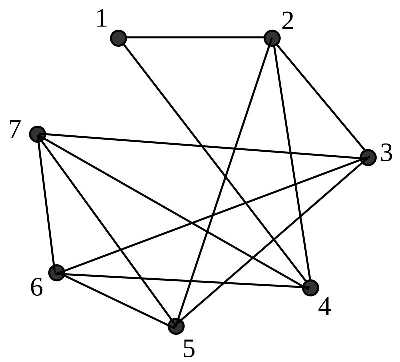
Построим неориентированный граф G2, используя способ перечисления.
G2=(V2,E2):

V2={1,2,3,4,5,6,7} – множество вершин графа;

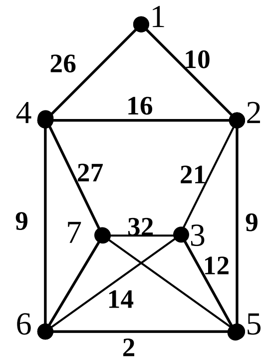
E2={(1,2),(1,4),(2,3),(2,4),(2,5),(3,5),(3,6),(3,7),(4,6),(4,7),(5,6), (5,7),(6,7)} – множество ребер графа.

Построим графическое изображение графов G1 и G2.

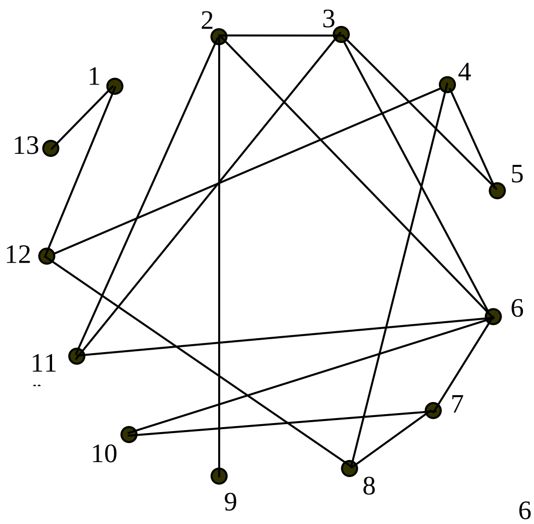
G2:



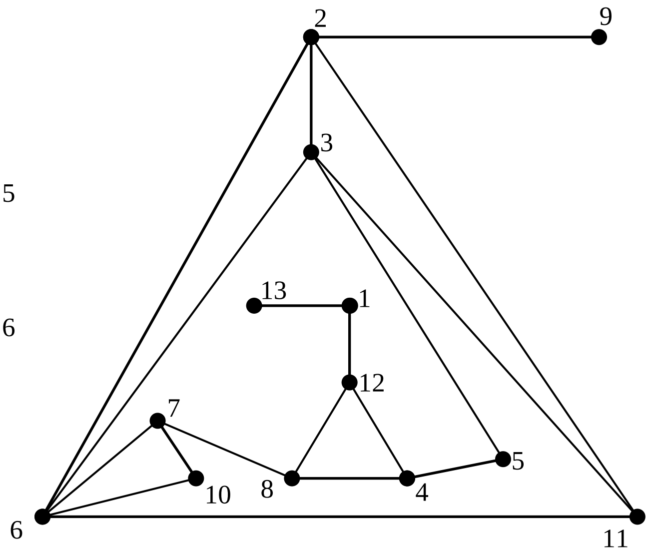
или:



G1:



или:



2. ***Определить, является ли граф* G1 *связным.***

Две вершины называются связными, если между ними существует маршрут.

Граф называется связным, если любая пара его вершин связна. Каждая вершина считается связной сама с собой.

Чтобы определить, является ли граф G1 связным, определим, существуют ли маршруты между любыми двумя его вершинами:

$M_{1,2}$={1,12,4,5,3,2}; $M_{1,3}$={1,12,4,5,3}; $M_{1,4}$={1,12,4};
$M_{1,5}$={1,12,4,5}; $M_{1,6}$={1,12,8,7,6}; $M_{1,7}$={1,12,8,7};

$M_{1,8}=\{1,12,8\}$;	$M_{1,9}=\{1,12,4,5,3,2,9\}$;	$M_{1,13}=\{1,13\}$;
$M_{1,10}=\{1,12,8,7,10\}$;	$M_{1,11}=\{1,12,4,5,3,11\}$;	$M_{1,12}=\{1,12\}$;
$M_{2,3}=\{2,3\}$;	$M_{2,4}=\{2,3,5,4\}$;	$M_{2,5}=\{2,3,5\}$;
$M_{2,6}=\{2,6\}$;	$M_{2,7}=\{2,6,7\}$;	$M_{2,8}=\{2,6,7,8\}$;
$M_{2,9}=\{2,9\}$;	$M_{2,10}=\{2,6,10\}$;	$M_{2,11}=\{2,11\}$;
$M_{2,12}=\{2,3,5,4,12\}$;	$M_{2,13}=\{2,3,5,4,12,1,13\}$	$M_{3,4}=\{3,5,4\}$;
$M_{3,5}=\{3,5\}$;	$M_{3,6}=\{3,6\}$;	$M_{3,7}=\{3,6,7\}$;
$M_{3,8}=\{3,5,4,8\}$;	$M_{3,9}=\{3,2,9\}$;	$M_{3,10}=\{3,6,10\}$;
$M_{3,11}=\{3,11\}$;	$M_{3,12}=\{3,5,4,12\}$;	$M_{3,13}=\{3,5,4,12,1,13\}$;
$M_{4,5}=\{4,5\}$;	$M_{4,6}=\{4,8,7,6\}$;	$M_{4,7}=\{4,8,7\}$;
$M_{4,8}=\{4,8\}$;	$M_{4,9}=\{4,5,3,2,9\}$;	$M_{4,10}=\{4,8,7,10\}$;
$M_{4,11}=\{4,5,3,11\}$;	$M_{4,12}=\{4,12\}$;	$M_{4,13}=\{4,12,1,13\}$;
$M_{5,6}=\{5,3,6\}$;	$M_{5,7}=\{5,3,6,7\}$;	$M_{5,8}=\{5,4,8\}$;
$M_{5,9}=\{5,3,2,9\}$;	$M_{5,10}=\{5,3,6,10\}$;	$M_{5,11}=\{5,3,11\}$;
$M_{5,12}=\{5,4,12\}$;	$M_{5,13}=\{5,4,12,1,13\}$;	$M_{6,7}=\{6,7\}$;
$M_{6,8}=\{6,7,8\}$;	$M_{6,9}=\{6,2,9\}$;	$M_{6,10}=\{6,10\}$;
$M_{6,11}=\{6,11\}$;	$M_{6,12}=\{6,7,8,12\}$;	$M_{6,13}=\{6,7,8,12,1,13\}$;
$M_{7,8}=\{7,8\}$;	$M_{7,9}=\{7,6,2,9\}$;	$M_{7,10}=\{7,10\}$;
$M_{7,11}=\{7,6,11\}$;	$M_{7,12}=\{7,8,12\}$;	$M_{7,13}=\{7,8,12,1,13\}$;
$M_{8,9}=\{8,7,6,2,9\}$;	$M_{8,10}=\{8,7,10\}$;	$M_{8,11}=\{8,7,6,11\}$;
$M_{8,12}=\{8,12\}$;	$M_{8,13}=\{8,12,1,13\}$;	$M_{9,10}=\{9,2,6,10\}$;
$M_{9,11}=\{9,2,11\}$;	$M_{9,12}=\{9,2,3,5,4,12\}$;	$M_{9,13}=\{9,2,3,5,4,12,1,13\}$;
$M_{10,11}=\{10,6,11\}$;	$M_{10,13}=\{10,7,8,12,1,13\}$;	$M_{11,12}=\{11,3,5,4,12\}$;
$M_{10,12}=\{10,7,8,12\}$;	$M_{12,13}=\{12,1,13\}$;	$M_{11,13}=\{11,3,5,4,12,1,13\}$;

Итак, между любыми двумя вершинами графа существует маршрут, следовательно, граф является связным. Он содержит только одну компоненту связности, которая совпадает с исходным графом.

3. ***Для максимальной компоненты графа* G1 *выделить:***

- ***a) открытый маршрут, не являющийся цепью;***
- ***b) замкнутый маршрут, не являющийся циклом;***
- ***c) цепь;***
- ***d) простую цепь;***
- ***e) цикл;***
- ***f) простой цикл;***
- ***g) определить обхват и окружение;***
- ***h) найти вершинную и реберную связность.***

Маршрут называется замкнутым, если его концевые вершины совпадают.

Открытый маршрут называется цепью, если все его ребра различны.

Открытый маршрут называется простой цепью, если все его вершины различны.

Маршрут называется циклом, если у него совпадают концевые вершины, и он является цепью.

Маршрут называется простым циклом, если у него совпадают концевые вершины, и он является простой цепью.

Длиной маршрута называется число ребер, содержащихся в маршруте.

Обхватом графа называется длина наименьшего простого цикла, если такой существует. Обозначается: g(G).

Окружением графа называется длина наибольшего простого цикла, если такой существует. Обозначается: C(G).

Вершинной связностью графа называется наименьшее число вершин, удаление которых приводит к несвязному графу. Обозначается: χ(G).

Реберной связностью графа называется наименьшее число ребер, удаление которых приводит к несвязному графу. Обозначается: λ(G).

В графе G1:

a) выделим открытый маршрут, не являющийся цепью:

M_1={2,**3,5,3**,6,7}=M(2,7);

b) выделим замкнутый маршрут, не являющийся циклом:

M_2={**1,12**,4,5,3,6,7,8,**12,1**}=M(1,1);

c) выделим цепь:

M_3={11,**2**,6,3,**2**,9}=M(11,9);

d) выделим простую цепь:

M_4={3,6,7,8,12,1,13}=M(3,13);

e) выделим цикл:

M_5={10,7,**6**,3,2,**6**,10}=M(10,10);

f) выделим простой цикл:

M_6={2,3,5,4,8,7,6.11,2}=M(2,2);

g) определим обхват и окружение:

g(G)=|M3|-1=3;

C(G)=|M13|-1=10;

h) найдем вершинную и реберную связность:

- если удалить вершину 2, или вершину 12, или вершину 1, то граф станет не связным, следовательно, вершинная связность данного графа χ(G)=1;

- если удалить ребро (2,9), или ребро (1,12), или ребро (1,13), то граф станет не связным, следовательно, реберная связность данного графа λ(G)=1.

4. ***Для каждой компоненты графа* G1:**

построить матрицу расстояний;

a) определить эксцентриситеты вершин;

b) радиус;

c) диаметр;

d) центр;

e) периферию;

f) выделить блоки;

g) найти точки сочленения и мосты.

Расстоянием между двумя вершинами v_i и v_j графа называется длина кратчайшей простой цепи, соединяющей эти вершины. Если вершины в графе не связаны, то расстояние полагается равным бесконечности. Обозначается: $d(v_i,v_j)$.

Матрицей расстояний D называется такая квадратная матрица размером p×p, элементы которой находятся по формуле:

$$d_{ij} = \begin{cases} d(v_i, v_j), \text{если} \exists M(i, j), i \neq j; \\ 0, \text{если} \quad i = j; \\ \infty, \text{если} \quad i \neq j. \end{cases}$$

Эксцентриситетом вершины называется длина максимальной геодезической, исходящей из этой вершины. Обозначается: e(v).

Геодезической называется кратчайшая простая цепь между вершинами. Обозначается: σ(u,v).

Радиусом графа называется минимальный среди всех эксцентриситетов. Обозначается: r(G).

Диаметром графа называется максимальный среди всех эксцентриситетов. Обозначается: d(G).

Центром графа называется такое множество его вершин, для которого радиус равен эксцентриситету.

Периферией графа называется такое множество его вершин, для которых эксцентриситет равен диаметру.

Вершина называется точкой сочленения, если ее удаление приводит к появлению в графе большего числа компонент связности.

Ребро называется мостом, если удаление его приводит к появлению большего числа компонент связности.

Граф называется неразделимым, если в нем отсутствуют точки сочленения.

Блок – это максимальный неразделимый подграф исходного графа.

Так как граф G1 связный, то он имеет только одну компоненту связности, совпадающую с самим графом. Построим матрицу расстояний графа G1:

D	1	2	3	4	5	6	7	8	9	10	11	12	13	$e(v_i)$
1	0	5	4	2	3	4	3	2	6	4	5	1	1	6
2	5	0	1	3	2	1	2	3	1	2	1	4	6	6
3	4	1	0	2	1	1	2	3	2	2	1	3	5	5
4	2	3	2	0	1	3	2	1	4	3	3	1	3	4
5	3	2	1	1	0	2	3	2	3	3	2	2	4	4
6	4	1	1	3	2	0	1	2	2	1	1	3	5	5
7	3	2	2	2	3	1	0	1	3	1	2	2	4	4
8	2	3	3	1	2	2	1	0	4	2	3	1	3	4
9	6	1	2	4	3	2	3	4	0	3	2	5	7	7
10	4	2	2	3	3	1	1	2	3	0	2	3	5	5
11	5	1	1	3	2	1	2	3	2	2	0	4	6	6
12	1	4	3	1	2	3	2	1	5	3	4	0	2	5
13	1	6	5	3	4	5	4	3	7	5	6	2	0	7

В графе G1:

a) определим эксцентриситеты вершин:

e(1)=6;	e(2)=6;	e(3)=5;	e(4)=4;	e(5)=4;
e(6)=5;	e(7)=4;	e(8)=4;	e(9)=7;	e(10)=5;
e(11)=6;	e(12)=5;	e(13)=7;		

b) определим радиус:

r(G1)=e(4)=4;

c) определим диаметр:

d(G1)=e(13)=7;

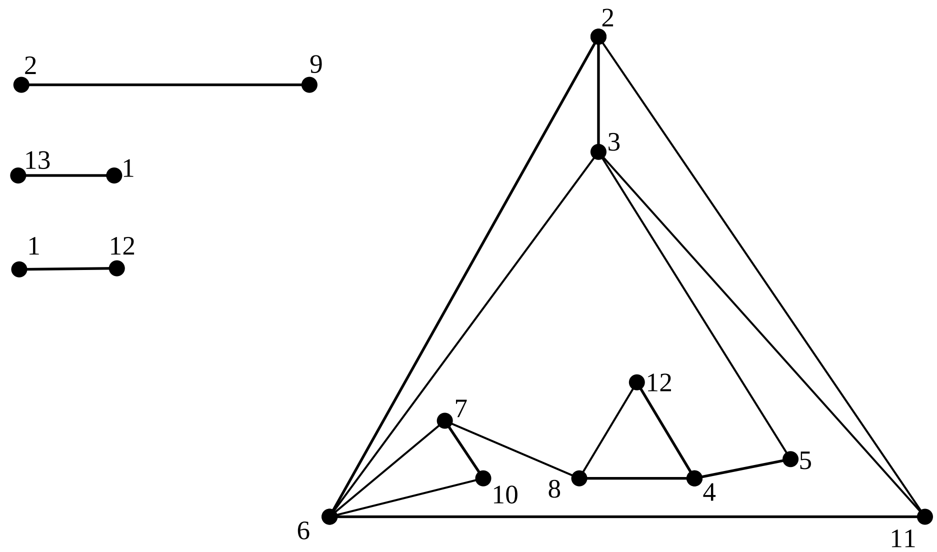
d) определим центр:

множество вершин {4,5,7,8} – центр графа;

e) определим периферию:

множество вершин {9,13} – периферия графа;

f) выделим блоки:



g) найдем точки сочленения и мосты:

если удалить вершину 2, или вершину 1, или вершину 12, то компонент связности в графе станет больше, следовательно, точками сочленения данного графа являются вершины 1, 2, 12;

если удалить ребро (2,9), или ребро (1,12), или ребро (1,13), то граф станет не связным, следовательно, мостами данного графа являются ребра (2,9), (1,12), (1,13).

5. ***В графе* G2:**

a) построить кратчайшие маршруты от произвольной вершины ко всем остальным при помощи алгоритма Дейкстры;

b) построить кратчайшие маршруты при помощи алгоритма Флойда. При построении вести две матрицы – матрицу маршрутов и матрицу расстояний.

Построим кратчайшие маршруты от вершины 5 ко всем остальным с помощью алгоритма Дейкстры.

Матрица весов:

C	1	2	3	4	5	6	7
1	0	10	∞	26	∞	∞	∞
2	10	0	21	16	9	∞	∞
3	∞	21	0	∞	12	14	32
4	26	16	∞	0	∞	9	27
5	∞	9	12	∞	0	2	20
6	∞	∞	14	9	2	0	18
7	∞	∞	32	27	20	18	0

1) $l(5)=0$; $\tilde{l}(1)=\infty$; $\tilde{l}(2)=\infty$; $\tilde{l}(3)=\infty$;
$\tilde{l}(4)=\infty$; $\tilde{l}(6)=\infty$; $\tilde{l}(7)=\infty$; $p=5$.

2) $l(5)=0$;
$\tilde{\tilde{l}}(1)=\min(\tilde{l}(1), l(5)+c(5,1))=\min(\infty, 0+\infty)=\infty$;
$\tilde{\tilde{l}}(2)=\min(\tilde{l}(2), l(5)+c(5,2))=\min(\infty, 0+9)=9$;
$\tilde{\tilde{l}}(3)=\min(\tilde{l}(3), l(5)+c(5,3))=\min(\infty, 0+12)=12$;
$\tilde{\tilde{l}}(4)=\min(\tilde{l}(4), l(5)+c(5,4))=\min(\infty, 0+\infty)=\infty$;
$\tilde{\tilde{l}}(6)=\min(\tilde{l}(6), l(5)+c(5,6))=\min(\infty, 0+2)=\mathbf{2}$;
$\tilde{\tilde{l}}(7)=\min(\tilde{l}(7), l(5)+c(5,7))=\min(\infty, 0+20)=20$;
$p=6$.

3) $l(6)=2$; $\tilde{l}(1)=\infty$; $\tilde{l}(2)=9$; $\tilde{l}(3)=12$;
$\tilde{l}(4)=\infty$; $\tilde{l}(7)=20$;
$\tilde{\tilde{l}}(1)=\min(\tilde{l}(1), l(6)+c(6,1))=\min(\infty, 2+\infty)=\infty$;
$\tilde{\tilde{l}}(2)=\min(\tilde{l}(2), l(6)+c(6,2))=\min(9, 2+\infty)=\mathbf{9}$;
$\tilde{\tilde{l}}(3)=\min(\tilde{l}(3), l(6)+c(6,3))=\min(12, 2+14)=12$;
$\tilde{\tilde{l}}(4)=\min(\tilde{l}(4), l(6)+c(6,4))=\min(\infty, 2+9)=11$;
$\tilde{\tilde{l}}(7)=\min(\tilde{l}(7), l(6)+c(6,7))=\min(20, 2+18)=20$;
$p=2$.

4) $l(2)=9$; $\tilde{l}(1)=\infty$; $\tilde{l}(3)=12$; $\tilde{l}(4)=11$;
$\tilde{l}(7)=20$;
$\tilde{\tilde{l}}(1)=\min(\tilde{l}(1), l(2)+c(2,1))=\min(\infty, 9+10)=19$;
$\tilde{\tilde{l}}(3)=\min(\tilde{l}(3), l(2)+c(2,3))=\min(12, 9+21)=12$;
$\tilde{\tilde{l}}(4)=\min(\tilde{l}(4), l(2)+c(2,4))=\min(11, 9+16)=\mathbf{11}$;
$\tilde{\tilde{l}}(7)=\min(\tilde{l}(7), l(2)+c(2,7))=\min(20, 9+\infty)=20$;
$p=4$.

5) $l(4)=11$; $\tilde{l}(1)=19$; $\tilde{l}(3)=12$; $\tilde{l}(7)=20$;

$\tilde{\tilde{l}}(1)=\min(\tilde{l}(1), l(4)+c(4,1))=\min(19, 11+26)=19$;

$\tilde{\tilde{l}}(3)=\min(\tilde{l}(3), l(4)+c(4,3))=\min(12, 11+\infty)=\mathbf{12}$;

$\tilde{\tilde{l}}(7)=\min(\tilde{l}(7), l(4)+c(4,7))=\min(20, 11+27)=20$;

p=3.

6) $l(3)=12$; $\tilde{l}(1)=19$; $\tilde{l}(7)=20$;

$\tilde{\tilde{l}}(1)=\min(\tilde{l}(1), l(3)+c(3,1))=\min(19, 12+\infty)=19$;

$\tilde{\tilde{l}}(7)=\min(\tilde{l}(7), l(3)+c(3,7))=\min(20, 12+32)=20$;

p=1.

7) $l(1)=19$; $\tilde{\tilde{l}}(7)=\min(\tilde{l}(7), l(1)+c(1,7))=\min(20, 19+\infty)=20$; p=7.

Итак, кратчайшие маршруты:

$M_{5,1}=\{5,2,1\}=19$;

$M_{5,2}=\{5,2\}=9$;

$M_{5,3}=\{5,3\}=12$;

$M_{5,4}=\{5,6,4\}=11$;

$M_{5,6}=\{5,6\}=2$;

$M_{5,7}=\{5,7\}=\{5,6,7\}=20$.

Построим кратчайшие маршруты при помощи алгоритма Флойда.

k=0

Матрица весов:

C	1	2	3	4	5	6	7
1	0	10	∞	26	∞	∞	∞
2	10	0	21	16	9	∞	∞
3	∞	21	0	∞	12	14	32
4	26	16	∞	0	∞	9	27
5	∞	9	12	∞	0	2	20
6	∞	∞	14	9	2	0	18
7	∞	∞	32	27	20	18	0

Матрица путей:

Q	1	2	3	4	5	6	7
1	1	1	1	1	1	1	1
2	2	2	2	2	2	2	2
3	3	3	3	3	3	3	3
4	4	4	4	4	4	4	4
5	5	5	5	5	5	5	5
6	6	6	6	6	6	6	6
7	7	7	7	7	7	7	7

k=1

Матрица весов:

C	1	2	3	4	5	6	7
1	0	10	∞	26	∞	∞	∞
2	10	0	21	16	9	∞	∞
3	∞	21	0	∞	12	14	32
4	26	16	∞	0	∞	9	27
5	∞	9	12	∞	0	2	20
6	∞	∞	14	9	2	0	18
7	∞	∞	32	27	20	18	0

Матрица путей:

Q	1	2	3	4	5	6	7
1	1	1	1	1	1	1	1
2	2	2	2	2	2	2	2
3	3	3	3	3	3	3	3
4	4	4	4	4	4	4	4
5	5	5	5	5	5	5	5
6	6	6	6	6	6	6	6
7	7	7	7	7	7	7	7

k=2

Матрица весов:

C	1	2	3	4	5	6	7
1	0	10	31	26	19	∞	∞
2	10	0	21	16	9	∞	∞
3	31	21	0	37	12	14	32
4	26	16	37	0	25	9	27
5	19	9	12	25	0	2	20
6	∞	∞	14	9	2	0	18
7	∞	∞	32	27	20	18	0

Матрица путей:

Q	1	2	3	4	5	6	7
1	1	1	**2**	1	**2**	1	1
2	2	2	2	2	2	2	2
3	**2**	3	3	**2**	3	3	3
4	4	4	**2**	4	**2**	4	4
5	**2**	5	5	**2**	5	5	5
6	6	6	6	6	6	6	6
7	7	7	7	7	7	7	7

k=3

Матрица весов:

C	1	2	3	4	5	6	7
1	0	10	31	26	19	45	63
2	10	0	21	16	9	35	53
3	31	21	0	37	12	14	32
4	26	16	37	0	25	9	27
5	19	9	12	25	0	2	20
6	45	35	14	9	2	0	18
7	63	53	32	27	20	18	0

Матрица путей:

Q	1	2	3	4	5	6	7
1	1	1	2	1	2	**3**	**3**
2	2	2	2	2	2	**3**	**3**
3	2	3	3	2	3	3	3
4	4	4	2	4	2	4	4
5	2	5	5	2	5	5	5
6	**3**	**3**	6	6	6	6	6
7	**3**	**3**	7	7	7	7	7

k=4

Матрица весов:

C	1	2	3	4	5	6	7
1	0	10	31	26	19	35	53
2	10	0	21	16	9	25	43
3	31	21	0	37	12	14	32
4	26	16	37	0	25	9	27
5	19	9	12	25	0	2	20
6	35	25	14	9	2	0	18
7	53	43	32	27	20	18	0

Матрица путей:

Q	1	2	3	4	5	6	7
1	1	1	2	1	2	**4**	**4**
2	2	2	2	2	2	**4**	**4**
3	2	3	3	2	3	3	3
4	4	4	2	4	2	4	4
5	2	5	5	2	5	5	5
6	**4**	**4**	6	6	6	6	6
7	**4**	**4**	7	7	7	7	7

5) k=5

Матрица весов:

C	1	2	3	4	5	6	7
1	0	10	31	26	19	21	39
2	10	0	21	16	9	11	29
3	31	21	0	37	12	14	32
4	26	16	37	0	25	9	27
5	19	9	12	25	0	2	20
6	21	11	14	9	2	0	18
7	39	29	32	27	20	18	0

Матрица путей:

Q	1	2	3	4	5	6	7
1	1	1	2	1	2	**5**	**5**
2	2	2	2	2	2	**5**	**5**
3	2	3	3	2	3	3	3
4	4	4	2	4	2	4	4
5	2	5	5	2	5	5	5
6	**5**	**5**	6	6	6	6	6
7	**5**	**5**	7	7	7	7	7

6) k=6

Матрица весов:

C	1	2	3	4	5	6	7
1	0	10	31	26	19	21	39
2	10	0	21	16	9	11	29
3	31	21	0	23	12	14	32
4	26	16	23	0	11	9	27
5	19	9	12	11	0	2	20
6	21	22	14	9	2	0	18
7	39	29	32	27	20	18	0

Матрица путей:

Q	1	2	3	4	5	6	7
1	1	1	2	1	2	5	5
2	2	2	2	2	2	5	5
3	2	3	3	**6**	3	3	3
4	4	4	**6**	4	**6**	4	4
5	2	5	5	**6**	5	5	5
6	5	5	6	6	6	6	6
7	5	5	7	7	7	7	7

7) k=7

Матрица весов:

C	1	2	3	4	5	6	7
1	0	10	31	26	19	21	39
2	10	0	21	16	9	11	29
3	31	21	0	23	12	14	32
4	26	16	23	0	11	9	27
5	19	9	12	11	0	2	20
6	21	22	14	9	2	0	18
7	39	29	32	27	20	18	0

Матрица путей:

Q	1	2	3	4	5	6	7
1	1	1	2	1	2	5	5
2	2	2	2	2	2	5	5
3	2	3	3	6	3	3	3
4	4	4	6	4	6	4	4
5	2	5	5	6	5	5	5
6	5	5	6	6	6	6	6
7	5	5	7	7	7	7	7

Итак, кратчайшие маршруты между всеми парами вершин:

$M_{1,2}$={1,2};	$M_{1,3}$={1,2,3};	$M_{1,4}$={1,4};
$M_{1,5}$={1,2,5};	$M_{1,6}$={1,2,5,6};	$M_{1,7}$={1,2,5,7};
$M_{2,3}$={2,3};	$M_{2,4}$={2,4};	$M_{2,5}$={2,5};
$M_{2,6}$={2,5,6};	$M_{2,7}$={2,5 ,7};	$M_{3,4}$={3,6,4};
$M_{3,5}$={3,5};	$M_{3,6}$={3,6};	$M_{3,7}$={3,7};
$M_{4,5}$={4,6,5};	$M_{4,6}$={4,6};	$M_{4,7}$={4,7};
$M_{5,6}$={5,6};	$M_{5,7}$={5,7};	$M_{6,7}$={6,7};

Вывод: в данной лабораторной работе изучены понятия связных графов и маршрутов, приобретены практические навыки построения матрицы расстояний и нахождения матрицы кратчайших маршрутов.

7.3 ЛАБОРАТОРНАЯ РАБОТА №3

Тема 1: «Деревья и остовы».
Цель работы: изучение понятий деревьев и остовов, приобретение практических навыков построения матрицы Кирхгофа и вычисления количества помеченных остовов.

1. ***Используя алгоритм генерации варианта*** **GV,** ***построить неориентированный граф*** **G1: GV(5,{2,3})** ***и граф*** **G2: GV(13,{6,7}).** ***Ребра графа*** **G2** ***взвешены соответствующими элементами матрицы*** **Y.**

Построим граф G1, используя алгоритм генерации варианта GV(5,{2,3}).
S=<юфаеленаяковлевна>;
S=<ю,ф,а,е,л,н,я,к,о,в>;
$n(S_i)$={32,22,1,6,13}.

Y1	32	22	1	6	13
32	0	10	31	26	29
22	10	0	21	16	9
1	31	21	0	5	12
6	26	16	5	0	7
13	29	9	12	7	0

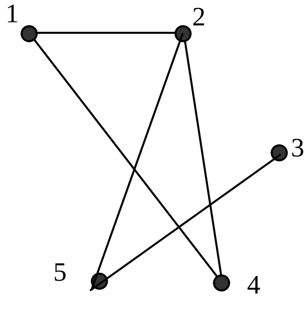
Матрица смежности

A1	1	2	3	4	5
1	0	1	0	1	0
2	1	0	0	1	1
3	0	0	0	0	1
4	1	1	0	0	0
5	0	1	1	0	0

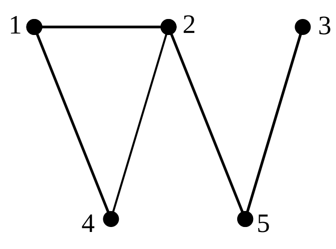
Так как вершина 2 доминирующая, то ребро (2,3) удалено**.**

Построим неориентированный граф G1, используя способ перечисления.
G1=(V1,E1): V1={1,2,3,4,5} – множество вершин графа;
E1={(1,2),(1,4),(2,4),(2,5),(3,5)} – множество ребер графа.
Построим графическое изображение графа G1.

G1:



или:



Построим граф G2, используя алгоритм генерации варианта GV(13,{6,7}).
S=<юфаеленаяковлевна>;
S=<ю,ф,а,е,л,н,я,к,о,в>;
n(S_i)={32,22,1,6,13,15,33,12,16,3,29,18,14}.

Y2	32	22	1	6	13	15	33	12	16	3	29	18	14
32	0	10	31	26	19	17	1	20	16	29	3	14	18
22	10	0	21	16	9	7	11	10	6	19	7	4	8
1	31	21	0	5	12	14	32	11	15	2	28	17	13
6	26	16	5	0	7	9	27	6	10	3	23	12	8
13	19	9	12	7	0	2	20	1	3	10	16	5	1
15	17	7	14	9	2	0	18	2	1	12	14	3	1
33	1	11	32	27	20	18	0	21	17	30	4	15	19
12	20	10	11	6	1	2	21	0	4	9	17	6	2
16	16	6	15	10	3	1	17	4	0	13	13	2	2
3	29	19	2	3	10	12	30	9	13	0	26	15	11
29	3	7	28	23	16	14	4	17	13	26	0	11	15
18	14	4	17	12	5	3	15	6	2	15	11	0	4
14	18	8	13	8	1	1	19	2	2	11	15	4	0

Матрица смежности

A2	1	2	3	4	5	6	7	8	9	10	11	12	13
1	0	0	0	0	0	0	0	0	0	0	0	1	1
2	0	0	1	0	0	1	0	0	1	0	1	0	0
3	0	1	0	0	1	1	0	0	0	0	1	0	0
4	0	0	0	0	1	0	0	1	0	0	0	1	0
5	0	0	1	1	0	0	0	0	0	0	0	0	0
6	0	1	1	0	0	0	1	0	0	1	1	0	0
7	0	0	0	0	0	1	0	1	0	1	0	0	0
8	0	0	0	1	0	0	1	0	0	0	0	1	0
9	0	1	0	0	0	0	0	0	0	0	0	0	0
10	0	0	0	0	0	1	1	0	0	0	0	0	0
11	0	1	1	0	0	1	0	0	0	0	0	0	0
12	1	0	0	1	0	0	0	1	0	0	0	0	0
13	1	0	0	0	0	0	0	0	0	0	0	0	0

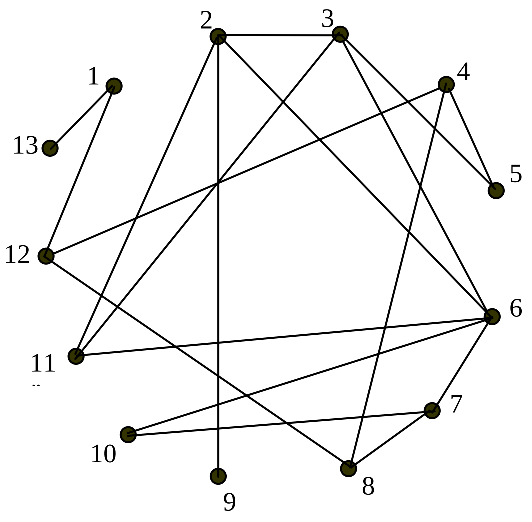
Построим неориентированный граф G2, используя способ перечисления.
G2=(V2,E2):

V2={1,2,3,4,5,6,7,8,9,10,11,12,13} – множество вершин графа;

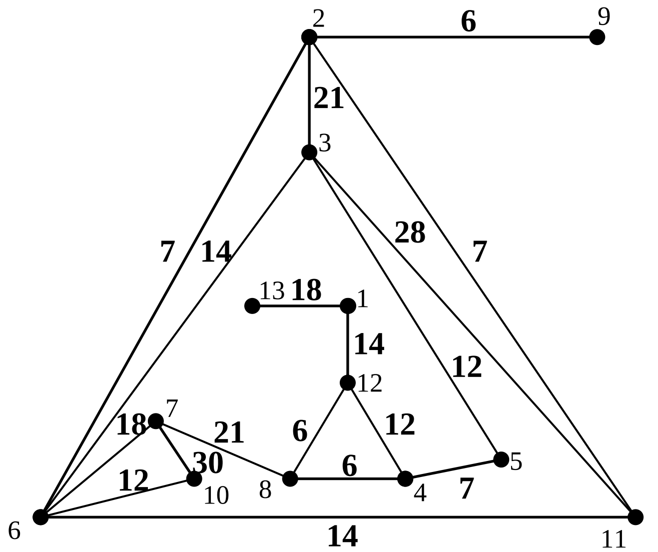
E2={(1,12),(1,13),(2,3),(2,6),(2,9),(2,11),(3,5),(3,6),(3,11),(4,5),(4,8),(4,12),(6,7),(6,10),(6,11),(7,8),(7,10),(8,12)} – множество ребер графа.

Построим графическое изображение графа G2.

G2:



или:



2. ***Для графа* G1 *составить матрицу Кирхгофа и посчитать количество помеченных остовов.***

Матрица Кирхгофа

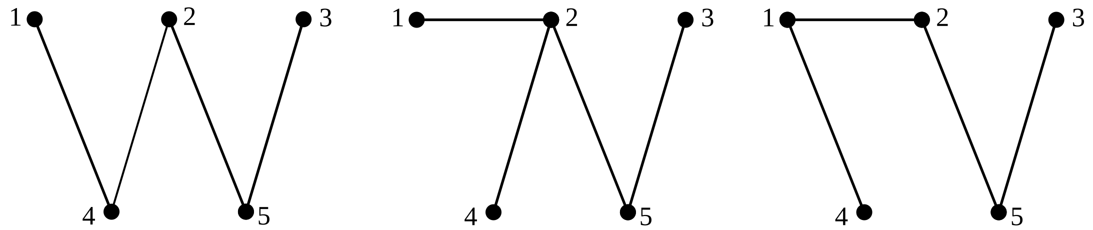
M	1	2	3	4	5
1	2	-1	0	-1	0
2	-1	3	0	-1	-1
3	0	0	1	0	-1
4	-1	-1	0	2	0
5	0	-1	-1	0	2

Количество помеченных остовов k графа G равно алгебраическому дополнению любого элемента матрицы Кирхгофа.

$$k = \begin{vmatrix} 3 & 0 & -1 & -1 \\ 0 & 1 & 0 & -1 \\ -1 & 0 & 2 & 0 \\ -1 & -1 & 0 & 2 \end{vmatrix} = \begin{vmatrix} 0 & -3 & -1 & 5 \\ 0 & 1 & 0 & -1 \\ 0 & 1 & 2 & -2 \\ -1 & -1 & 0 & 2 \end{vmatrix} = (-1)\cdot(-1)\cdot \begin{vmatrix} -3 & -1 & 5 \\ 1 & 0 & -1 \\ 1 & 2 & -2 \end{vmatrix} =$$

$$= \begin{vmatrix} 2 & -1 & 5 \\ 0 & 0 & -1 \\ -1 & 2 & -2 \end{vmatrix} = (-1)\cdot(-1)\cdot \begin{vmatrix} 2 & -1 \\ -1 & 2 \end{vmatrix} = 4 - 1 = 3.$$

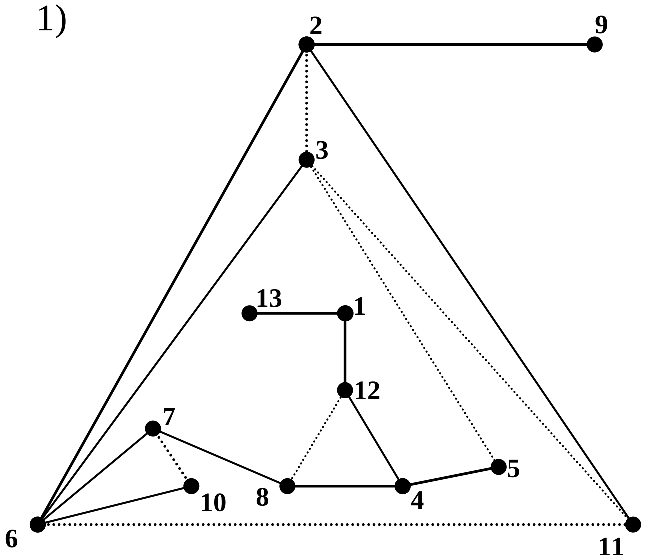
Итак, в графе G1 содержится 3 помеченных остова:



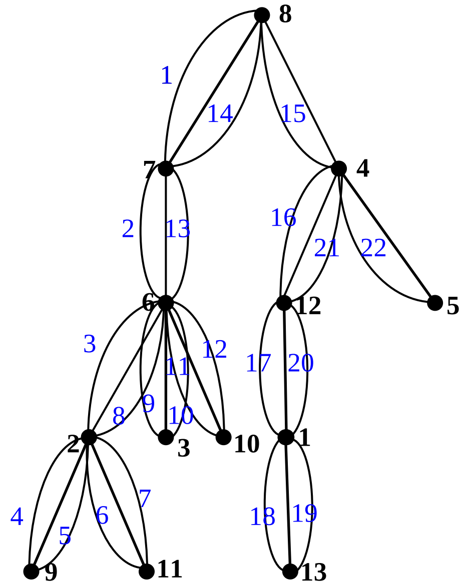
3. ***Для графа G2 построить дерево обхода вершин графа в ширину и в глубину.***

Построим остов графа G2.

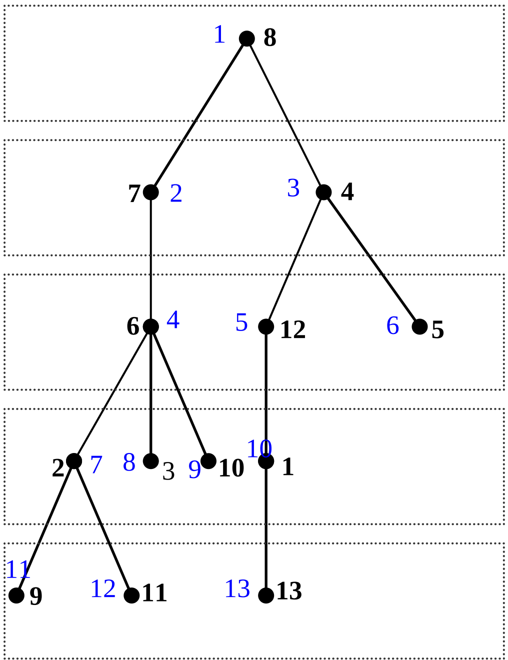
1)



Обход в глубину:



Обход в ширину:



- 1 ярус

- 2 ярус

- 3 ярус

- 4 ярус

- 5 ярус

Для графа G2 решить задачу построения остовов кратчайших маршрутов, используя алгоритмы Прима и Краскала (в качестве весов ребер использовать элементы матрицы Y).

1) Алгоритм Краскала

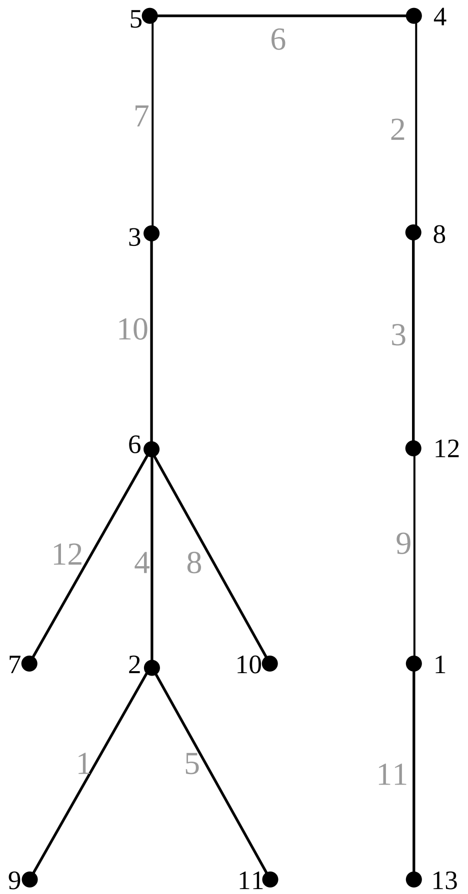
Ребра графа G упорядочиваются в порядке не убывания их весов. На каждом шаге к пустому графу O_p добавляется ребро с минимальным весом из списка ребер. Добавляемое ребро не должно приводить к образованию цикла. Алгоритм заканчивает работу, если количество ребер в графе станет равным p-1.

Матрица весов

C2	1	2	3	4	5	6	7	8	9	10	11	12	13
1	0	∞	∞	∞	∞	∞	∞	∞	∞	∞	∞	14	18
2	∞	0	21	∞	∞	7	∞	∞	6	∞	7	∞	∞
3	∞	21	0	∞	12	14	∞	∞	∞	∞	28	∞	∞
4	∞	∞	∞	0	7	∞	∞	6	∞	∞	∞	12	∞
5	∞	∞	12	7	0	∞	∞	∞	∞	∞	∞	∞	∞
6	∞	7	14	∞	∞	0	18	∞	∞	12	14	∞	∞
7	∞	∞	∞	∞	∞	18	0	21	∞	30	∞	∞	∞
8	∞	∞	∞	6	∞	∞	21	0	∞	∞	∞	6	∞
9	∞	6	∞	∞	∞	∞	∞	∞	0	∞	∞	∞	∞
10	∞	∞	∞	∞	∞	12	30	∞	∞	0	∞	∞	∞
11	∞	7	28	∞	∞	14	∞	∞	∞	∞	0	∞	∞
12	14	∞	∞	12	∞	∞	∞	6	∞	∞	∞	0	∞
13	18	∞	∞	∞	∞	∞	∞	∞	∞	∞	∞	∞	0

Ребра	Вес
$(2,9)_1$, $(4,8)_2$, $(8,12)_3$	6
$(2,6)_4$, $(2,11)_5$, $(4,5)_6$	7
$(3,5)_7$, (4,12), $(6,10)_8$	12
$(1,12)_9$, $(3,6)_{10}$, (6,11)	14
$(1,13)_{11}$, $(6,7)_{12}$	18
(2,3), (7,8)	21
(3,11)	28
(7,10)	30

Хорды:

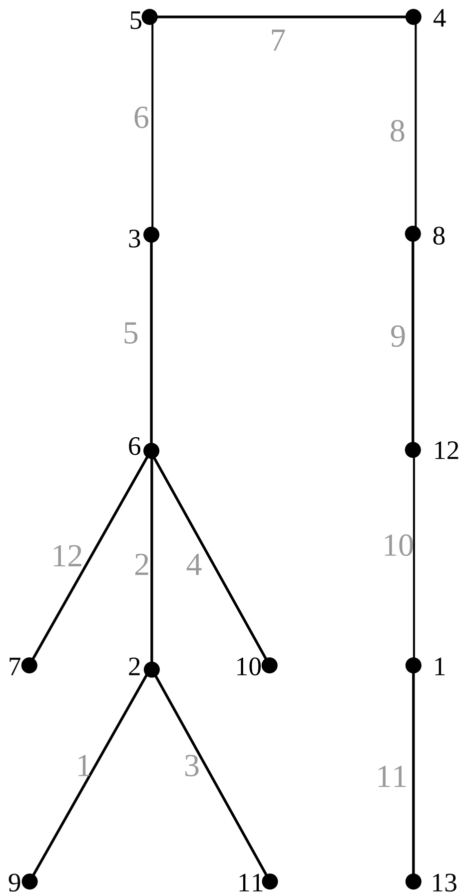


(4,12);
(6,11);
(2,3);
(7,8);
(3,11);
(7,10).

2) Алгоритм Прима

Алгоритм Прима отличается от алгоритма Краскала тем, что на каждом шаге строится не просто ацикличный граф, а дерево.

Ребра	Вес
$(2,9)_1$, $(4,8)_8$, $(8,12)_9$	6
$(2,6)_2$, $(2,11)_3$, $(4,5)_7$	7
$(3,5)_6$, (4,12), $(6,10)_4$	12
$(1,12)_{10}$, $(3,6)_5$, (6,11)	14
$(1,13)_{11}$, $(6,7)_{12}$	18
(2,3), (7,8)	21
(3,11)	28
(7,10)	30



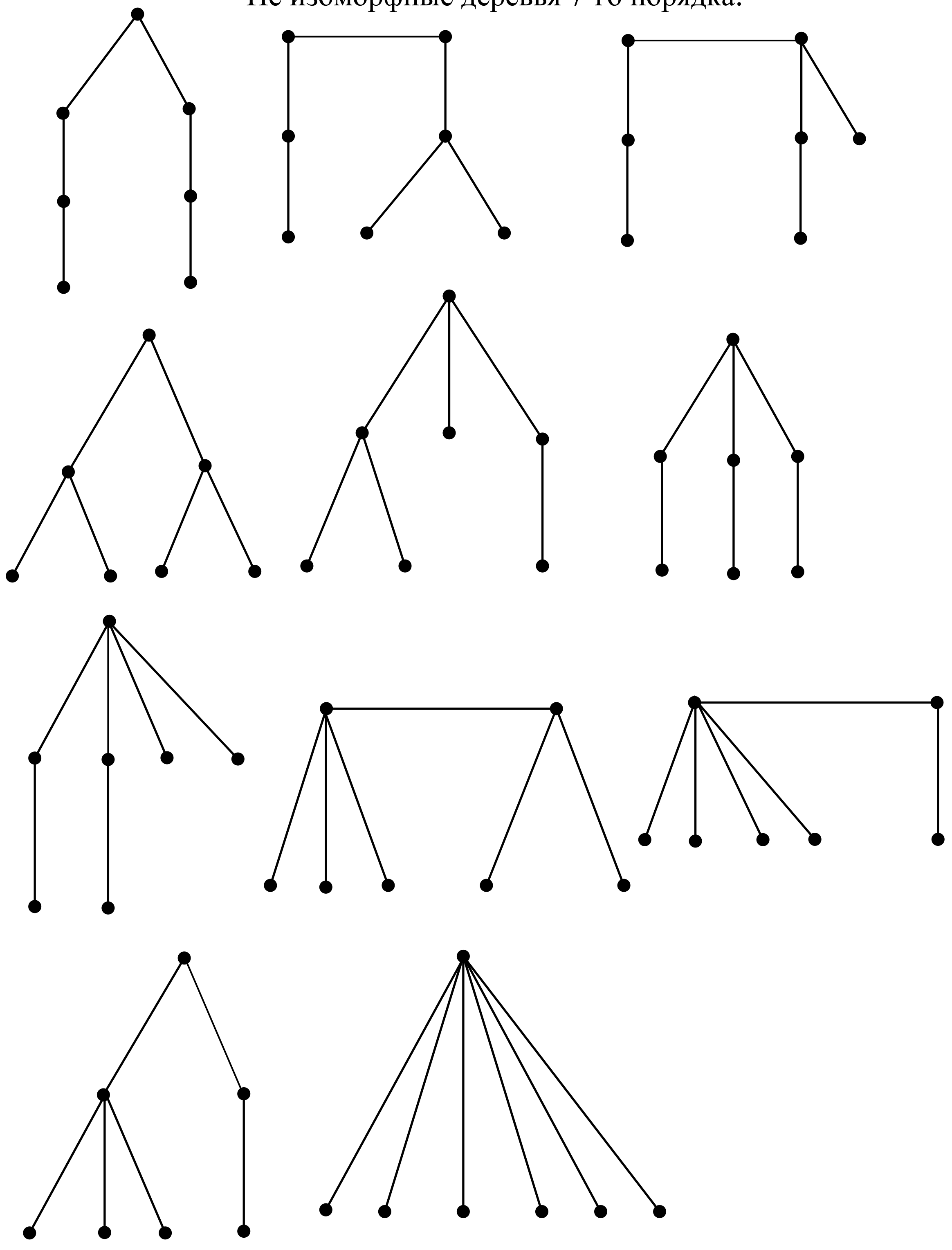
Хорды:

(4,12);
(6,11);
(2,3);
(7,8);
(3,11);
(7,10).

4. ***Сгенерировать все различные абстрактные не изоморфные друг другу деревья порядка 4 и 7.***

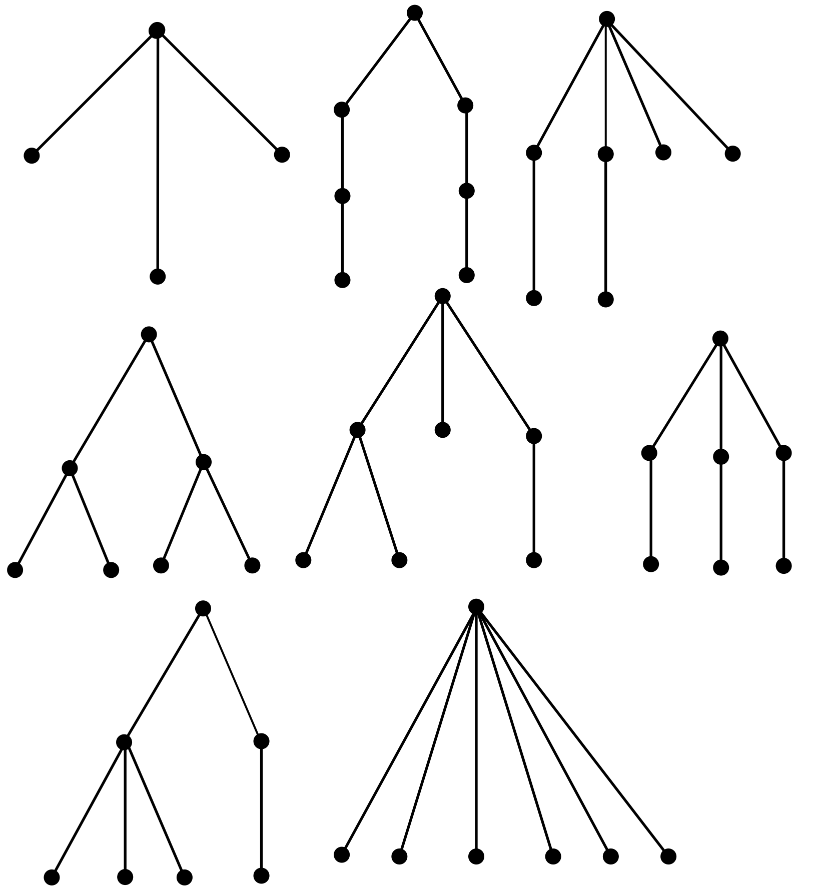
Не изоморфные деревья 4-го порядка:

Не изоморфные деревья 7-го порядка:

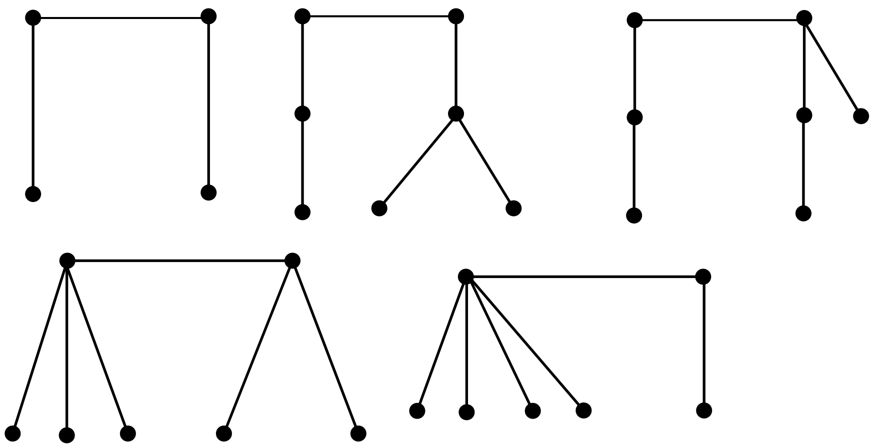


Разделить множество деревьев на 2 подмножества: с одной и с двумя центральными вершинами.

1) Деревья с одной центральной вершиной:



2) Деревья с двумя центральными вершинами:



Вывод: изучены понятия деревьев и остовов, приобретены практические навыки построения матрицы Кирхгофа и вычисления количества помеченных остовов.

Тема 2: «Циклы и обходы».

Цель работы: изучение понятий гамильтоновых и эйлеровых циклов, приобретение практических навыков построения матрицы циклов и матрицы фундаментальных циклов.

1. ***Используя алгоритм генерации варианта* GV*, построить неориентированный граф* G1: GV(5,{2,3}) *и граф* G2: GV(13,{6,7}).**

Построим граф G1, используя алгоритм генерации варианта GV(5,{2,3}).
S=<юфаеленаяковлевна>;
S=<ю,ф,а,е,л,н,я,к,о,в>;
n(S_i)={32,22,1,6,13}.

Y1	32	22	1	6	13
32	0	10	31	26	29
22	10	0	21	16	9
1	31	21	0	5	12
6	26	16	5	0	7
13	29	9	12	7	0

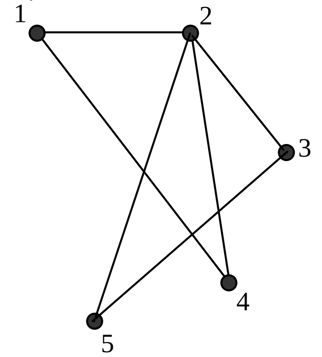
Матрица смежности

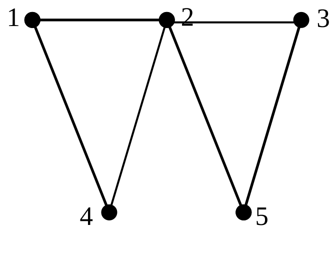
A1	1	2	3	4	5	deg(v_i)
1	0	1	0	1	0	2
2	1	0	1	1	1	4
3	0	1	0	0	1	2
4	1	1	0	0	0	2
5	0	1	1	0	0	2

Построим неориентированный граф G1, используя способ перечисления.
G1=(V1,E1): V1={1,2,3,4,5} – множество вершин графа;
E1={(1,2),(1,4),(2,3),(2,4),(2,5),(3,5)} – множество ребер графа.

Построим графическое изображение графа G1.

G1, или





Построим граф G2, используя алгоритм генерации варианта GV(13,{6,7}).

S=<юфаеленаяковлевна>;
S=<ю,ф,а,е,л,н,я,к,о,в>;
$n(S_i)$={32,22,1,6,13,15,33,12,16,3,29,18,14}.

Y2	32	22	1	6	13	15	33	12	16	3	29	18	14
32	0	10	31	26	19	17	1	20	16	29	3	14	18
22	10	0	21	16	9	7	11	10	6	19	7	4	8
1	31	21	0	5	12	14	32	11	15	2	28	17	13
6	26	16	5	0	7	9	27	6	10	3	23	12	8
13	19	9	12	7	0	2	20	1	3	10	16	5	1
15	17	7	14	9	2	0	18	2	1	12	14	3	1
33	1	11	32	27	20	18	0	21	17	30	4	15	19
12	20	10	11	6	1	2	21	0	4	9	17	6	2
16	16	6	15	10	3	1	17	4	0	13	13	2	2
3	29	19	2	3	10	12	30	9	13	0	26	15	11
29	3	7	28	23	16	14	4	17	13	26	0	11	15
18	14	4	17	12	5	3	15	6	2	15	11	0	4
14	18	8	13	8	1	1	19	2	2	11	15	4	0

Матрица смежности

A2	1	2	3	4	5	6	7	8	9	10	11	12	13	$deg(v_i)$
1	0	0	0	0	0	0	0	0	0	0	0	1	1	2
2	0	0	1	0	0	1	0	0	1	0	1	0	0	4
3	0	1	0	0	1	1	0	0	0	0	1	0	0	4
4	0	0	0	0	1	0	0	1	0	0	0	1	0	3
5	0	0	1	1	0	0	0	0	0	0	0	0	0	2
6	0	1	1	0	0	0	1	0	0	1	1	0	0	5
7	0	0	0	0	0	1	0	1	0	1	0	0	0	3
8	0	0	0	1	0	0	1	0	0	0	0	1	0	3
9	0	1	0	0	0	0	0	0	0	0	0	0	0	1
10	0	0	0	0	0	1	1	0	0	0	0	0	0	2
11	0	1	1	0	0	1	0	0	0	0	0	0	0	3
12	1	0	0	1	0	0	0	1	0	0	0	0	0	3
13	1	0	0	0	0	0	0	0	0	0	0	0	0	1

Построим неориентированный граф G2, используя способ перечисления.

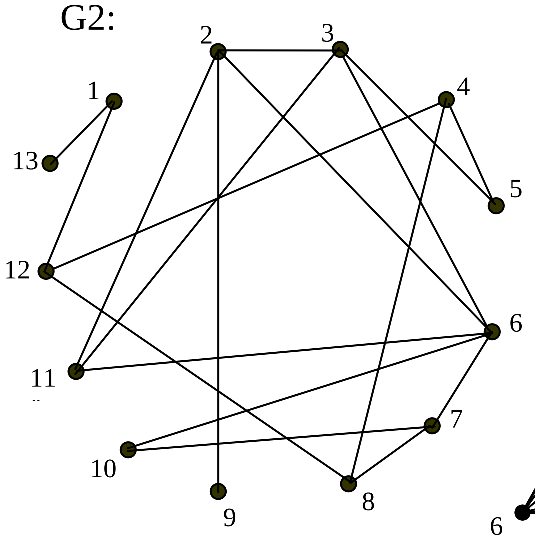
G2=(V2,E2):

V2={1,2,3,4,5,6,7,8,9,10,11,12,13} – множество вершин графа;

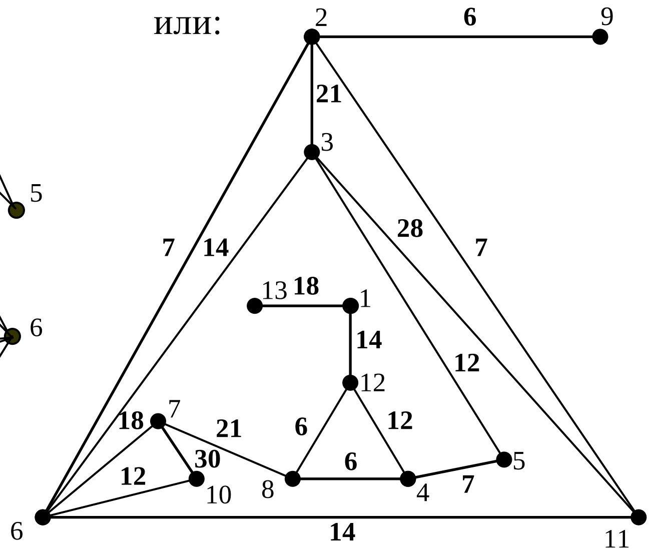
E2={(1,12),(1,13),(2,3),(2,6),(2,9),(2,11),(3,5),(3,6),(3,11),(4,5),(4,8),(4,12),(6,7),(6,10),(6,11),(7,8),(7,10),(8,12)} – множество ребер графа.

Построим графическое изображение графа G2.

G2:

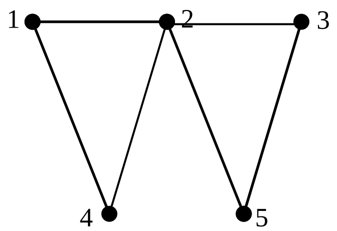


или:

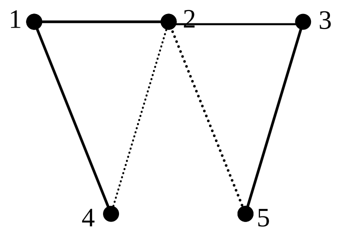


2. Для произвольного остова графа G1 построить матрицу фундаментальных циклов. Посчитать циклический и ациклический ранг, выразить три непростых цикла (если таковые имеются) через минимальную комбинацию базисных.

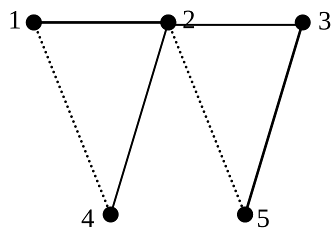
Граф G1:



Остов T1:



Остов T2:



$| V | = p = 5$, $| E | = q = 6$, $k = 1$.

$\nu=q-p+k=6-5+1=2$ – циклический ранг (хорды);

$\nu^*=p-k=5-1=4$ – коциклический ранг (ветви).

Выпишем все циклы в графе G1:

C1={1,2,4,1};

C2={2,5,3,2};

C3={2,1,4,2,5,3,2}.

Каждому циклу в графе G1 ставится в соответствие вектор длиной q.

Обозначим этот вектор: $z=(z_1,z_2,z_3,\ldots,z_q)$:

$z_1=(1,1,0,1,0,0)$;

$z_2=(0,0,1,0,1,1)$;

$z_3=(1,1,1,1,1,1)$.

В графе G1 все имеется только один непростой цикл C3.

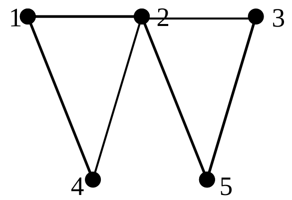
Выразим цикл C3 через базисные циклы C1 и C2:
$C3=C1\oplus C2=(1,1,0,1,0,0)\oplus(0,0,1,0,1,1)=(1,1,1,1,1,1)$.

Матрица фундаментальных циклов:

T	(1,2)	(1,4)	(2,3)	(2,4)	(2,5)	(3,5)
C1	1	1	0	1	0	0
C2	0	0	1	0	1	1
C3	1	1	1	1	1	1

3. Определить, являются ли графы G1 и G2 эйлеровыми, построить эйлеровы циклы по алгоритму Флёри, эйлеровы цепи. Если граф не эйлеров, добавить минимальное число ребер, делающих его эйлеровым.

1)Рассмотрим граф G1: в нем все вершины имеют четную степень, следовательно, граф G1 имеет эйлеров цикл, то есть является эйлеровым графом.

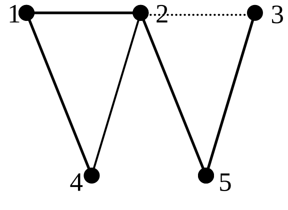


Построим в графе эйлеров цикл с помощью **алгоритма Флери**. Задача сводится к поиску способа нумеровации всех ребер в полученном графе таким образом, чтобы номер каждого ребра указывал порядок вхождения ребра в цикл.

V	(1,2)	(1,4)	(2,3)	(2,4)	(2,5)	(3,5)
№	1	2	4	3	6	5
№	4	5	3	6	1	2

Эйлеровы циклы:
C1={2,1,4,2,3,5,2};
C2={2,5,3,2,1,4,2}.

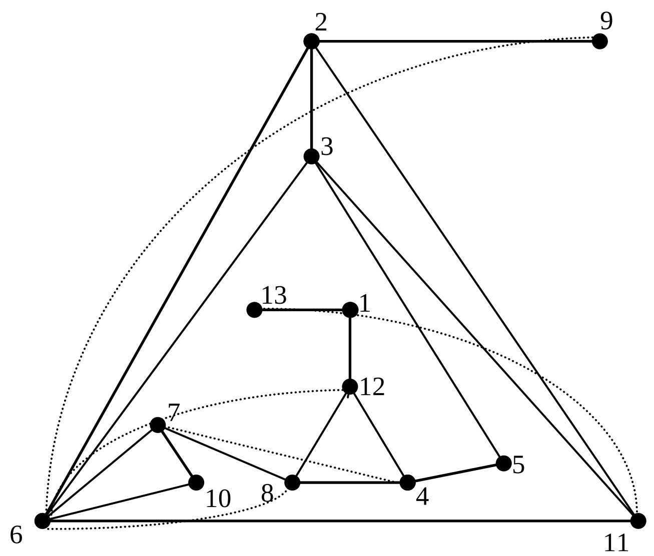
Построим в графе эйлеровы цепи. Для этого удалим ребро (2,3), тогда только две вершины – 2 и 3 – будут иметь нечетную степень, а это является необходимым и достаточным условием того, что граф G1 покрывается эйлеровой цепью.



Эйлеровы цепи:
P1={2,1,4,2,5,3};
P2={3,5,2,4,1,2}.

2) Рассмотрим граф G2: так как в нем вершины 4, 6, 7, 8, 10, 11, 12, 13 имеют нечетную степень, то граф G2 не имеет эйлерова цикла и, следовательно, не является эйлеровым графом.

Добавим ребра (6,9), (6,12),.(6,8), (4,7), (11,13). Получим граф, все вершины которого будут иметь четную степень. Это является достаточным условием для того, чтобы граф стал эйлеровым.

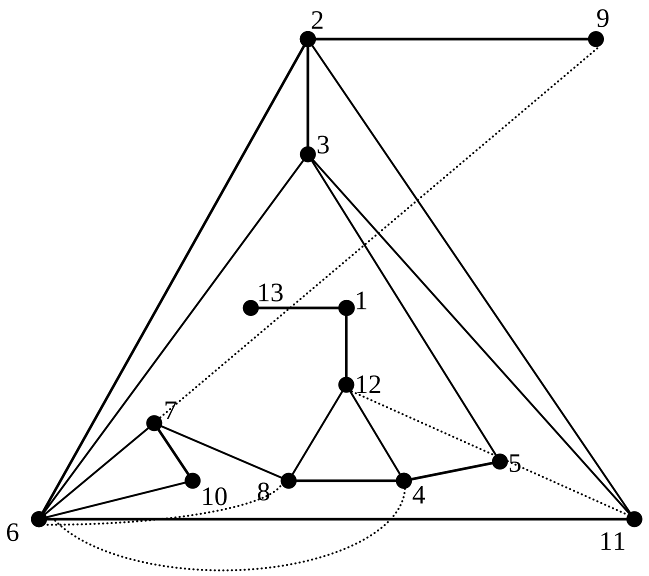


Построим в графе эйлеров цикл с помощью **алгоритма Флери**.

V	(1,12)	(1,13)	(2,3)	(2,6)	(2,9)	(2,11)	(3,5)	(3,6)
№	9	8	5	23	1	4	12	13
V	(3,11)	(4,5)	(4,7)	(4,8)	(4,12)	(6,7)	(6,8)	(6,9)
№	6	11	17	16	10	18	22	2
V	(6,11)	(6,12)	(7,8)	(7,10)	(8,12)	(11,13)	(6,10)	
№	3	14	21	20	15	7	19	

Эйлеров цикл:
C={2,9,6,11,2,3,11,13,1,12,4,5,3,6,12,8,4,7,6,10,7,8,6,2}.

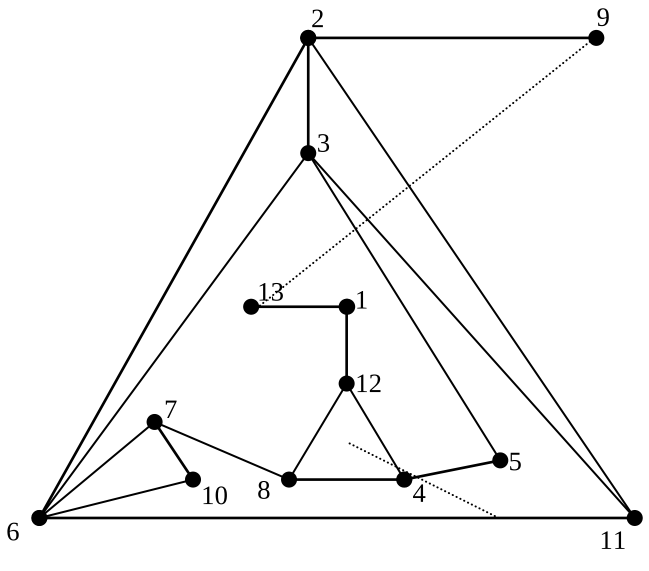
Построим в графе эйлерову цепь. Добавим ребра (1,6), (3,2), тогда только две вершины – 7 и 13 – будут иметь нечетную степень, а это является необходимым и достаточным условием того, чтобы граф G2 покрывался эйлеровой цепью.



Эйлерова цепь:
P2={13,1,12,11,2,9,7,8,12,4,3,11,6,7,10,6,8,4,6,2,3,6}.

4. Определить, является ли граф G2 гамильтоновым, построить гамильтонов цикл, используя алгоритм Робертса-Флореса. Если граф не является гамильтоновым, добавить минимальное число ребер, делающих его гамильтоновым.

Граф G2 не является гамильтоновым, так как в графе есть 2 висячие вершины: 9, 13, следовательно, нельзя выделить гамильтонов цикл, то есть такой простой цикл, который содержит каждую вершину графа.



Добавив ребро (9,13), можно выделить гамильтонов цикл:
C_g = {9,2,3,11,6,10,7,8,4,5,12,1,13,9}.

Следовательно, полученный граф - гамильтонов.

Алгоритм Робертса-Флореса.

Выберем вершину 10 в качестве начальной.

0) S={};	V={1,2,3,4,5,6,7,8,9,10,11,12,13};
1) S={10};	V={1,2,3,4,5,6,7,8,9,11,12,13};
2) S={10,7};	V={1,2,3,4,5,6,8,9,11,12,13};
3) S={10,7,8};	V={1,2,3,4,5,6,9,11,12,13};
4) S={10,7,8,4};	V={1,2,3,5,6,9,11,12,13};
5) S={10,7,8,4,5};	V={1,2,3,6,9,11,12,13};
6) S={10,7,8,4,5,12};	V={1,2,3,6,9,11,13};
7) S={10,7,8,4,5,12,1};	V={2,3,6,9,11,13};
8) S={10,7,8,4,5,12,1,13};	V={2,3,6,9,11};
9) S={10,7,8,4,5,12,1,13,9};	V={2,3,6,11};
10) S={10,7,8,4,5,12,1,13,9,2};	V={3,6,11};
11) S={10,7,8,4,5,12,1,13,9,2,3};	V={6,11};
12) S={10,7,8,4,5,12,1,13,9,2,3,11};	V={6};
13) S={10,7,8,4,5,12,1,13,9,2,3,11,6}	V={};

Так как цепь, проходящая через вершины множества S, имеет длину p-1=13-1=12, и в графе G2 существует ребро, соединяющее последнюю и первую вершины цепи, замыкая ее, то в графе существует гамильтонов цикл и множество S={10,7,8,4,5,12,1,13,9,2,3,11,6,10} – множество вершин этого цикла.

Вывод: изучены понятия гамильтоновых и эйлеровых циклов, приобретены практические навыки построения матрицы циклов и матрицы фундаментальных циклов.

7.4 ЛАБОРАТОРНАЯ РАБОТА №4

Тема: «Ориентированные графы».
Цель работы: изучение основных понятий ориентированных графов, приобретение практических навыков построения матрицы смежности, матрицы инцидентности, определение типов связности графа, построение конденсации орграфа.

1. ***Используя алгоритм генерации варианта*** **GV*, построить ориентированный граф* G1: GV(9,{6,7}).**

Построим граф G1, используя алгоритм генерации варианта GV(9,{3,4}).
S=<юфаеленаяковлевна>;
S=<ю,ф,а,е,л,н,я,к,о,в>;
$n(S_i)$={32,22,1,6,13,15,33,12,16}.

Y	32	22	1	6	13	15	33	12	16
32	0	10	31	26	19	17	1	20	16
22	54	0	21	16	9	7	11	10	6
1	33	23	0	5	12	14	32	11	15
6	38	28	7	0	7	9	27	6	10
13	45	35	14	19	0	2	20	1	3
15	47	37	16	21	28	0	18	3	1
33	65	55	34	39	46	48	0	21	17
12	44	34	13	18	25	27	45	0	4
16	48	38	17	22	29	31	49	28	0

Матрица смежности

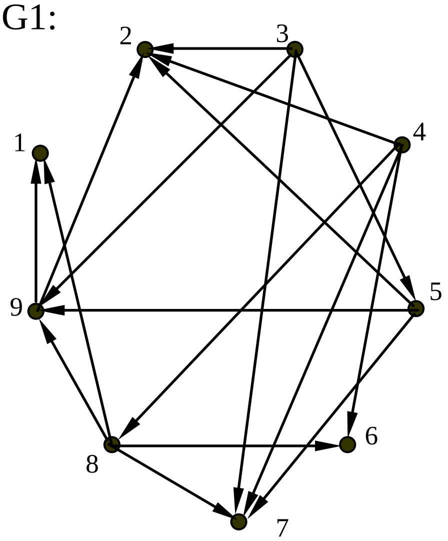
A	1	2	3	4	5	6	7	8	9
1	0	0	0	0	0	0	0	0	0
2	0	0	0	0	0	0	0	0	0
3	0	1	0	0	1	0	1	0	1
4	0	1	0	0	0	1	1	1	0
5	0	1	0	0	0	0	1	0	1
6	0	0	0	0	0	0	0	0	0
7	0	0	0	0	0	0	0	0	0
8	1	0	0	0	0	1	1	0	1
9	1	1	0	0	0	0	0	0	0

G1=(V1,A1):
V1={1,2,3,4,5,6,7,8,9} – множество вершин графа;

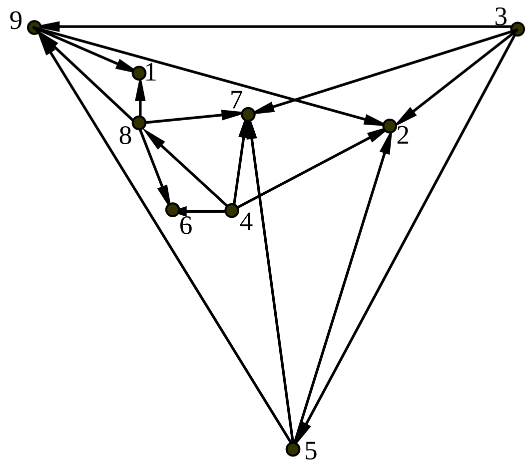
A1={(3,2),(3,5),(3,7),(3,9),(4,2),(4,6),(4,7),(4,8),(5,2),(5,7),(5,9),(8,1),(8,6),
(8,7),(8,9),(9,1),(9,2)} – множество дуг графа.

Построим графическое изображение графа G1.

G1:



или:



2. ***Построить матрицу смежности и матрицу инцидентности заданного графа.***

Матрица смежности

A	1	2	3	4	5	6	7	8	9
1	0	0	0	0	0	0	0	0	0
2	0	0	0	0	0	0	0	0	0
3	0	1	0	0	1	0	1	0	1
4	0	1	0	0	0	1	1	1	0
5	0	1	0	0	0	0	1	0	1
6	0	0	0	0	0	0	0	0	0
7	0	0	0	0	0	0	0	0	0
8	1	0	0	0	0	1	1	0	1
9	1	1	0	0	0	0	0	0	0

Матрица инцидентности

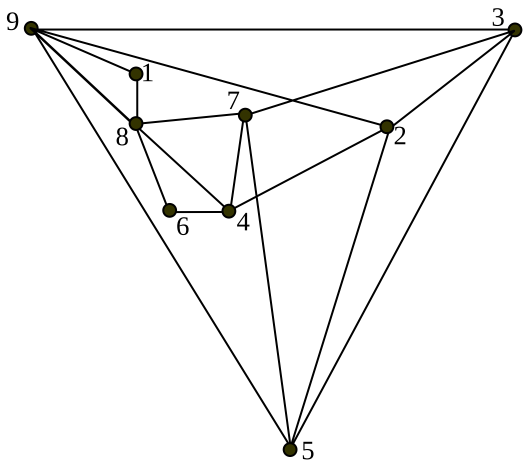
В	(3,2)	(3,5)	(3,7)	(3,9)	(4,2)	(4,6)	(4,7)	(4,8)	(5,2)	(5,7)	(5,9)	(8,1)	(8,6)	(8,7)	(8,9)	(9,1)	(9,2)	deg+	deg -	deg
1	0	0	0	0	0	0	0	0	0	0	0	-1	0	0	0	-1	0	0	2	2
2	-1	0	0	0	-1	0	0	0	-1	0	0	0	0	0	0	0	-1	0	4	4
3	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	4	0	4
4	0	0	0	0	1	1	1	1	0	0	0	0	0	0	0	0	0	4	0	4
5	0	-1	0	0	0	0	0	0	1	1	1	0	0	0	0	0	0	3	1	4
6	0	0	0	0	0	-1	0	0	0	0	0	0	-1	0	0	0	0	0	2	2
7	0	0	-1	0	0	0	-1	0	0	-1	0	0	0	-1	0	0	0	0	4	4
8	0	0	0	0	0	0	0	-1	0	0	0	1	1	1	1	0	0	4	1	5
9	0	0	0	-1	0	0	0	0	0	0	-1	0	0	0	-1	1	1	2	3	5

3. ***Построить основание и обратный граф. Определить, является ли граф симметричным.***

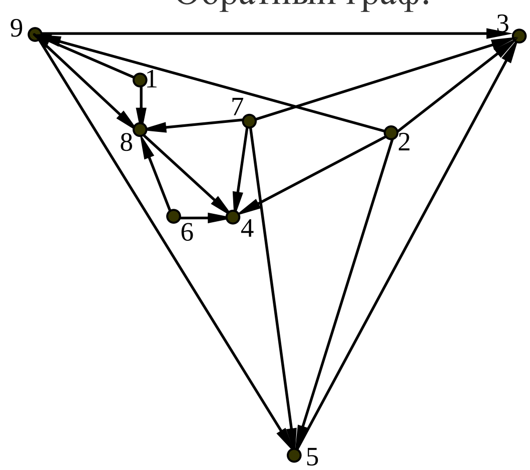
Основание орграфа – это неориентированный граф, полученный из орграфа в результате снятия ориентации дуг.

Обратный граф – это граф, дуги которого переориентированы по отношению к исходному орграфу.

Основание:



Обратный граф:



Граф G симметричным не является, так как для каждой его дуги (u,v) не существует дуга (v,u).

4. ***Построить ормаршрут, цепь, путь, полумаршрут, полуцепь, полупуть, замкнутый маршрут, цикл и контур.***

M1={4,8,9,2} – ормаршрут: чередующаяся последовательность вершин и дуг графа такая, что каждая дуга исходит из предыдущей вершины и заходит в последующую вершину;

M2={4,8,9,1} – цепь: ориентированный маршрут без повторяющихся дуг;
M3={3,5,9,2} – путь: цепь без повторяющихся вершин;
M4={3,2,5,9,1,7,4,2,5,9} – полумаршрут: маршрут основания орграфа;
M5={2,3,5,2,4,7,8,6} – полуцепь: цепь в основании графа;
M6={1,8,7,3,2,4} – полупуть: простой цикл в основании графа.

Замкнутый маршрут – это маршрут, который начинается и заканчивается в одной и той же вершине.

Цикл – это замкнутая цепь.

Контур – это замкнутый путь.

Замкнутый маршрут в графе выделить невозможно, следовательно, нельзя также выделить также цикл и контур.

5. ***Построить матрицу достижимости, контрдостижимости, взаимной достижимости. Представить ограниченно достижимую матрицу для числа достижимости, равного 2.***

Матрица достижимости

R	1	2	3	4	5	6	7	8	9
1	1	0	0	0	0	0	0	0	0
2	0	1	0	0	0	0	0	0	0
3	1	1	1	0	1	0	1	0	1
4	1	1	0	1	0	1	1	1	1
5	1	1	0	0	1	0	1	0	1
6	0	0	0	0	0	1	0	0	0
7	0	0	0	0	0	0	1	0	0
8	1	1	0	0	0	1	1	1	1
9	1	1	0	0	0	0	0	0	1

Матрица контрдостижимости

Q	1	2	3	4	5	6	7	8	9
1	1	0	1	1	1	0	0	1	1
2	0	1	1	1	1	0	0	1	1
3	0	0	1	0	0	0	0	0	0
4	0	0	0	1	0	0	0	0	0
5	0	0	1	0	1	0	0	0	0
6	0	0	0	1	0	1	0	1	0
7	0	0	1	1	1	0	1	1	0
8	0	0	0	1	0	0	0	1	0
9	0	0	1	1	1	0	0	1	1

Матрица взаимной достижимости

S	1	2	3	4	5	6	7	8	9
1	1	0	0	0	0	0	0	0	0
2	0	1	0	0	0	0	0	0	0
3	0	0	1	0	0	0	0	0	0
4	0	0	0	1	0	0	0	0	0
5	0	0	0	0	1	0	0	0	0
6	0	0	0	0	0	1	0	0	0
7	0	0	0	0	0	0	1	0	0
8	0	0	0	0	0	0	0	1	0
9	0	0	0	0	0	0	0	0	1

Ограниченно достижимая матрица для числа достижимости, равного 2.

R_2	1	2	3	4	5	6	7	8	9
1	1	0	0	0	0	0	0	0	0
2	0	1	0	0	0	0	0	0	0
3	1	1	1	0	0	0	1	0	1
4	1	0	0	1	0	1	1	0	1
5	1	1	0	0	1	0	0	0	0
6	0	0	0	0	0	1	0	0	0
7	0	0	0	0	0	0	1	0	0
8	1	1	0	0	0	0	0	1	0
9	0	0	0	0	0	0	0	0	1

6. ***Определить тип связности орграфа, выделить сильные компоненты.***

Так как в графе любые две вершины соединены полумаршрутом, то граф G слабосвязанный.

Так как в матрице взаимодостижимости единицы стоят только на главной диагонали, то сильными компонентами графа G являются вершины графа 1, 2, 3, 4, 5, 6, 7, 8, 9.

7. ***Построить конденсацию. Определить базы и антибазы.***

Так как сильными компонентами графа G1 являются его вершины, то конденсацией является исходный граф G1.

Так как в графе нет контуров, то базу образуют вершины, которые имеют полустепень захода равную нулю, то есть вершины {3,4}, а антибазу – вершины, имеющие полустепень исхода равную нулю, то есть вершины {1,2,6,7}.

Вывод: в данной лабораторной работе изучены основные понятия ориентированных графов, приобретены практические навыки построения матрицы смежности, матрицы инцидентности, определен тип связности графа, построена конденсация орграфа, определены его базы и антибазы.

ЛИТЕРАТУРА

1. Берж К. Теория графов и ее применения. – М.: Изд-во иностр. лит., 1962.
2. Оре О. Теория графов. – М.: Наука, 1968.
3. Харари Ф. Теория графов. – М.: Мир, 1973.
4. Басакер Р. Саати Т. Конечные графы и сети. – М.: Наука, 1974.
5. Кристофидес Н.. Теория графов. Алгоритмический подход. – М.: Мир, 1973.
6. Ахо А., Хопфорд Дж., Ульман Дж. Построение и анализ вычислительных алгоритмов. – М.: Мир, 1979.
7. Майника Э. Алгоритмы оптимизации на сетях и графах. – М.: Мир, 1981.
8. Свами М., Тхуласироман К. Графы, сети и алгоритмы. – М.: Мир, 1984.
9. Яблонский С. В. Введение в дискретную математику. – М.: Наука, 1986.
10. Липский В. Комбинаторика для программистов. – М.: Мир, 1988.
11. Лекции по теории графов/ Емеличев В.А., Мельников О.И., Сарванов В.И., Тышкевич Р.И. и др. – М.: Наука, 1990.
12. Нефедов В.Н., Осипова В.А. Курс дискретной математики. – М.: Изд-во МАИ, 1992.
13. Ловас Л., Планер М. Прикладные задачи теории графов. Теория паросочетаний в математике, физике, химии. – М.: Мир, 1998.

СОДЕРЖАНИЕ