1. Понятие множества. Способы задания множеств. Определение подмножества. Равные множества. Примеры.

**Множество** – совокупность объектов произвольной природы, которая рассматривается как единое целое.

**Задание** мн:

Перечислением элементов. Например, М={Ваня, Маня, Петя}

Заданием порождающей процедуры, например А={2,4,8,…,2x,…}

Указанием свойств элементов множества. Например, М={х|х – четное число.

Два множества называются **равными**, если они состоят из одних и тех же

элементов. Например, М={2,7}, А={2,7,2}.

Множество В называется **подмножеством** А, если любой элемент В принадлежит множеству А . Обозначается включение так: .

2. Последовательности. Декартово произведение двух множеств. Примеры. Декартово произведение n-множеств.

**Последовательности** - совокупности элементов с фиксированным порядком. <1,2>, <a,b,c>, <a1,a2,a3>.

Две последовательности считаются **равными**, если они состоят из одинаковых элементов, и порядок этих элементов у них одинаков.

**Декартовым произведением** множеств А, В – **множество** упорядоченных пар <a,b>, a принадлежит А, b принадлежит В. a1b1, a1b2, a2b1, a2b2. 

**Декартово произведение n-множеств**: множество последовательностей, содержащих **n элементов**, причем первый элемент должен быть из А1, второй из А2, …, n –элемент из An. <a1,b1, .., n1>.

3. Законы алгебры множеств. С доказательством одного из законов с помощью диаграмм Эйлера-Венна.

Законы алгебры множеств



4. Определение соответствия. Функциональное, полное, сюръективное, инъективное, биективное соответствие. Примеры.

Соответствие между множествами, считается заданным, если любому элементу а из А сопоставлен один (или несколько или ни одного) элемент b из В. 

**Свойства** соответствий.

1. Если соответствие определено для любого а из А (т.е. f(a) ≠ 0) , то оно

называется всюду определённым.

2. Если каждому элементу а из А поставлено в соответствие не более

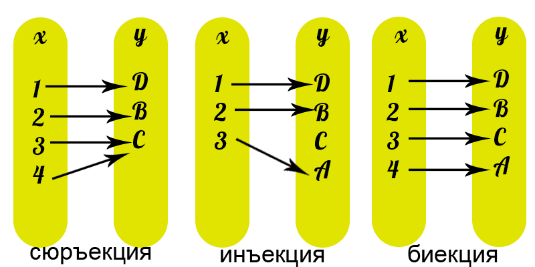
одного элемента b из В, то соответствие называется **функциональным**. В этом случае запись f(a) обозначает, не множество образов, а один образ и используется запись b=f(a).

3. Если разным элементам множества А поставлены в соответствия разные элементы множества В, т.е. если а1≠а2, то f(a1)∩ f(a2)= 0, то соответствие называется **инъективным**.

4. Если любой элемент b из В имеет непустое множество прообразов, то

соответствие называется **сюръективным.**

5. *Инъективное и* *сюръективное* отображение называется взаимно однозначным соответствием или **биекцией**



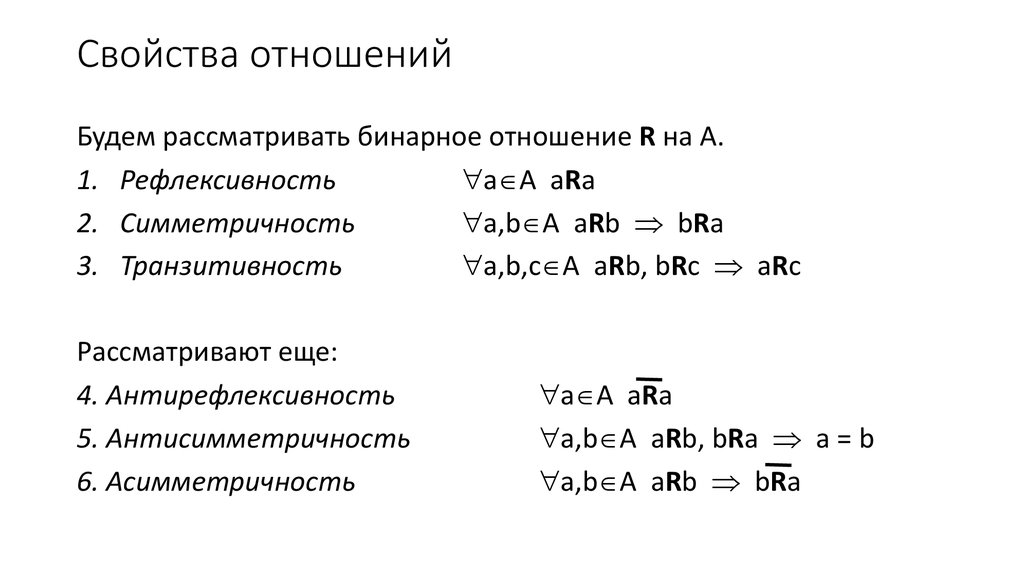
5. Отношения. Свойства бинарных отношений. Примеры.

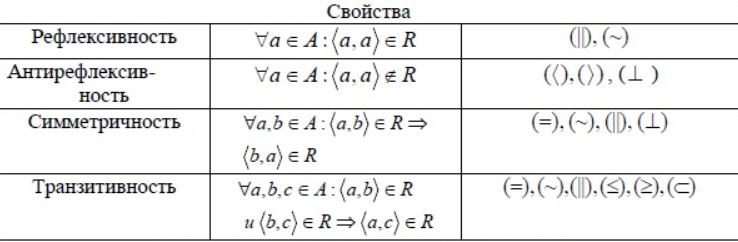
Если <a,b> из L, то говорят, что элемент а находится в отношении L к

элементу b или что **отношение L для a,b** выполняется (aLb)

Если В=А, отношение называется **бинарным** (двуместным) отношением на

множестве А.





6. Отношение эквивалентности. Классы эквивалентности. Фактор-множество. Примеры.

Бинарное отношение L на множестве А называется **отношением эквивалентности**, если оно является *рефлексивным*, *симметричным* и *транзитивным* (≡)

Примером отношения эквивалентности на множестве натуральных чисел N является равенство остатков при делении на некоторое фиксированное число n: a = b (mod n).

**Классы эквивалентности**

Для каждого а из А определяется класс эквивалентности[a]≡ , который включает все эквивалентные a элементы, т.е. [a]≡ ={b|a ≡ b}.

1. Классы эквивалентности покрывают все множество А.
2. Классы эквивалентности не пересекаются.

Множества классов эквивалентности множества А по эквивалентности ≡

называется **фактор-множеством** множества А по эквивалентности ≡ и

обозначается А/≡.

Пример: N/≡ ={ N0, N1, N2, N3, N4, , N5, N6}.

класс Ni (i=0,1,2,3,4,5,6) войдут числа, дающие при делении на 7 остаток i.

7. Отношение частичного порядка. Наименьший и наибольший, минимальный и максимальный элементы. Примеры.

**Строгое** **отношение частичного порядка** – это антирефлексивное антисимметричное, транзитивное отношение(>, <, строгое включение множ )

**Нестрогое отношение** частичного порядка - это рефлексивное антисимметричное, транзитивное отношение.( (≤), (≥), нестрогое включение множ )

**Нестрогое отношение частичного порядка** (≼)

Частично уп. множество (А; ≼).

Элемент а ч.у. множества А называется наибольшим, если для любого

в из А, верно в ≼ а.

Элемент а ч.у. множества А называется наименьшим, если для любого

в из А, верно а ≼ в.

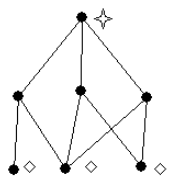
Элемент а ч.у. множества А называется максимальным, если для

любого в из А, верно в ≼ а или в несравнимо с а (в | а).

Элемент а ч.у. множества А называется минимальным, если для любого

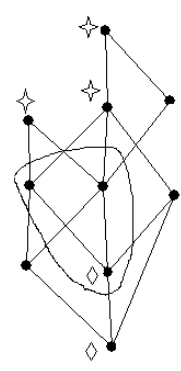
в из А, верно а ≼ в или в | а.

Очевидно, что наибольший (наименьший) элемент является максимальным (минимальным).



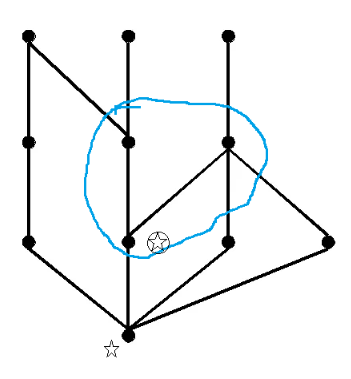
8. Частично упорядоченное множество. Верхняя (наименьшая верхняя) и нижняя (наибольшая нижняя) грани подмножества В в множестве А. Примеры.

Элемент а из А называется **верхней (нижней)** **гранью** множества В в ч.у. множестве А, если для любого элемента в из В верно **в ≼ а. (а ≼ в).**



Наименьший элемент множества всех верхних граней множества В в ч.у.

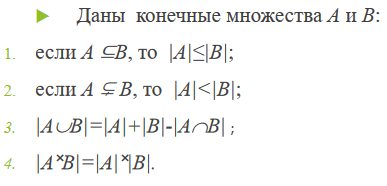
множестве А называется **наименьшей верхней гранью**. Аналогичным образом определяется набольшая нижняя грань.



9. Равномощные множества. Мощность конечного множества. Примеры равномощных бесконечных множеств.

Число элементов в конечном множестве А называют также его **мощностью** и обозначают |A|. Два конечных множества называются **равномощными** если они имеют одинаковое число элементов.

Свойства множеств.



Например, множество А – это множество всех натуральных чисел, а множество В – это множество всех четных чисел, тогда отображение f: x→2x, будет взаимно однозначным соответствием, которое каждому натуральному числу поставит единственное четное число, причем для любого четного числа найдется его единственный прообраз. Отсюда получим, что |А|=|В|.

10. Мощность конечного множества. Чему равна мощность множества всех подмножеств конечного множества? Доказательство.

Число подмножеств конечного множества, состоящего из n элементов, равно 2n.

Доказательство проведем методом математической индукции.

База. Если n=0, т. е. множество пусто, то у него только одно подмножество — оно само, и интересующее нас число равно 20=1. Индукционный шаг. Пусть утверждение справедливо для некоторого n и пусть M — множество содержащее n+1 элемент. Зафиксировав некоторый элемент a0 из M, разделим подмножества множества Mна два типа:

1. M1, содержащее a0,

2. M2, не содержащее a0, то есть являющиеся подмножествами множества M \ { a0}.

Подмножеств типа (2) по предположению индукции 2n. Но подмножеств типа (1) ровно столько же, так как подмножество типа (1) получается из некоторого и притом единственного подмножества типа (2) добавлением элемента a0 и, следовательно, из каждого подмножества типа (2) получается этим способом одно и только одно подмножество типа (1).Таким образом, мы получили, что суммарное количество подмножеств множества M будет 2n+1, и тем самым доказали теорему.

Множество всех действительных чисел равномощно множеству всех подмножеств натурального ряда

11. Конечные множества. Мощность конечного множества. Чему равна мощность объединения двух конечных множеств? Обосновать ответ.



Если А и В не пересекаются, то их объединение содержит элементы и того и другого множества, подсчитываем сначала элементы А, потом В и получаем |А и В|=|А|+|В|. Если пересечение А и В не пусто, то тогда элементы пересечения подсчитываются дважды и тогда, когда считаются элементы множества А и тогда, когда считаются элементы множества В. Для того что бы избавится от двойного подсчета общих элементов нужно вычесть их количество, т.е. |А и В|.

12. Равномощные множества. Отношение «иметь не большую мощность» на множествах. Свойства отношения «иметь не большую мощность».

А по мощности не больше множества В, если оно равномощно некоторому подмножеству множества В (возможно, самому В).

**Свойства**.

1. Если А и В равномощны, то А имеет не большую мощность, чем В.
2. Если А имеет не большую мощность, чем В, а В имеет не большую

мощность, чем С, то А имеет не большую мощность, чем С.

1. Если А имеет не большую мощность, чем В, а В имеет не большую

мощность, чем А, то они равномощны.

1. Для любых двух множеств А и В верно (хотя бы) одно из двух: либо А

имеет не большую мощность, чем В, либо В имеет не большую мощность,

чем А.

13. Равномощные множества. Счетное множество. Свойства счетных множеств. Доказательство одного из свойств.

Множество равномощное натуральному ряду называется **счётным**.

Счетное множество можно записать так: {x1, x2,…,xn,…}, где х1- это элемент поставленный в соответствие единицы, х2 – двойки и т. д. Мы привыкли в такой форме записывать любое бесконечное множество, но только для счетного множества эта запись говорит еще и том, что его элементы можно пронумеровать натуральными числам. Счётным можно назвать бесконечное множество, элементы которого можно пронумеровать натуральными числами

Свойства

(а) Подмножество счётного множества конечно или счётно.

(б) Всякое бесконечное множество содержит счётное подмножество.

(в) Объединение конечного или счётного числа конечных или счётных

множеств конечно или счётно.

(г) Декартово произведение конечного числа счётных множеств счётно.

Доказательство

(а) Пусть B - подмножество счётного множества A={a1, a2,…,an,…}.

Выбросим из последовательности a1, a2,…,an,… те члены, которые не

принадлежат B (сохраняя порядок оставшихся). Тогда оставшиеся члены

образуют либо конечную последовательность (и тогда B конечно), либо

бесконечную (и тогда B счётно).

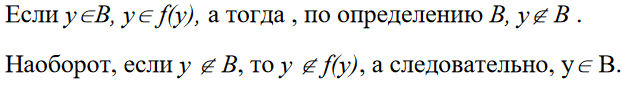
14. Теорема Кантора о мощности множества всех подмножеств множества А.

Т(Кантора) Никакое множество А не равномощно множеству всех своих подмножеств.

Доказательство. Предположим, что существует множество А, равномощное множеству всех своих подмножеств 2A, то есть, что существует взаимно однозначное соответствие f: A→2A , которое ставит в соответствие каждому элементу множества A некоторое подмножество множества A. Рассмотрим множество B, состоящее из всех элементов A, не принадлежащих своим образам при отображении f.



f взаимно однозначное соответствие, а В ⊆А, значит существует у из А, такой, что f(y)=B. Посмотрим может ли у принадлежать В.



В любом случае, получаем противоречие.

Скажем, что множество А менее мощно, чем множество В, если А по мощности не больше множества В и множества А и В не равномощны.

Поскольку любое множество А равномощно множеству одноэлементных подмножеств (любому а можно поставить в соответствие множество {а}), то можно сказать, что А менее мощно, чем 2A. Поскольку отношение равномощности на множествах является отношением эквивалентности, то мощностью или кардинальным числом множества A называется соответствующий ему класс эквивалентности.

Т Множество всех действительных чисел равномощно множеству всех подмножеств натурального ряда

15. Представление множества в алгебре множеств, порожденной системой образующих. Конституэнта, элементарное пересечение, нормальная форма кантора. Совершенная нормальная форма кантора. Примеры.

Фиксируем множества А1,…,An, которые назовем образующими.

Множество А1 \/ … \/ An будем считать универсальным.

Будем рассматривать все множества, которые получаются из образующих с помощью операций пересечения, объединения и взятия дополнения. Совокупность всех таких множеств обозначается С(А1…Аn).

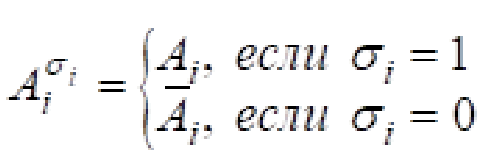
Ясно, что операции пересечения, объединения и взятия дополнения не выводят нас из множества С(А1…Аn).

Значит можно рассматривать алгебру А = <С(А1…Аn); \/,/\, ⁻ >,

В которой С(А1…Аn) – основное множество.

**Конституента** единицы - это такая функция, которая принимает значение единицы только для одной комбинации значений переменных.

Первичный терм



Пересечение первичных термов называется *элементарным пересечением*.

**Конституента** – это элементарное пересечение, которое отличается от

остальных элементарных пересечений тем, что

1) оно содержит ровно n членов (n -число образующих);

2) каждое образующее множество либо само входит в конституенту, либо входит его дополнение.

Если множество В принадлежит (А1…Аn) записано в виде объединения элементарных пересечений, то говорят, что оно представлено в **нормальной форме Кантора** (ДНФК).

Если множество В записано в виде объединения *конституент*, то говорят,

что В представлено в **совершенной нормальной форме** Кантора (СНФК)

16. Алгоритм нахождения минимальной формы кантора. Покрытие столбцов строками в двумерной таблице. Ядро покрытия. Минимальная нормальная форма кантора.

Алгоритм Квайна (Куайна) поиска минимальной НФК:

НФК называется минимальной, если она содержит минимальное количество термов;

Алгоритм Квайна поиска минимальной НФК, в качестве отправной точки используют СНФК. Алгоритм состоит из двух этапов.

1) Первый этап. Нахождение СНФК.

2) Второй этап. Нахождения минимальной НФК

**Покрытием столбцов строками** в двумерной таблице (записаны только нули и единицы) называют множество строк, в котором для каждого столбца найдется хотя бы одна строка из этого множества, на пересечении с которой этот столбец имеет единицу, причем при вычеркивании хотя бы одного элемента из этого множества строк указанное свойство не выполняется.

В **ядро покрытия** войдут те строки, которые содержатся в любом покрытии таблицы:

- отмечаем столбцы, имеющие только одну единицу

- Затем отмечаем строки, в которых стоят эти единицы. именно они войдут в ядро покрытия.

**Нахождение покрытий**

- Отмечаем столбцы, которые покрываются строками из ядра. Если покрываются все столбцы, то тогда покрытие совпадает со своим ядром.

- Если ядро покрытия не покрывает все столбцы, то перебором строк находим все покрытия.

- Среди найденных покрытий находим минимальные покрытия, именно они будут определять итоговые **минимальные НФК**

17. Алгоритм нахождения минимальной формы кантора. Нахождение простых импликант. Сокращенная нормальная форма кантора. Таблица Квайна. Пример.

**Простая импликанта** - элементарное пересечение, которое либо является *конституентой*, входящей в СНФК исходного множества, либо получено на шаге 3 в результате склеивания, и которое не может быть использовано в дальнейшем процессе склеивания.

**Сокращенная НФК** – это объединение всех простых импликант исходного множества

**Таблица Квайна** – двумерная таблица каждой строке которой взаимно – однозначно соответствует простая импликанта, столбцу- конституента, а на пересечении i-й строки и j-ого столбца находится единица, если j–я конституента участвовала в получении i–й простой импликанты; в противном случаи клетку (i, j) не заполняют или ставят в ней 0.

18. Граф ориентированный и не ориентированный – основные определения. Понятие псевдографа, мультиграфа.

Графом G(V,E) называется совокупность двух множеств — непустого множества V (множества вершин) и множества Е (Е — множество рёбер). Любое *ребро* (элемент из множества Е) должно соединять две вершины, поэтому его можно задать парой вершин. Если е из Е, то е=(v1,v2), где v1,v2 две разные вершины из V . Если порядок вершин при задании ребра важен, то первая вершина называется *начальной* вершиной, в вторая – *конечной* вершиной, само ребро называется *дугой*, а граф **ориентированным графом** или орграфом. В противном случае, граф называется **неориентированным** или просто графом.

**Мультиграф –** имеет*кратные ребра* (пара вершин может быть соединена двумя или более ребрами*).* Множество Е – ***мультимножество.***

**Псевдограф** – мультиграф + *петля* (ребро может начинаться и заканчиваться в одной и той же вершине).

Граф G'(V',E') называется **подграфом** графа G(V,E) (обозначаетcя G' ⊆G), если V' ⊆V к Е' ⊆ Е. Подграф называется ***собственным***, если он отличен от самого графа. Если V' = V, то G' называется ***остовным*** подграфом G.

Вершины, соединенные ребром (дугой) называются ***смежными***. Ребра, имеющие общую вершину, также называются ***смежными***. Каждое ребро (дуга) соединяет две вершины, говорят, что оно ***инцидентно*** этим вершинам. Также про вершину, являющуюся концом ребра (дуги), говорят, что она ***инцидентна*** этому ребру (дуге).

Если степень вершины равна ***нулю***, то вершина называется изолированной. Если степень вершины равна единицы, то вершина называется ***висячей.***

***Полным*** графом называется граф, у которого каждая вершина соединена с каждой. ***Пустым*** графом называется граф, у которого все вершины изолированы.

Граф называется ***регулярным***, если степени всех его вершин одинаковы.

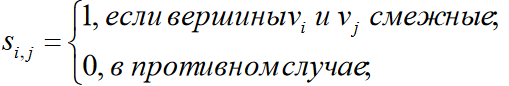
19. Способы задания графа. Примеры.

**Графический**. Все элементы множ V обозначаются точками на плоскости, проводятся линии соединяющие вершины.

**Матричный**. С помощью матрицы смежности и матрицы инцидентности.

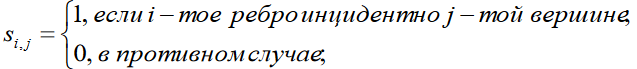
**Аналитический**. Задаётся список рёбер и список вершин.

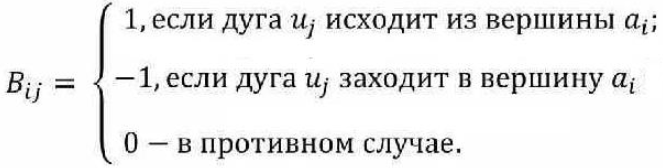
*Матрица смежности* – это квадратная матрица размерности n x n, значения элементов которой характеризуются смежностью вершин графа. (n-это количество вершин графа). Элементы матрицы определяются следующим образом:



В *матрице инцидентности* хранятся связи между инцидентными элементами – вершинами и ребрами. Размерность этой матрицы m x n, где m - количество ребер, n - количество вершин. Столбцы матрицы соответствуют вершинам, строки – ребрам. Если рассматривается неориентированный граф, то матрица

инцидентности определяется так:





20. Изоморфизм графов. Инварианты графа. Примеры.

Говорят, что два графа, G1(V,X) и G2(W,Y), изоморфны (обозначается G1 ~ G2, или G1 = G2), если существует взаимно-однозначное соответствие h: V → W, сохраняющее смежность, т.е. вершины u, v из V соединены ребром т. и т. т., когда их образы h(u),h(v) соединены ребром: (u, v) из X ↔ (h(u),h(v)) из Y.

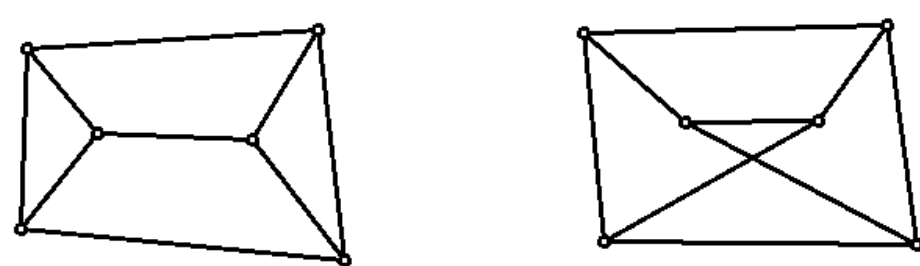
Изоморфизм графов является эквивалентностью.

Числовая характеристика, одинаковая для всех изоморфных графов,

называется инвариантом графа. Так, количество вершин графа или его ребер

— инварианты графа.

У изоморфных графов одинаковые числовые характеристики, одинаковое число ребер, вершин, одинаковые степени, одинаковое число вершин, имеющих определенные степени и т.д. Но из равенства не следует изоморфность.



21. Маршруты, цепи, циклы. Способы записи маршрутов. Простые цепи и циклы. Примеры.

**Маршрутом** в псевдографе называется чередующаяся последовательность вершин и рёбер, начинающаяся и заканчивающаяся вершиной, v0,x1,v1,x2,…,xn,vn, в которой для любого 0<i≤n ребро xi соединяет вершины vi-1, vi. n – длина маршрута.

**Цепь** – маршрут, у которого все ребра различны.

**Цикл** – замкнутая цепь.

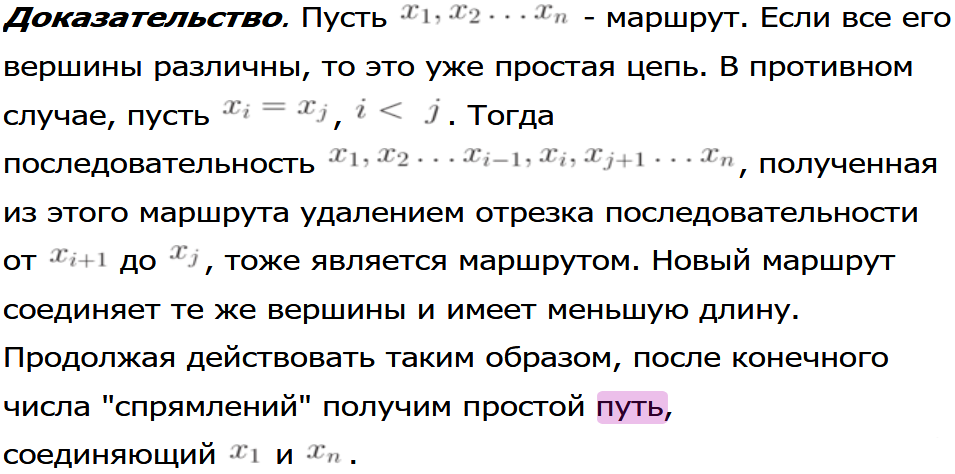
Цепь и цикл ***простые,*** если не содержат повторов вершин.

**Запись маршрута**. Последовательность вершин. v0,v1, …, vn. В псевдографе указать последовательность ребер x1, x2,…,xn.

**Т.** Если в графе степень каждой вершины не меньше 2, то в нем есть цикл.

22. Теорема о возможности выделения простой цепи в маршруте, соединяющем две разные вершины. Доказательство.

**Т.** В любом маршруте, соединяющем две различные вершины, содержится простой путь, соединяющий те же вершины.

****

**Т.** В любом цикле, проходящем через некоторое ребро, содержится простой цикл, проходящий через это ребро.

23. Связный граф. Компоненты связности графа. Разделяющее множество, разрез, мост(перешеек). Точки сочленения. Примеры.

Связные вершины – можно соединить маршрутом.

**Связный граф –** все вершины связные.

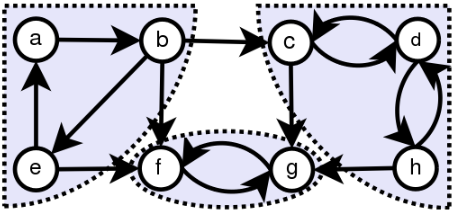
*Классы эквивалентности* по отношению связанности называются **компонентами связности** графа

Сильная связность орграфа: любые две вершины взаимодостижимы (путь)

Слабая – не взаимодостижимы (цепь)

Компонентами сильной связности орграфа называются его максимальные по включению сильно связные подграфы.

Областью сильной связности называется множество вершин компоненты сильной связности.



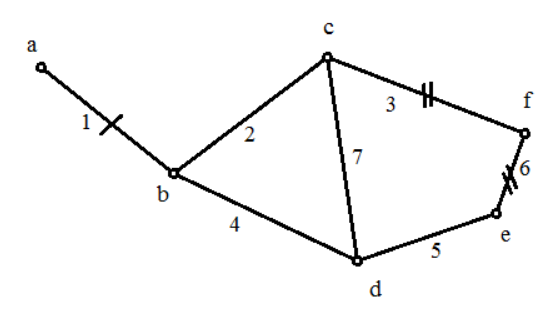
**Разделяющее множество** – это множество ребер, удаление которых из графа приводит к увеличению числа компонент связности. Связный граф делает граф несвязным.

**Разрез -** разделяющее множество ребер, которое не имеет собственного разделяющего подмножества.

**Мост** - Разрез, состоящий из одного ребра

**Точкой сочленения** называется вершина, удаление которой приводит к увеличению числа компонент связности.

Точка сочленения – вершина b. На рисунке отмечен одной черточкой мост, он в графе единственный. Разрезов много, на этом рисунке приведен пример одного из них, ребра входящие в него отмечены двумя черточками. Другие примеры разрезов {2,4}, {2,3,7}, {4,5,7}. Примерами разделяющих множеств будут сами разрезы и любые множества ребер графа, включающие разрезы.



24. Степени вершин. Расстояние между вершинами. Диаметр графа. Четные (нечетные) вершины. Число нечетных вершин. Доказательство.

**Расстояние между вершин** – длина самой короткой цепи, соединяющей эти вершины.

**Эксцентриситет** – расстояние от вершины до самой дальней вершины.

**Радиусом** – минимальный эксцентриситет

**Диаметр** – наибольшее расстояние между всеми парами вершин

**Центральной вершиной** графа является вершина чей эксцентриситет равен радиусу графа.

**Периферийной вершиной** графа является вершина чей эксцентриситет равен диаметру графа.

Теорема 14. (Лемма о рукопожатиях) Сумма степеней вершин графа (псевдографа) равна удвоенному количеству (q) рёбер: 

Доказательство. При подсчёте суммы степеней вершин каждое ребро учитывается два раза: для одного конца ребра и для другого. Вершину четной (нечетной) степени будем называть четной (нечетной) вершиной.

Следствие 1. **Число нечетных вершин чётно**.

**Доказательство.** По теореме 14 сумма степеней всех вершин — чётное число. Сумма степеней вершин чётной степени чётна, значит, сумма степеней вершин нечётной степени также чётна, следовательно, число нечетных вершин должно быть чётным числом.

25. Алгоритм Терри. Первая часть обоснования алгоритма Терри.

Нахождение маршрута (не оптимального)

1. идя по произвольному ребру, всякий раз отмечать направление, в котором оно пройдено,

2. исходя из некоторой вершины, всегда двигаться только по тому ребру, которое либо не было пройдено, либо было пройдено в другом направлении,

3. для всякой вершины, кроме начальной, отмечать ребро, по которому в эту вершину заходим впервые (первое заходящее ребро).

4. исходя из некоторой вершины, отличной от начальной, двигаться по первому заходящему ребру лишь в том случае, когда другой возможности нет.

Обоснование алгоритма Терри. Часть 1.

Если, начав движение согласно алгоритму Терри из вершины v, мы попали в тупик не достигнув вершины w, то

1. мы находимся в начальной вершине v.

2. все вершины графа G являются пройденными.

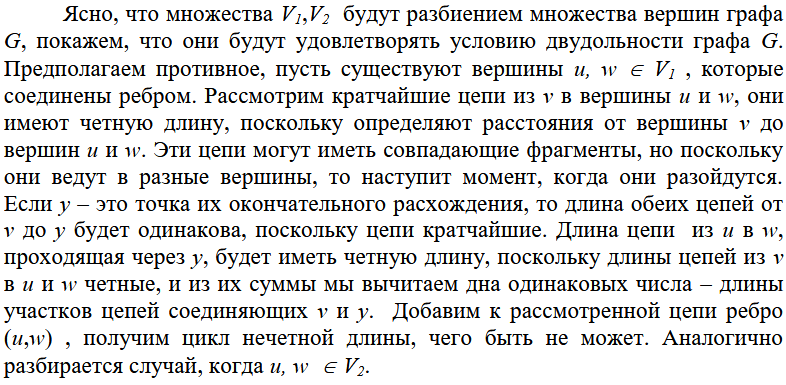
Докажем первый пункт. Предполагаем противное. Мы находимся в вершине u ≠ v. Пусть в вершине u мы побывали k раз (включая последний). Вершина u не начальная, значит по ребрам инцидентным вершине u в направлении «к» u мы прошли к-раз. Поскольку по одному ребру в одном направлении мы можем пройти только один раз, то ясно что ребер инцидентных u не менее, чем к. Из вершины u мы вышли к-1 раз, а сейчас выйти не можем, значит ребер инцидентных u всего к-1, что противоречит тому, что их должно быть не менее, чем к.

26. Двудольный граф. Признак двудольности графа. Доказательство достаточности.

Граф называется **двудольным**, если множество вершин можно разбить на два пересекающихся подмножества V1,V2, таких, что для любого ребра x, если х соединяет вершины u и v, то либо u из V1, v изV2; либо v из V1, u изV2 .

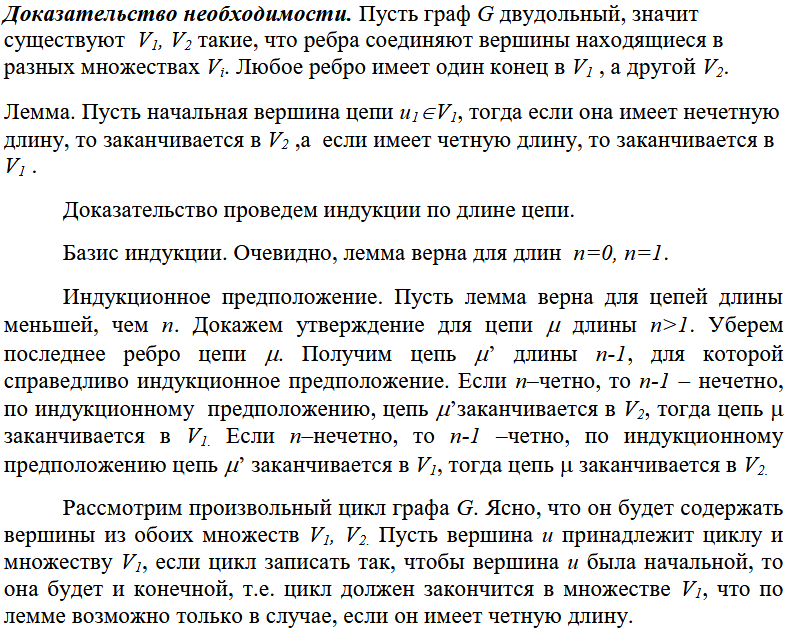
Теорема 17. (**Признак двудольности графа**). Граф является двудольным тогда и только тогда, когда все его циклы имеют четную длину

**Доказательство достаточности**. Рассмотрим связный граф. Пусть в графе все циклы имеют четную длину. Выберем произвольную вершину *v*. Множество *V1* – образуем как множество вершин, находящихся на четном расстоянии от *v, V2* как множество вершин на нечетном расстоянии от вершины *v*.



27. Двудольный граф. Признак двудольности графа. Доказательство необходимости.

Теорема 17. (**Признак двудольности графа**). Граф является двудольным тогда и только тогда, когда все его циклы имеют четную длину

****

28. Гамильтоновы цикл и цепь. Сложность определения наличия гамильтонова цикла в графе. Примеры.

**Простой цикл** называется **гамильтоновым**, если он содержит все вершины

графа.

**Простая цепь** называется **гамильтоновой**, если она содержит все вершины графа.

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

Граф, который содержит простую цепь, проходящую через каждую его вершину, называется **полугамильтоновым**



*Эйлеровы циклы содержат все ребра,* по одному разу каждое, *гамильтоновы циклы - все вершины*, по одному разу каждую.

Для решения вопроса о наличии эйлерова цикла в графе достаточно выяснить, все ли его вершины четны.

Простой критерий существования гамильтонова цикла в произвольном графе не найден. *Проблема существования гамильтонова цикла* принадлежит к классу так называемых *NP-полных задач*.

29. Эйлеровы циклы и цепи. Признак существования эйлерова цикла в связном графе. Доказательство.

**Эйлеров цикл –** цикл, содержащий *все ребра* графа.

Связный **эйлеров граф** – содержит эйлеров цикл.

**Эйлерова цепь** – содержит *все ребра* графа.

Полуэйлеров граф – содержит эйлерову цепь.

Т. Если граф G *связный* и *все вершины четные*, то он обладает эйлеровым циклом.

Т. Если граф связный и А и В единственные нечетные вершины его, то граф обладает эйлеровой цепью с концами А и В.

30. Планарный и плоский графы. Определение грани плоского графа. Примеры.

Граф называется **плоским**, если его можно изобразить на плоскости так, чтобы его ребра пересекались только в вершинах

Граф **планарен**, если его можно уложить на плоскости так, чтобы ребра *пересекались только в вершинах*

Граф, *изоморфный* плоскому графу, называется планарным

**Гранью** в плоском представлении графа называется часть плоскости, ограниченная простым циклом и не содержащая внутри других циклов

***мост - ребро,*** соединяющее циклы. Такие мосты называются перегородками

Простой цикл, ограничивающий грань, называется ***границей грани***. Две грани будем называть соседними, если их границы имеют хотя бы одно общее ребро

*Два графа* называют **гомеоморфными**, если они получены разбиением ребер двух изоморфных графов

Т. **Граф планарен** тогда и только тогда, когда не содержит подграфов, гомеоморфных К5 или К3,3.

31. Пять определений дерева.

1. Граф является деревом, то есть ***связным без циклов***графом.

2. Граф является связным и число его рёбер ровно на единицу меньше числа вершин. ***(|U| - 1= |V|)***

3. Две любые различные вершины графа G можно ***соединить*** ***единственной*** ***простой******цепью***.

4. Граф является *связным* и *каждое его ребро является мостом*.

5. Граф G ***не содержит циклов***, но, *добавляя* к нему любое новое *ребро*, ***получаем*** ровно один ***цикл*** (и притом простой цикл), проходящий через добавляемое ребро.

32. Доказательство того, что у дерева число ребер на единицу меньше, чем вершин.

Доказательство проведем индукцией.

Если n=1, то очевидно, что m=0 .

Предполагаем, что утверждение справедливо для всех деревьев, имеющих не более чем k вершин.

Рассмотрим, дерево, содержащее k+1 вершину, т.е. n=k+1.

По лемме 2, дерево G имеет хотя бы одну висячую вершину. Удалим одну висячую вершину вместе с инцидентным ей ребром и получим граф G’, который по лемме 3 является деревом.

Поскольку G’ имеет n-1 вершину, то к нему можно применить индукционное предположение, следовательно, m-1=(n-1)-1.

Отсюда получим m=n-1.

33. Доказательство того, что связный граф, имеющий число ребер на единицу меньше чем вершин является деревом.

Доказательство проведем индукцией по n количеству вершин.

Если n=1, то m=0. Граф, состоящий из одной изолированной вершины, является деревом.

Пусть доказываемое утверждение справедливо для любого графа с менее чем n вершинами.

Докажем справедливость этого утверждения для графа G с n-вершинами.

Покажем, что в G имеется висячая вершина. Если её нет и граф связан, то степень любой вершины не меньше двух. Тогда сумма степеней всех вершин не меньше 2n, с другой стороны, для любого графа сумма степеней всех вершин равна 2m. Следовательно, 2m >= 2n, т.е. m >= n, что противоречит тому, что m=n-1, следовательно, в G есть висячая вершина v.

Удалим её вместе с инцидентным ей ребром. В результате получим граф G`с n-1 вершиной и m-1 ребром. В силу леммы 3, граф G`-связный. Поскольку m=n-1 по условию теоремы, то количество рёбер в графе G`на единицу меньше чем вершин, т.е. для графа G` можно воспользоваться индукционным предположением, т.е. граф G`-дерево. Но тогда в силу леммы 4, граф G-тоже дерево.

34. Остовное дерево связного графа. Алгоритм выделения остовного дерева с обоснованием.

**Остовное дерево (остов)** связного графа G - подграф дерево, содержащее все вершины графа.

**Алгоритм ( 1-(2-3) )**

Дан связный граф G. Строим подграф D графа G, а затем покажем, что он является остовным деревом.

Шаг 1. Выбираем произвольную вершину и помещаем ее в подграф D.

Шаг 2. Если D содержит все вершины графа, то завершает работу, в противном случае, переходим к шагу 3.

Шаг 3. Находим вершину v, которая еще не попала в подграф D и является смежной некоторой вершине w, находящейся в подграфе D. Поскольку граф G является связным, то такая вершина найдется. Помещаем вершину v и ребро (v,w) в подграф D. Переходим к шагу 2.

**Обоснование**

Поскольку алгоритм завершает работу только тогда, когда все вершины графа G окажутся в подграфе D, то D является остовом графа G.

На первом шаге в подграф D помещена 1 вершина, на шаге 3 в подграф D помещаются 1 вершина и 1 ребро, причем вершина и ребро выбираются так,

чтобы не нарушить связность подграфа D.

Таким образом, мы построим связный граф, у которого ребер на единицу меньше, чем вершин, т.е. дерево. Значит, подграф D является остовным деревом графа G.

35. Обходы графа. «Поиск в глубину».

Обход графа – это некоторое *систематическое перечисление его вершин или ребер*

* Шаг 1. В стек помещаем произвольную вершину и метим ее.
* Шаг 2. Из стека извлекаем вершину и помещаем в список обхода. Обозначим, извлеченную из стека вершину буквой х. В стек помещаем все непомеченные смежные вершине х вершины, попавшие в стек вершины метим.
* Шаг 3. Проверяем, есть ли в стеке вершины, если есть, то переходим к шагу 2, в противном случае завершаем работу.

Очевидно, что если граф связный, то мы перечислим все его вершины, если не связный, то только те вершины, которые находятся той же компоненте связности, что и выбранная вершина.

36. Обходы графа. «Поиск в ширину».

* Шаг 1. В очередь помещаем произвольную вершину и метим ее.
* Шаг 2. Из очереди извлекаем вершину и помещаем в список обхода. Обозначим, извлеченную из очереди вершину буквой х. В очередь помещаем все непомеченные смежные вершине х вершины. Попавшие в очередь вершины метим.
* Шаг 3. Проверяем, есть ли в очереди вершины, если есть, то переходим к шагу 2, в противном случае завершаем работу.

37. Взвешенный граф. Алгоритм Примы поиска наилегчайшего остовного дерева.

Граф называется **взвешенным**, если его *ребрам* ***приписаны*** некоторые ***значения***, называемые ***весом***.

Матрица смежности взвешенного графа содержит вместо единиц веса ребёр.

**Алгоритм Прима**

1. Включим любую вершину в остов.

2. Найдём *минимального веса ребро*, исходящее из вершин остова *к неостовным вершинам*. *Включим* в остов *вершину* этого ребра. *Повторяем* пункт, пока все вершины не окажутся в остове.

38. Взвешенный граф. Жадный алгоритм поиска наилегчайшего остовного дерева.

Выбирается ребро, имеющее *минимальный вес среди всех рёбер*, и включается в остов. Из оставшихся ребер выбирается снова то, которое имеет минимальный вес, и включается в остов, если при этом *не образуются циклы*. Процесс продолжается до тех пор, пока все вершины не будут включены в остов.

39. Длина пути (цепи) во взвешенном графе. Кратчайший путь. Задачи поиска кратчайших путей. Первое свойство кратчайшего пути с обоснованием.

Длина пути – это сумма длин ребер, входящих в этот путь.

Задачи поиска кратчайшего пути:

Между двумя вершинами. 1-1

От данной вершины во все. 1-Мн

В данную вершину из всех. Мн-1

Из каждой вершины в каждую. Мн-Мн

**Первое свойство кратчайшего пути** (Подпуть кратчайшего пути – кратчайший)

Имеется кратчайший путь p1k=(v1,v2,… ,vk) от вершины v1 до вершины vk, а также его подпуть p'(vi,vi+1,… ,vj), при этом действует 1 <= i <= j <= k.

Если p — кратчайший путь от v1 до vk, то p' также является кратчайшим путем от вершины vi до vj

**Обоснование**

Стоимость пути p равна стоимость пути p' и стоимость остальной части. Пусть есть более короткий путь p'. Тогда стоимость p будет меньше, что противоречит данному условию минимальности p.

40. Кратчайший путь. Алгоритм Дейкстры нахождения кратчайшего пути между двумя вершинами графа.

**Алгоритм Де́йкстры**

Шаг 1. Первой вершине присваивается индекс равный 0. Остальным – бесконечность. Первая вершина объявляется пройденной и текущей.

Шаг 2. Рассматриваем все вершины, смежные текущей и не пройденные. Для этих вершин проверяем неравенство L(с,у) < I(у) - I(с) (c- текущая вершина, у- рассматриваемая). Если это неравенство выполнено, то заменяем индекс вершины у на I(у) = L(с,y) + I(с).

Шаг 3. Среди не пройденных вершин ищется вершина с минимальным индексом. Ее мы проходим и объявляем текущей (предыдущая текущая вершина теряет свой статус «текущей вершины»).

Шаг 4. Если текущая вершина совпадает с конечной, то завершаем работу. Иначе Переходим к шагу 2.

41. Кратчайший путь. Алгоритм Форда-Беллмана нахождения кратчайшего пути.

**Алгоритм Форда-Беллмана**. Расчет длины кратчайшего пути.

Дан неориентированный взвешенный граф G и указана некоторая начальная вершина н и некоторая конечная вершина к.

Суть: Сначала ищем *все кратчайшие пути от вершины н до остальных* вершин. Затем *строим кратчайший путь от* вершины *к до н*.

Алгоритм: присвоим вершинам индексы, которые по завершению работы будут равны длинам кратчайших путей. Полагаем, что индекс вершины н равен 0, а всех остальных вершин ∞ .

На каждой шаге просматриваются все рёбра графа, и алгоритм пытается уменьшить значение индекса. Это возможно, если выполняется неравенство L(с,у) < I(у) - I(с) (где L(с,у) длина ребра (с,у), I(у), I(с)- индексы вершин у, с, соответственно). Если это неравенство выполнено, то заменяем индекс вершины у на L(с,y) + I(с). Расстановка индексов завершается, если данное неравенство не будет выполнено ни для одного ребра.

Построение самого кратчайшего пути начинается от конечной вершины к. Сначала находим ребро, длина которого участвовала в вычислении индекса вершины к, это ребро включаем в кратчайший путь, и переходим по этому ребру к вершине к’, и ищем ребро, определившее индекс, уже вершины к’ и т.д. Процедуру повторяем до тех пор, пока не достигнем вершины н.

P.s.: Кратчайших путей может быть несколько, но по этому алгоритму будет построен только один из них

42. Цикломатика. Цикломатическая матрица. Базисная система циклов.

Цикл может быть задан:

1. Перечислением вершин в порядке их прохождения, начиная с любой вершины цикла.

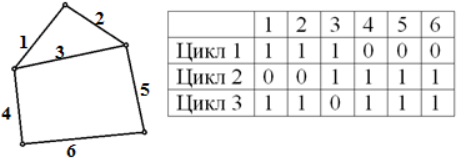
2. Перечислением ребер (в любом порядке), входящих в цикл;

3. С помощью цикломатических матриц

**Цикломатическая матрица**

Количество столбцов в цикломатической матрице равно количеству ребер, количество строк – количеству циклов. Каждый элемент этой матрицы определяется так:





**Базисная система циклов**

Вектор М называется линейной комбинацией векторов R1,R2,…Rn, если

Если система векторов R1,R2,…Rn. такая, ни один вектор Ri, (1≤i≤n) нельзя представить линейной комбинацией других, то она называется ***линейно независимой***.

Если система циклов такова, что вектора, определяющие эти циклы, линейно независимы, то такая система циклов также называется ***линейно независимой***

**Базисная** с**истема циклов**– состоит только из *простых линейно независимых циклов* и любой цикл графа можно представить в виде линейной комбинации циклов из этой системы. Вектор, задающий базисный цикл, будем так же называть базисным вектором

43. Цикломатика. Алгоритм нахождения базисной системы циклов.

**Хорда** *- ребро*, не входящие в остовное дерево.

Пусть дан связный граф G, содержащий n вершин и m рёбер, и в нем выделено остовное дерево D. Остовное дерево содержит n-1 ребро, то граф содержит m-n+1 хорду.

Построим систему циклов связного графа G следующим образом:

Выделим остов в графе G, обозначим его буквой D. К остову D добавим некоторую хорду, получим граф D’, который содержит единственный простой цикл. Этот цикл включаем в строящуюся систему векторов, затем к остову добавляем другую хорду, получаем граф D’’и следующий простой цикл, и т.д. Каждая хорда определяет свой цикл, в который входит только она и ребра из остова. В построенную систему циклов войдут ровно **m-n+1 цикл**, ровно столько, сколько хорд в графе. Обозначим эту систему циклов С(D).

**Т**. Каждый остов D задает систему базисных циклов С(D)

44. Цикломатика. Задание циклов векторами. Сложение векторов по   
модулю 2. Доказательство того, что эту операцию можно использовать для нахождения новых циклов. (Что задает вектор Р + Т, если вектора Р, Т задают циклы?).

1. Строим некоторое *остовное дерево* D графа G.

2. По каждой *хорде* построенного остовного дерева *строим цикл*, тот цикл, который содержит одну хорду и ребра остовного дерева. Таким образом мы построим базисную систему циклов.

3. Вектора всех базисных циклов заносим в цикломатическую матрицу.

4. Выполняем операцию  - побитового сложения по модулю два, над различными наборами векторов базисных циклов. После выполнения очередной операции , проводим исследование графа, задаваемого полученным вектором, на связность, если граф связный, то вектор задает цикл, и его помещаем в матрицу.

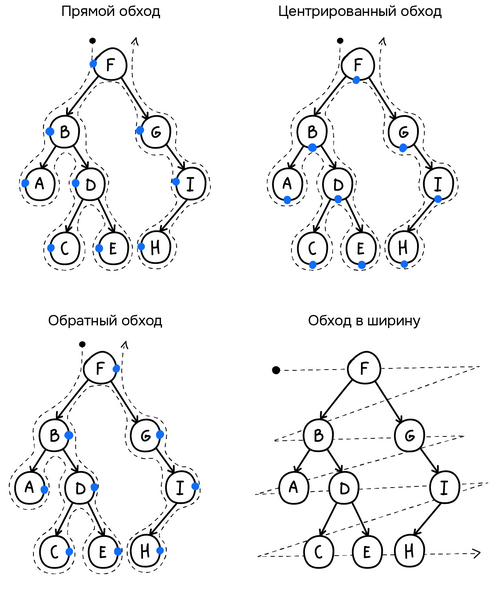
**Доказательство**

Поскольку RR=0 и 0R=R, то в наборы векторов бессмысленно вводить два одинаковых базисных вектора. Поэтому нам нужно будет рассмотреть не более 2i наборов, где i – число базисных векторов.

Пусть граф G имеет n – вершин, m –ребер, k – компонент связности, тогда число m-n+k называется цикломатическим.

Если для каждой компоненты связности построить остовное дерево, то общее количество хорд будет равно m-n+k, и, значит, столько будет базисных циклов, т.е. цикломатическое число определяет количество базисных циклов.

45. Дерево, как модель иерархических отношений. Симметричный обход. Пример.



Дерево - связный граф без циклов. Но для моделирования иерархических отношений вершины называют, **узлами**, а ребра задают родительские отношения. Один узел выделен как **корень**. Корень может иметь **сыновей.** Узлы, не имеющие сыновей, называются **листьями**

**Симметричный обход**

Если дерево Т - ***нулевое дерево***, то в список обхода заносится пустая запись.

Если дерево состоит из ***одного узла***, то в список записывается этот узел.

***Сложное дерево*** Т

- n-корень дерева Т;

- Т1,…, Тк, - поддеревья дерева Т, корнями которых, являются сыновья узла n.

При прохождении в симметричном порядке узлов дерева Т, сначала посещаются в симметричном порядке все узлы поддерева Т1, далее корень n, затем последовательно в симметричном порядке все узлы поддеревьев Т2,…Тк.

46. Дерево, как модель иерархических отношений. Обратный обход. Пример.

***Сложное дерево*** Т

Спосещается в обратном порядке все узлы поддерева Т1, затем Т2, затем Т3, …, Тк, так же в обратном порядке и затем корень n.

47. Дерево, как модель иерархических отношений. Прямой обход. Пример.

***Сложное дерево*** Т

При прохождении в прямом порядке узлов дерева Т сначала посещается корень n, затем узлы поддерева Т1,…, Тk, так же в прямом порядке.

50. Ориентированный граф. Полустепени вершин. Путь, контур. Сильная и слабая связность орграфа. Матрица смежности орграфа.

**Поустепень исхода**, это число дуг, исходящих из этой вершины.

**Поустепень захода**, это число дуг, входящих в эту вершину

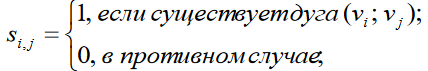
**Сильная связность** орграфа: любые две вершины взаимодостижимы (путь)

**Слабая связность** – не взаимодостижимы (цепь)

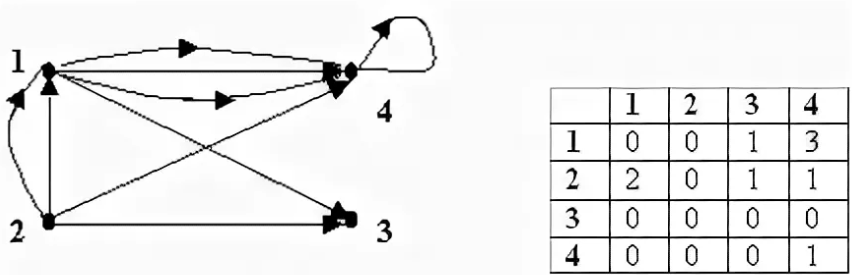
**Путь –** последовательность вершин в орграфе.

**Контур(**цикл**)** — замкнутый путь в орграфе; контур называется простым или элементарным, если ни одна вершина в нем не встречается дважды

**Матрица смежности** – это квадратная матрица размерности n x n, значения элементов которой характеризуются смежностью вершин графа. (n-это количество вершин графа). Элементы матрицы определяются следующим образом

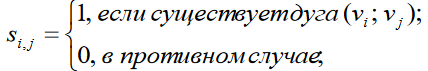


Если рассматривается псевдограф, то вместо единиц ставится число кратных дуг, соединяющих вершины vi и vj и имеющих начало в вершине vi.

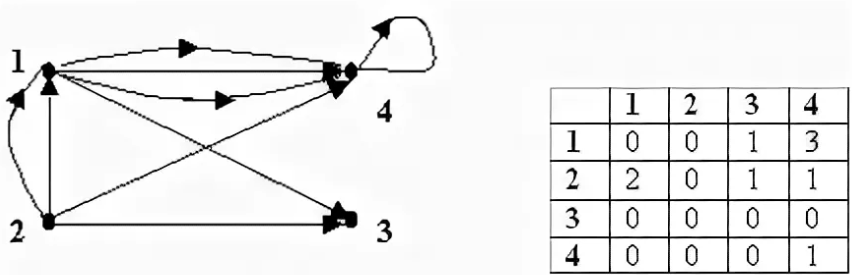


51. Матрица смежности орграфа. Бесконтурный орграф. Доказательство того, что в бесконтурном орграфе есть вершина полустепень исхода которой равна нулю

Матрица смежности – это квадратная матрица размерности n x n, значения элементов которой характеризуются смежностью вершин графа. (n-это количество вершин графа). Элементы матрицы определяются следующим образом



Если рассматривается псевдограф, то вместо единиц ставится число кратных дуг, соединяющих вершины vi и vj и имеющих начало в вершине vi.



Т. В каждом **бесконтурном орграфе** имеется по крайней мере одна вершина, ***полустепень исхода*** которой *равна* ***нулю.***

**Доказательство**. Пусть G = (V ,E ) — бесконтурный орграф и w1 — его произвольная вершина. Если ее полустепень исхода не равна нулю, выберем произвольную вершину w2, такую, что w1w2 ∈ E , затем w3 так, что w2w3 ∈ E и т.д. Поскольку в графе нет вершин с полустепенью исхода равной 0, то процесс этот бесконечен. Поскольку граф конечный, то наступит момент, когда некоторая вершина повторится и мы получим контур, что противоречит условию отсутствия контуров в орграфе.

