**1. Операции над множествами. Диаграммы Эйлера-Венна.**

Image**Объединением** множеств А и В называется множество, состоящее из всех тех элементов, которые принадлежат хотя бы одному из множеств А, В:

Image**Пересечением** множеств А и В называется множество, состоящее из всех тех и только тех элементов, которые принадлежат одновременно как множеству А, и множеству В :

Image**Разностью** множеств А и В называется множество всех тех и только тех элементов А, которые не содержатся в В:

A \bigtriangleup B = \left( A \setminus B \right) \cup \left ( B \setminus A \right).**Симметрической разностью** множеств А и В называется множество элементов этих множеств, которые принадлежат либо только множеству А, либо только множеству В (рис. 4):

Если А – подмножество множества U, то **дополнением** множества А -- это множество состоящее только из тех элементов U, которые не принадлежат А,

т.е. A = U \ A = {x | х U и х A}.

**Диаграмма Эйлера-Венна** — схематичное изображение всех возможных [отношений](https://ru.wikipedia.org/wiki/%D0%9E%D1%82%D0%BD%D0%BE%D1%88%D0%B5%D0%BD%D0%B8%D0%B5_(%D1%82%D0%B5%D0%BE%D1%80%D0%B8%D1%8F_%D0%BC%D0%BD%D0%BE%D0%B6%D0%B5%D1%81%D1%82%D0%B2)) ([объединение](https://ru.wikipedia.org/wiki/%D0%9E%D0%B1%D1%8A%D0%B5%D0%B4%D0%B8%D0%BD%D0%B5%D0%BD%D0%B8%D0%B5_%D0%BC%D0%BD%D0%BE%D0%B6%D0%B5%D1%81%D1%82%D0%B2), [пересечение](https://ru.wikipedia.org/wiki/%D0%9F%D0%B5%D1%80%D0%B5%D1%81%D0%B5%D1%87%D0%B5%D0%BD%D0%B8%D0%B5_%D0%BC%D0%BD%D0%BE%D0%B6%D0%B5%D1%81%D1%82%D0%B2), [разность](https://ru.wikipedia.org/wiki/%D0%A0%D0%B0%D0%B7%D0%BD%D0%BE%D1%81%D1%82%D1%8C_%D0%BC%D0%BD%D0%BE%D0%B6%D0%B5%D1%81%D1%82%D0%B2), [симметрическая разность](https://ru.wikipedia.org/wiki/%D0%A1%D0%B8%D0%BC%D0%BC%D0%B5%D1%82%D1%80%D0%B8%D1%87%D0%B5%D1%81%D0%BA%D0%B0%D1%8F_%D1%80%D0%B0%D0%B7%D0%BD%D0%BE%D1%81%D1%82%D1%8C)) нескольких (часто — трёх) [подмножеств](https://ru.wikipedia.org/wiki/%D0%9F%D0%BE%D0%B4%D0%BC%D0%BD%D0%BE%D0%B6%D0%B5%D1%81%D1%82%D0%B2%D0%BE) [универсального множества](https://ru.wikipedia.org/wiki/%D0%A3%D0%BD%D0%B8%D0%B2%D0%B5%D1%80%D1%81%D0%B0%D0%BB%D1%8C%D0%BD%D0%BE%D0%B5_%D0%BC%D0%BD%D0%BE%D0%B6%D0%B5%D1%81%D1%82%D0%B2%D0%BE). На диаграммах Венна универсальное множество U изображается множеством точек некоторого прямоугольника, в котором располагаются в виде кругов или других простых фигур все остальные рассматриваемые множества.

**2. Понятие прямоего произведениея множеств, его мощность, понятие булеана, мощность булеана.**

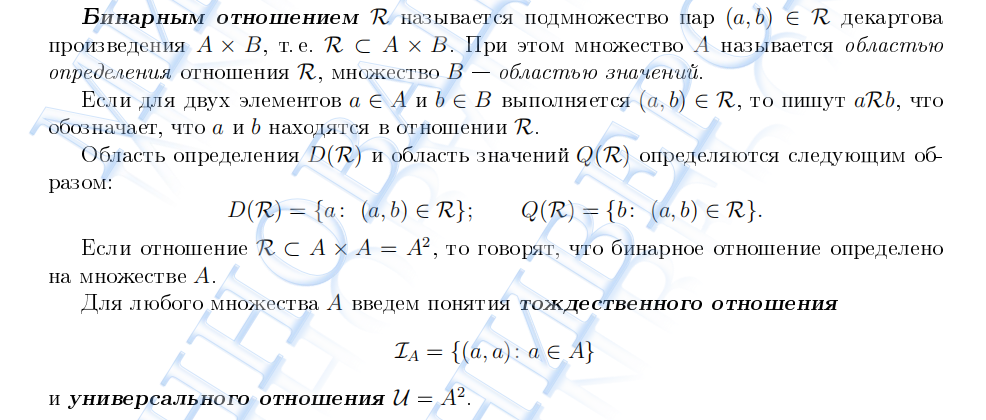
**Прямо́е, или дека́ртово произведе́ние** двух множеств — [множество](https://ru.wikipedia.org/wiki/%D0%9C%D0%BD%D0%BE%D0%B6%D0%B5%D1%81%D1%82%D0%B2%D0%BE), [элементами](https://ru.wikipedia.org/wiki/%D0%AD%D0%BB%D0%B5%D0%BC%D0%B5%D0%BD%D1%82_%D0%BC%D0%BD%D0%BE%D0%B6%D0%B5%D1%81%D1%82%D0%B2%D0%B0) которого являются все возможные [упорядоченные пары](https://ru.wikipedia.org/wiki/%D0%A3%D0%BF%D0%BE%D1%80%D1%8F%D0%B4%D0%BE%D1%87%D0%B5%D0%BD%D0%BD%D0%B0%D1%8F_%D0%BF%D0%B0%D1%80%D0%B0) элементов исходных множеств, такое что на первом месте в паре стоят элементы только первого множества на втором второго и т.д.(для X и Y называется множество Image, элементами которого являются все возможные упорядоченные пары {x, y}, такие, что Image).

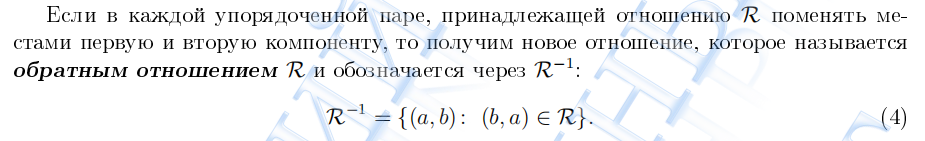
**Мощностью прямого произведения** является произведение мощностей множеств участвующих в операции.

**Булеан** - множество всех возможных подмножеств данного множества.

**Мощность булеана** определяется по формуле 2^n, где n - количество элементов множества. В булеан обязательно входит само множество и пустое множество.

**3. Область определения, область значений отношений; , , обратное отношение.**





Виды бинарных отношений на множестве A

1. Обратное отношение:

http://www.urtt.ru/bib/dataindex/dm/glava_2~.files/image140.gif

1. Дополнение:

http://www.urtt.ru/bib/dataindex/dm/glava_2~.files/image142.gif.

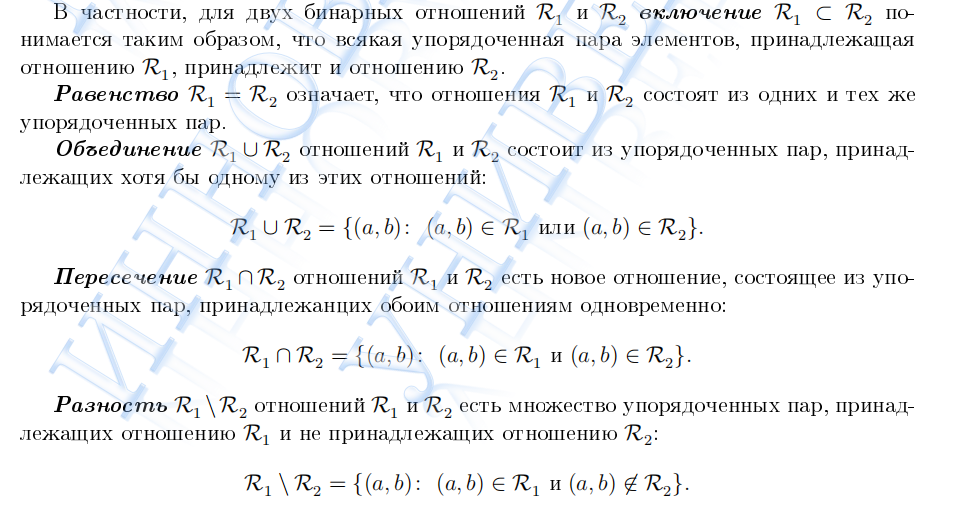
1. Тождественные:

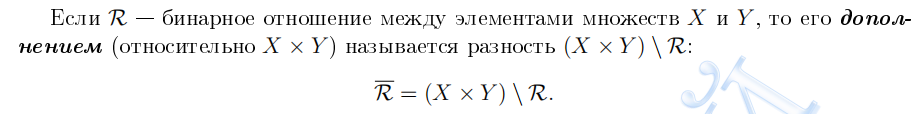
http://www.urtt.ru/bib/dataindex/dm/glava_2~.files/image144.gif.

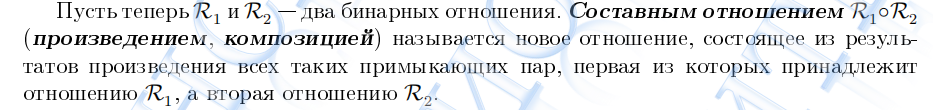
1. Универсальные:

http://www.urtt.ru/bib/dataindex/dm/glava_2~.files/image146.gif.Ua = A^2

**4. Операции над отношениями. Операция композиции отношений.**





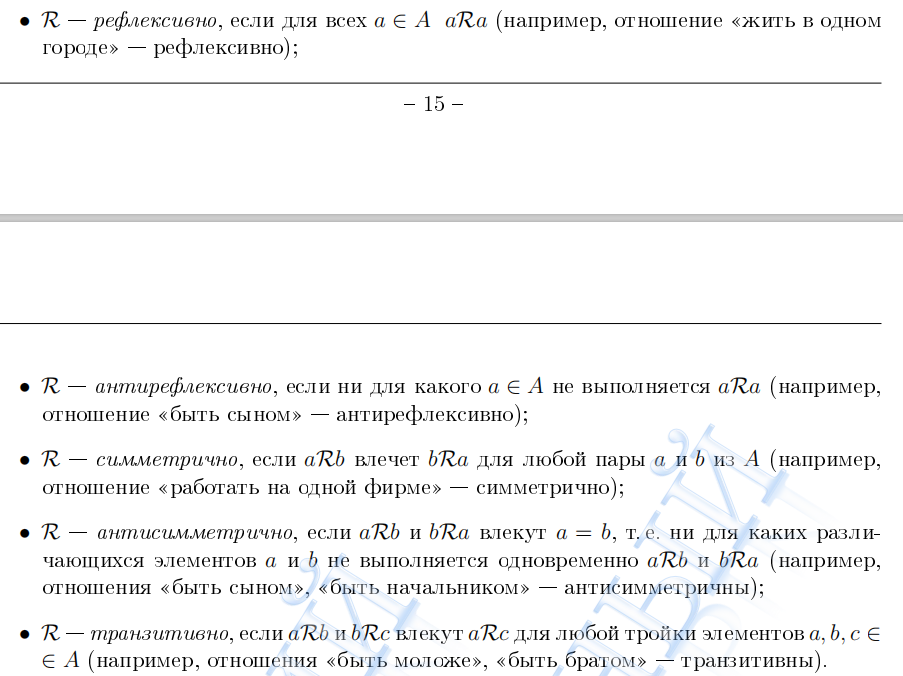


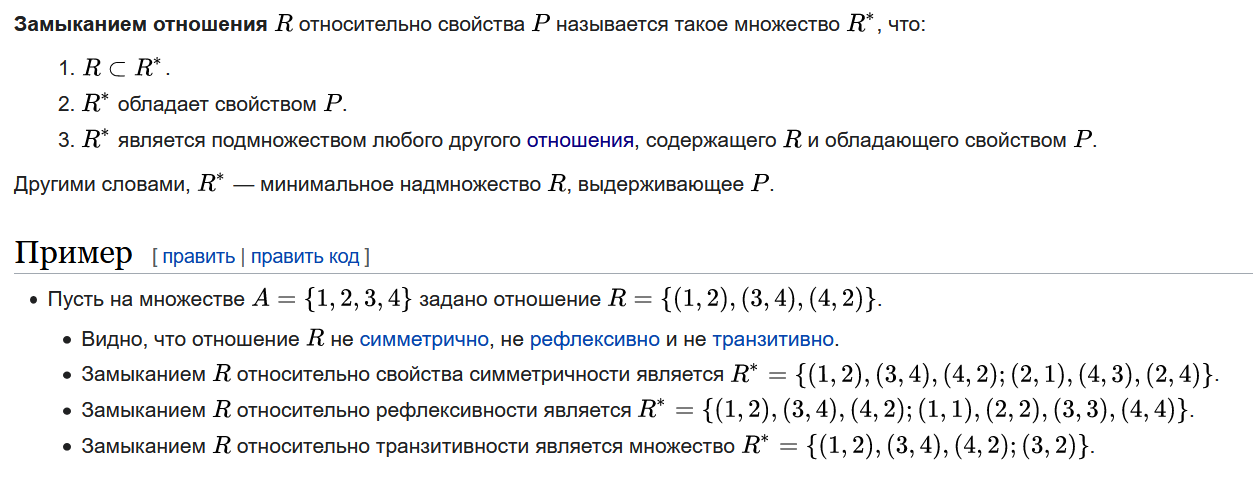
(для нас сначало в R2 потом в R1).

Если два отношения на некотором множестве, заданным перечислением пар, то композицию R1 o R2 можно построить по следующему правилу:

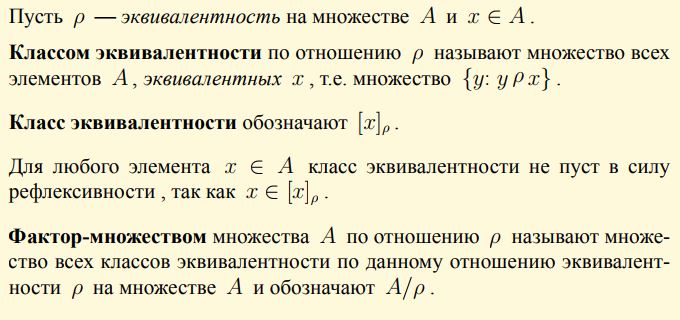
В отношении R2 рассматривают второй каждой пары, ищут в отношении R1 равный ему первый элемент пары. Если находится совпадение – записывают новую пару, состоящую из первого элемента из R2 и второго элемента из R1.

**5. Свойства отношений, замыкание отношений.**





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

****Рефлексивное, симметричное и транзитивное отношение называется **отношением эквивалентности**. Это отношения равенства, подобия, параллельности, равенства по модулю и тд.

**Классом эквивалентности** для элемента а называется множество всех элементов в эквивалентности а.

**Фактор-множество** — множество всех классов эквивалентности для заданного отношения эквивалентности на множестве.

**7. Отношения порядка. Отношение частичного порядка.**

**Отношения порядка** - предусматривает порядок следования эл-ов на мн-ве(>, <, старше/младше).

Рефлексивное, антисимметричное, транзитивное отношение называется отношением ***частичного* (нестрогого) порядка.**

Отношение частичного порядка принято обозначать знаком , это обозначение является условным.

Если отношение R на множестве A является отношением частичного порядка, то  называют частично-упорядоченным множеством или ЧУ-множеством.

Множество , на котором задано отношение *частичного порядка*, называется *полностью упорядоченным* (линейно упорядоченным, цепью), если каждые два элемента *ЧУ-множества*  сравнимы.

Отношение < ⊆ А называется *строгим порядком*, если оно определяется по следующему правилу: (∀ x, y ∈ A) х < у ⇔ х ≤ у и х ≠ у. Отношение строгого порядка не является частичным порядком, так как оно не рефлексивно.

Частичный порядок ≤ ⊆ А 2 называется линейным порядком, если (∀ х, у ∈ А) х ≤ у или у ≤ х. [Бинарное отношение](http://neerc.ifmo.ru/wiki/index.php?title=%D0%91%D0%B8%D0%BD%D0%B0%D1%80%D0%BD%D0%BE%D0%B5_%D0%BE%D1%82%D0%BD%D0%BE%D1%88%D0%B5%D0%BD%D0%B8%D0%B5) R на множестве X называется отношением полного порядка, если оно является отношением линейного порядка и обладает следующим свойством:\forall Y \in X \exists a \in Y \forall b \in Y: aRb.

**8. Понятие *ЧУ* – множества. Сравнимые и несравнимые элементы. Диаграммы *Хассе*. Наибольший /наименьший и максимальный/минимальный элементы *ЧУ*-множества.**

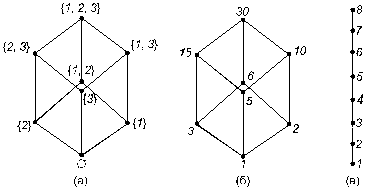
Множество $ S $ называется **частично упорядоченным**, если на нем задано отношение частичного порядка.

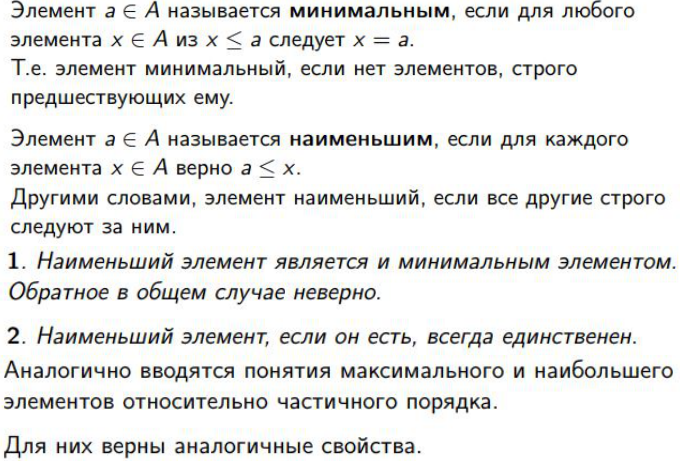
Если в ЧУМ для любых двух элементов а,в выполняется отношение а <= в или а >= в, то такие элементы называются сравнимыми между собой, в противном случае, элементы называются несравнимыми.

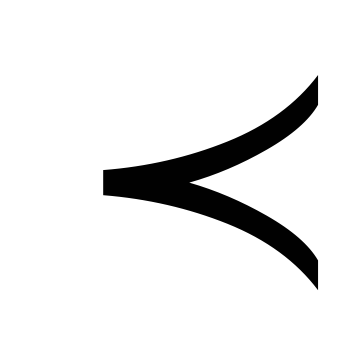
Например, элементы {1} , {1,2} – сравнимы, а {1}, {2} – несравнимы, т.к. ни один не является подмножеством другого.

Множества, в котором любые два элемента сравнимы, называется полностью упорядоченным или линейно упорядоченным, или цепью.

ЧУМ принято изображать диаграммой Хассе. Диаграмма Хассе представляет собой граф, иногда ОР граф. Булев куб:



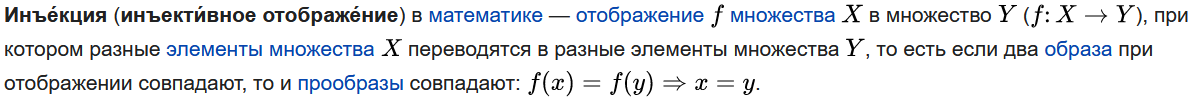


*Диаграммой Хассе* частично упорядоченного множества *Х* называ­ется граф, вершинами которого являются элементы множества *X*, а пара (*х,у*) образует ребро, если элемент *у* покрывает элемент *х*, и такой что, если *х**у*, то *у* рисуют с большей вертикальной координатой чем *х*. **Диаграмма Хассе  — вид**[**диаграмм**](https://ru.wikipedia.org/wiki/%D0%94%D0%B8%D0%B0%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D0%B0)**, используемый для представления конечного**[**частично упорядоченного множества**](https://ru.wikipedia.org/wiki/%D0%A7%D0%B0%D1%81%D1%82%D0%B8%D1%87%D0%BD%D0%BE_%D1%83%D0%BF%D0%BE%D1%80%D1%8F%D0%B4%D0%BE%D1%87%D0%B5%D0%BD%D0%BD%D0%BE%D0%B5_%D0%BC%D0%BD%D0%BE%D0%B6%D0%B5%D1%81%D1%82%D0%B2%D0%BE)**в виде**[**рисунка**](https://ru.wikipedia.org/wiki/%D0%92%D0%B8%D0%B7%D1%83%D0%B0%D0%BB%D0%B8%D0%B7%D0%B0%D1%86%D0%B8%D1%8F_%D0%B3%D1%80%D0%B0%D1%84%D0%BE%D0%B2)**его**[**транзитивного сокращения**](https://ru.wikipedia.org/wiki/%D0%A2%D1%80%D0%B0%D0%BD%D0%B7%D0%B8%D1%82%D0%B8%D0%B2%D0%BD%D0%BE%D0%B5_%D1%81%D0%BE%D0%BA%D1%80%D0%B0%D1%89%D0%B5%D0%BD%D0%B8%D0%B5)**.**

**9. Понятие функции. Свойства функций: инъекция, сюръекция, биекция.**

Функции это специальное отношение(R: A->B), в кот для каждого aэA существует единственный элемент bэB, такой что (a,b)эR.

Отношение является функцией если сумма каждой строчки в таблице = 1.



Инъекция – если разиличным зн Х ставятся в соответствии различные зн Y.

Сюръекция – когда для каждого Y зн сущ нект зн Х(т.е. весь Y задействован).

Функция инъективна если f(a1) = f(a2) => a1 = a2, если сумма каждого столбца не больше 1. Функция сурьективна если сумма каждого столбца не равна 0 (задействовано все второе множество). Если функция и инъективна и суръективна, то она биективна.

**10. Счетно бесконечные множества. Несчетные   бесконечные множества.**

**Счётное множество** — [бесконечное множество](https://ru.wikipedia.org/wiki/%D0%91%D0%B5%D1%81%D0%BA%D0%BE%D0%BD%D0%B5%D1%87%D0%BD%D0%BE%D0%B5_%D0%BC%D0%BD%D0%BE%D0%B6%D0%B5%D1%81%D1%82%D0%B2%D0%BE), элементы которого возможно пронумеровать [натуральными числами](https://ru.wikipedia.org/wiki/%D0%9D%D0%B0%D1%82%D1%83%D1%80%D0%B0%D0%BB%D1%8C%D0%BD%D0%BE%D0%B5_%D1%87%D0%B8%D1%81%D0%BB%D0%BE). Более формально: [множество](https://ru.wikipedia.org/wiki/%D0%9C%D0%BD%D0%BE%D0%B6%D0%B5%D1%81%D1%82%D0%B2%D0%BE) X является счётным, если существует [биекция](https://ru.wikipedia.org/wiki/%D0%91%D0%B8%D0%B5%D0%BA%D1%86%D0%B8%D1%8F) со множеством натуральных чисел: X ↔ N, другими словами, счётное множество — это множество, [равномощное](https://ru.wikipedia.org/wiki/%D0%A0%D0%B0%D0%B2%D0%BD%D0%BE%D0%BC%D0%BE%D1%89%D0%BD%D0%BE%D1%81%D1%82%D1%8C) множеству натуральных чисел.

**Несчётное мно́жество** — [бесконечное множество](https://ru.wikipedia.org/wiki/%D0%91%D0%B5%D1%81%D0%BA%D0%BE%D0%BD%D0%B5%D1%87%D0%BD%D0%BE%D0%B5_%D0%BC%D0%BD%D0%BE%D0%B6%D0%B5%D1%81%D1%82%D0%B2%D0%BE), не являющееся [счётным](https://ru.wikipedia.org/wiki/%D0%A1%D1%87%D1%91%D1%82%D0%BD%D0%BE%D0%B5_%D0%BC%D0%BD%D0%BE%D0%B6%D0%B5%D1%81%D1%82%D0%B2%D0%BE).

Таким образом, любое множество является либо конечным, либо счётным, либо несчётным.

Некоторые эквивалентные определения несчётности для множества X:

1)не существует [инъективного](https://ru.wikipedia.org/wiki/%D0%98%D0%BD%D1%8A%D0%B5%D0%BA%D1%86%D0%B8%D1%8F_(%D0%BC%D0%B0%D1%82%D0%B5%D0%BC%D0%B0%D1%82%D0%B8%D0%BA%D0%B0)) отображения X во [множество натуральных чисел N;](https://ru.wikipedia.org/wiki/%D0%9C%D0%BD%D0%BE%D0%B6%D0%B5%D1%81%D1%82%D0%B2%D0%BE_%D0%B2%D1%81%D0%B5%D1%85_%D0%BD%D0%B0%D1%82%D1%83%D1%80%D0%B0%D0%BB%D1%8C%D0%BD%D1%8B%D1%85_%D1%87%D0%B8%D1%81%D0%B5%D0%BB)

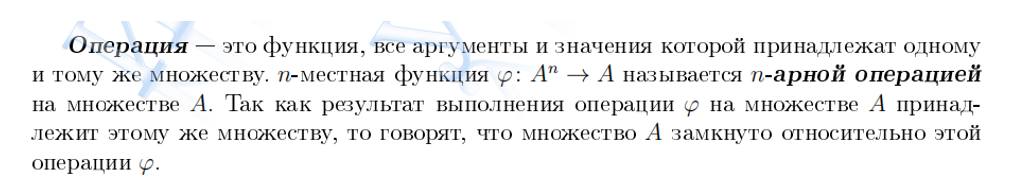
2)X не [пустое](https://ru.wikipedia.org/wiki/%D0%9F%D1%83%D1%81%D1%82%D0%BE%D0%B5_%D0%BC%D0%BD%D0%BE%D0%B6%D0%B5%D1%81%D1%82%D0%B2%D0%BE), и для каждой нумерованной последовательности элементов X существует по крайней мере один элемент X, не входящий в неё;

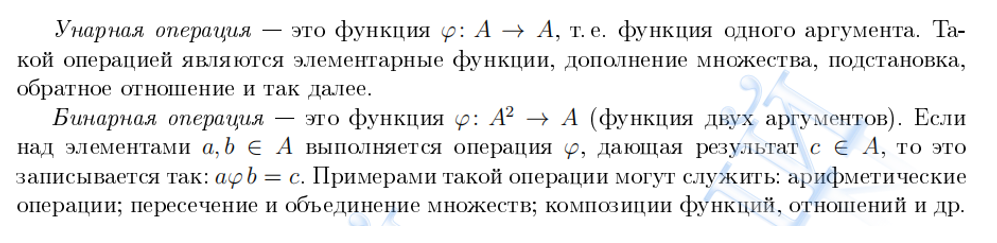
3)иными словами: X непусто, и не существует [сюръективного](https://ru.wikipedia.org/wiki/%D0%A1%D1%8E%D1%80%D1%8A%D0%B5%D0%BA%D1%86%D0%B8%D1%8F) отображения множества натуральных чисел N на X;

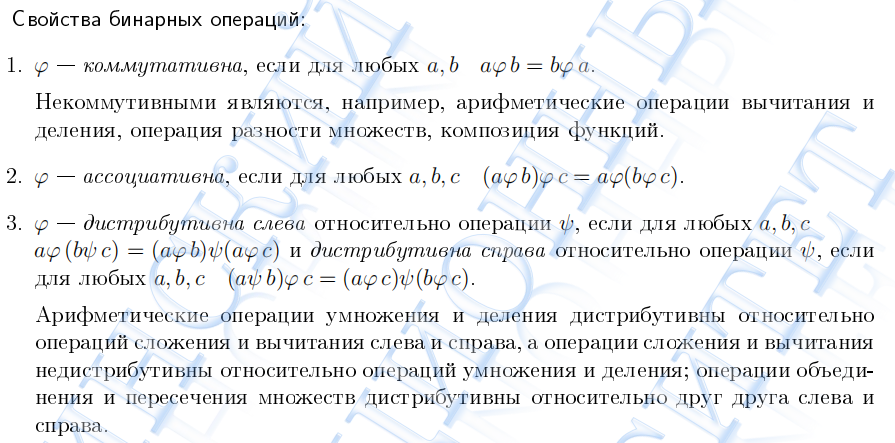
**Счетные множества:** [Простые числа](http://ru.wikipedia.org/wiki/%D0%9F%D1%80%D0%BE%D1%81%D1%82%D1%8B%D0%B5_%D1%87%D0%B8%D1%81%D0%BB%D0%B0), [Натуральные числа](http://ru.wikipedia.org/wiki/%D0%9D%D0%B0%D1%82%D1%83%D1%80%D0%B0%D0%BB%D1%8C%D0%BD%D1%8B%D0%B5_%D1%87%D0%B8%D1%81%D0%BB%D0%B0), [Целые числа](http://ru.wikipedia.org/wiki/%D0%A6%D0%B5%D0%BB%D1%8B%D0%B5_%D1%87%D0%B8%D1%81%D0%BB%D0%B0), [Рациональные числа](http://ru.wikipedia.org/wiki/%D0%A0%D0%B0%D1%86%D0%B8%D0%BE%D0%BD%D0%B0%D0%BB%D1%8C%D0%BD%D1%8B%D0%B5_%D1%87%D0%B8%D1%81%D0%BB%D0%B0), [Алгебраические числа](http://ru.wikipedia.org/wiki/%D0%90%D0%BB%D0%B3%D0%B5%D0%B1%D1%80%D0%B0%D0%B8%D1%87%D0%B5%D1%81%D0%BA%D0%B8%D0%B5_%D1%87%D0%B8%D1%81%D0%BB%D0%B0), [Вычислимые числа](http://ru.wikipedia.org/wiki/%D0%92%D1%8B%D1%87%D0%B8%D1%81%D0%BB%D0%B8%D0%BC%D1%8B%D0%B5_%D1%87%D0%B8%D1%81%D0%BB%D0%B0), [Арифметические числа](http://ru.wikipedia.org/wiki/%D0%90%D1%80%D0%B8%D1%84%D0%BC%D0%B5%D1%82%D0%B8%D1%87%D0%B5%D1%81%D0%BA%D0%B8%D0%B5_%D1%87%D0%B8%D1%81%D0%BB%D0%B0), Множество всех слов над конечным алфавитом, Любое бесконечное множество точек на плоскости, все попарные расстояния между элементами которого рациональны.

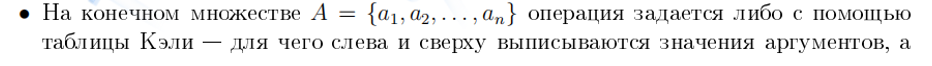
**Несчетные множества:** [Вещественные числа](http://ru.wikipedia.org/wiki/%D0%92%D0%B5%D1%89%D0%B5%D1%81%D1%82%D0%B2%D0%B5%D0%BD%D0%BD%D1%8B%D0%B5_%D1%87%D0%B8%D1%81%D0%BB%D0%B0), [Комплексные числа](http://ru.wikipedia.org/wiki/%D0%9A%D0%BE%D0%BC%D0%BF%D0%BB%D0%B5%D0%BA%D1%81%D0%BD%D1%8B%D0%B5_%D1%87%D0%B8%D1%81%D0%BB%D0%B0), [Числа Кэли](http://ru.wikipedia.org/wiki/%D0%A7%D0%B8%D1%81%D0%BB%D0%B0_%D0%9A%D1%8D%D0%BB%D0%B8)**.**

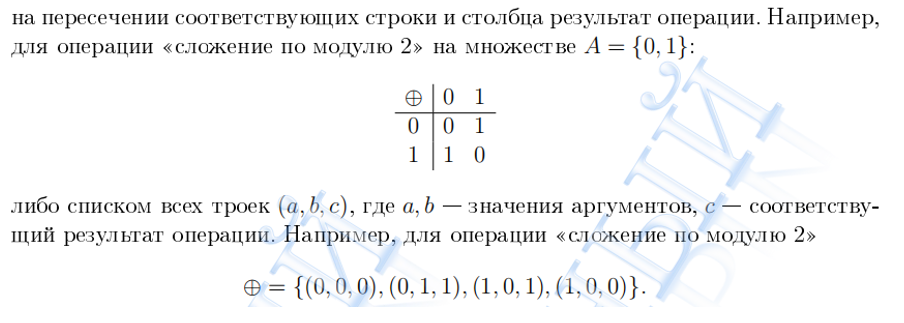
**11. Понятие операции. Таблица Кэли для задания операции. Свойства операций.**

****

****







Говорят, что на М определена **бинарная алгебраическая операция**, если всякой упорядоченной паре элементов множества М по некоторому закону ставится в соответствие вполне определенный элемент этого же множества.

Для того, чтобы задать на [множестве](http://fxdx.ru/page/otnoshenie-porjadka-sledovanija-na-mnozhestve) А алгебраическую операцию \* необходимо выполнить два условия:

1) нужно определить правило, по которому любым двум элементам х и у множества А ставился бы в соответствие единственный для этой пары элементов элемент http://fxdx.ru/site/fxdx_ru/uploads/ag1/image033.gif;

2) этот элемент http://fxdx.ru/site/fxdx_ru/uploads/ag1/image033.gif должен принадлежать множеству А. В этом случае множество А замкнуто относительно данной операции \*.

Табличный способ задания операции **таблица Кэли**. Слева и сверху квадратной таблицы выписывают все элементы множества. На пересечении строки, соответствующей элементу *a*, и столбца, соответствующего элементу *b*, записывают результат операции над *a* и *b*. Если операцию называют сложением, то таблицу Кэли называют таблицей сложения. Если операцию называют умножением, то таблицей умножения.

Алгебраическая операция \*, определенная на [множестве](http://fxdx.ru/page/otnoshenie-porjadka-sledovanija-na-mnozhestve) А называется **коммутативной**, если она подчиняется закону коммутативности, т.е. для любых [двух](http://fxdx.ru/page/orientacija-dvuh-koordinatnyh-osej-na-ploskosti-obshhaja-i-prjamougolnaja-dekartovaja-sistema-koordinat-na-ploskosti) элементов х и у множества А выполняется равенство: х\*у = у\*х.

Алгебраическая операция \*, определенная на [множестве](http://fxdx.ru/page/otnoshenie-porjadka-sledovanija-na-mnozhestve) А называется **ассоциативной**, если она подчиняется закону ассоциативности, т.е. для любых [трех](http://fxdx.ru/page/vzaimnoe-raspolozhenie-treh-ploskostej) элементов х, у, z множества А выполняется равенство: http://fxdx.ru/site/fxdx_ru/uploads/ag1/image039.gif.

Пусть на [множестве](http://fxdx.ru/page/otnoshenie-porjadka-sledovanija-na-mnozhestve) А определены две алгебраических операции, которые обозначим символами \* и http://fxdx.ru/site/fxdx_ru/uploads/ag1/image040.gif. Операция \* **дистрибутивна** относительно операции http://fxdx.ru/site/fxdx_ru/uploads/ag1/image040.gif, если http://fxdx.ru/site/fxdx_ru/uploads/ag1/image041.gif верны два равенства:

http://fxdx.ru/site/fxdx_ru/uploads/ag1/image042.gif и http://fxdx.ru/site/fxdx_ru/uploads/ag1/image043.gif.

**12. Принцип математической индукции.**

Метод математической индукции является важным способом доказательства предложений (утверждений), зависящих от натурального аргумента.

Метод математической индукции состоит в следующем:

Предложение (утверждение) *P*(*n*), зависящее от натурального числа *n*, справедливо для любого натурального *n* если:

1. *P*(1) является истинным предложением (утверждением);
2. *P*(*n*) остается истинным предложением (утверждением), если *n* увеличить на единицу, то есть *P*(*n* + 1) - истинное предложение (утверждение).

Таким образом метод математической индукции предполагает два этапа:

1. Этап проверки: проверяется, истинно ли предложение (утверждение) *P*(1).
2. Этап доказательства: предполагается, что предложение *P*(*n*) истинно, и доказывается истинность предложения *P*(*n* + 1) (*n* увеличено на единицу).

**13. Классы булевых функций. Теорема *Поста-Яблонского*.**

Смысл теоремы сводится к следующему:



Набор функций двух переменных, является функционально полной системой, если хотя бы одна из его функций:

1. Не сохраняет «0»;
2. Не сохраняет «1» ;
3. Не самодвойственна;
4. Не монотонна;
5. Не линейна.

Эти свойства логических функций разбивают их на классы сохраняющих и не сохраняющих 0; сохраняющих и не сохраняющих 1; самодвойственных и не самодвойственных и т.д. При этом суперпозиция логических функций одного класса дает функцию того же класса.

**14. СДНФ,  СКНФ булевой функции, Полином *Жегалкина*.**

СДНФ - равносильная ей формула, представляющая собой дизъюнкцию элементарных конъюнкций, обладающая свойствами

1. Каждое логическое слагаемое формулы содержит все переменные, входящие в функцию F(x1,x2,...xn)

2. Все логические слагаемые формулы различны

3. Ни одно логическое слагаемое не содержит переменную и её отрицание

4. Ни одно логическое слагаемое формулы не содержит одну и ту же переменную дважды.

Чтобы получить СКНФ, надо взять все наборы, на которых значение функции равно 0 и записать для каждого из них дизъюнкцию переменных и их отрицаний. Если в наборе значение переменной равно 0 – то переменную надо взять без отрицания, если 1 – с отрицанием. Из получившихся дизъюнкций надо построить конъюнкцию.

СКНФ - Совершенная конъюнктивная нормальная форма формулы (СКНФ) это равносильная ей формула, представляющая собой конъюнкцию элементарных дизъюнкций, удовлетворяющая свойствам:

1. Все элементарные дизъюнкции содержат все переменные, входящие в функцию F(x1,x2,...xn)

2. Все элементарные дизъюнкции различны

3. Каждая элементарная дизъюнкция содержит переменную один раз

4. Ни одна элементарная дизъюнкция не содержит переменную и её отрицание

Чтобы получить совершенную дизъюнктивную нормальную форму (**СДНФ)**, надо взять все наборы, на которых значение функции равно 1 и записать для каждого из них конъюнкцию переменных и их отрицаний. Если в наборе значение переменной 0 – то переменную надо взять с отрицанием, если 1 – без отрицания. Из получившихся конъюнкций надо построить дизъюнкцию.

Полином Жегалкина представляет собой полином, коэффициентами которого являются числа 0 или 1, причём в качестве операции умножения и сложения используются операции конъюнкции и сумма по модулю 2.

Конъюнкция - результат равен 1, если все операнды равны 1, во всех остальных случаях результат равен 0 (птичка вверх)0 {\displaystyle 0} .

Дизъюнкция - Таким образом, результат равен 00 {\displaystyle 0} , если все операнды равны 0, во всех остальных случаях результат равен 1(птичка вниз)1 {\displaystyle 1} .

**Полином Жегалкина можно построить различными способами: используя таблицу истинности (через СДНФ), методом неопределенных коэффициентов, методом треугольника.**

**15. Правило суммы. Правило произведения.Формула включений и исключений в комбинаторике.**

Правило произведения. Пусть объект A можно выбрать **n** способами и после каждого такого выбора объект B можно выбрать **m** способами. Тогда выбор пары (A,B) можно осуществить **mn** способами.

Правило суммы. Пусть некоторый объект A можно выбрать **n** различными способами, а другой объект B можно выбрать **m** способами. Тогда существует **n+m** способов выбрать либо объект A, либо объект B.

Формула включений-исключений (или принцип включений-исключений) — [комбинаторная](https://ru.wikipedia.org/wiki/%D0%9A%D0%BE%D0%BC%D0%B1%D0%B8%D0%BD%D0%B0%D1%82%D0%BE%D1%80%D0%B8%D0%BA%D0%B0) формула, позволяющая определить [мощность](https://ru.wikipedia.org/wiki/%D0%9C%D0%BE%D1%89%D0%BD%D0%BE%D1%81%D1%82%D1%8C_%D0%BC%D0%BD%D0%BE%D0%B6%D0%B5%D1%81%D1%82%D0%B2%D0%B0) [объединения](https://ru.wikipedia.org/wiki/%D0%9E%D0%B1%D1%8A%D0%B5%D0%B4%D0%B8%D0%BD%D0%B5%D0%BD%D0%B8%D0%B5_%D0%BC%D0%BD%D0%BE%D0%B6%D0%B5%D1%81%D1%82%D0%B2) конечного числа [конечных множеств](https://ru.wikipedia.org/wiki/%D0%9A%D0%BE%D0%BD%D0%B5%D1%87%D0%BD%D0%BE%D0%B5_%D0%BC%D0%BD%D0%BE%D0%B6%D0%B5%D1%81%D1%82%D0%B2%D0%BE), которые в общем случае могут [пересекаться](https://ru.wikipedia.org/wiki/%D0%9F%D0%B5%D1%80%D0%B5%D1%81%D0%B5%D1%87%D0%B5%D0%BD%D0%B8%D0%B5_%D0%BC%D0%BD%D0%BE%D0%B6%D0%B5%D1%81%D1%82%D0%B2) друг с другом.



**16. Понятие выборки, примеры различных выборок.**

Пусть A={a1, ..., an} – множество из n элементов. Набор элементов ai1, ..., air, r ≥ 1, называется выборкой объема r из n элементов, или (n,r)-выборкой.

Выборка называется упорядоченной, если порядок следования элементов в ней задан.  
Две упорядоченные выборки, отличающиеся лишь порядком следования элементов, считаются различными.  
Если порядок следования элементов не является существенным, то выборка называется неупорядоченной. В выборках могут допускаться или не допускаться повторения элементов.

Примеры: выборка имен из списка, покупатели в магазине, соседи, база данных по потребителям, случайно названный телефонный номер для телефонного опроса.

**17. Размещения.**

В [комбинаторике](https://ru.wikipedia.org/wiki/%D0%9A%D0%BE%D0%BC%D0%B1%D0%B8%D0%BD%D0%B0%D1%82%D0%BE%D1%80%D0%B8%D0%BA%D0%B0) размеще́нием (из n по k) называется [упорядоченный набор](https://ru.wikipedia.org/wiki/%D0%9A%D0%BE%D1%80%D1%82%D0%B5%D0%B6_(%D0%BC%D0%B0%D1%82%D0%B5%D0%BC%D0%B0%D1%82%D0%B8%D0%BA%D0%B0)) из k различных элементов из некоторого [множества](https://ru.wikipedia.org/wiki/%D0%9C%D0%BD%D0%BE%D0%B6%D0%B5%D1%81%D1%82%D0%B2%D0%BE) различных n элементов.

Размещением из n элементов по k называется упорядоченная (n,k)-выборка

Пример 1: ⟨ 1 , 3 , 2 , 5 ) — это 4-элементное размещение из 6-элементного множества { 1 , 2 , 3 , 4 , 5 , 6 }.

Пример 2: некоторые размещения элементов множества { 1 , 2 , 3 , 4 , 5 , 6 } по 2: ⟨ 1 , 2 ⟩⟨ 1 , 3 ⟩⟨ 1 , 4 ⟩⟨ 1 , 5 ⟩… ⟨ 2 , 1 ⟩ ⟨ 2 , 3 ⟩⟨ 2 , 4 ⟩ … ⟨ 2 , 6 ⟩ …

В отличие от [сочетаний](https://ru.wikipedia.org/wiki/%D0%A1%D0%BE%D1%87%D0%B5%D1%82%D0%B0%D0%BD%D0%B8%D0%B5), размещения учитывают порядок следования предметов. Так, например, наборы ⟨ 2 , 1 , 3 ⟩и ⟨ 3 , 2 , 1 ⟩являются различными, хотя состоят из одних и тех же элементов { 1 , 2 , 3 } (то есть совпадают как сочетания).

Заполнить ряд - значит надо поместить на каком-нибудь месте этого ряда какой-либо объект из данного множества (причем каждый объект можно использовать всего лишь один раз). Ряд, заполненный объектами данного множества, называется размещением , т.е мы разместили объекты на данных местах.

*(n,k)-размещением с повторениями* называется упорядоченная *(n,k)* выборка, элементы в которой могут повторяться;

*(n,k)-размещением без повторений* называется упорядоченная *(n,k)* выборка, элементы которой попарно различны;

Число *(n,k)-размещений с повторениями*обозначается, как ивычисляется по формуле: ,.

Число всех *(n,k)-размещений без повторений* обозначается через . Подсчитаем их число. 

На первое место выборки мы можем поставить любой из *n* элементов множества, поскольку повторения не разрешены, то на второе место мы можем поставить любой из *(n-1)* оставшихся элементов. На третье место - из *(n-2)* и так далее, вплоть до *k* места, куда можно поставить любой из *(n-k+1)*элементов. По правилу произведения имеем:

 , 

**18. Перестановки. Разупорядочения*.***

Перестановка–это упорядоченный набор из n различных элементов множества различных n элементов.

Размещения без повторений из *n* элементов по *n* называются *перестановками*из *n* элементов без повторений или *перестановками* множества *X*. Их число обозначается и вычисляется по формуле:

, 

Группы элементов, состоящие из одних и тех же элементов и отличающиеся друг от друга только их порядком, называются перестановками этих элементов.   
Число всевозможных перестановок n элементов обозначается Pn. Как это будет ниже показано, оно равно произведению всех натуральных чисел от 1 до n. Для краткости это произведение обозначают символом n! (читается "эн факториал").

Множество всех перестановок из заданных k элементов разбивается на группы, в каждой из которых на первом месте стоит один и тот же элемент. Число таких групп равно k - числу всех элементов.

Разупорядочиванием называется перестановка n различных упорядоченных символов, при которой ни один из символов не остаётся на своем месте.

Количество разупорядочений на *n* различных упорядоченных символах обозначается 

В общем случае, для *n>1* количество разупорядочений на *n* символах вычисляется по формуле:

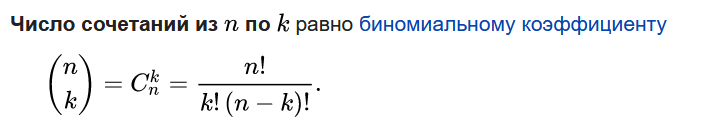


**19. Сочетания.**

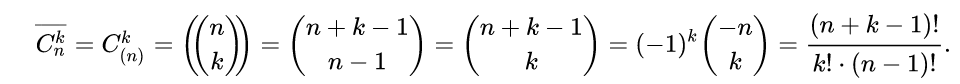
В [комбинаторике](https://ru.wikipedia.org/wiki/%D0%9A%D0%BE%D0%BC%D0%B1%D0%B8%D0%BD%D0%B0%D1%82%D0%BE%D1%80%D0%B8%D0%BA%D0%B0) сочетанием из n по k называется набор k элементов, выбранных из данного [множества](https://ru.wikipedia.org/wiki/%D0%9C%D0%BD%D0%BE%D0%B6%D0%B5%D1%81%D1%82%D0%B2%D0%BE), содержащего n различных элементов.

Наборы, отличающиеся только порядком следования элементов (но не составом), считаются одинаковыми, этим сочетания отличаются от [размещений](https://ru.wikipedia.org/wiki/%D0%A0%D0%B0%D0%B7%D0%BC%D0%B5%D1%89%D0%B5%D0%BD%D0%B8%D0%B5).

Так, например, наборы (3-элементные сочетания, подмножества, k = 3 ) {2, 1, 3} и {3, 2, 1} 6-элементного множества {1, 2, 3, 4, 5, 6} ( n = 6 ) являются одинаковыми (в то время как размещения были бы разными) и состоят из одних и тех же элементов {1,2,3}.



Сочетанием с повторениями называются наборы, в которых каждый элемент может участвовать несколько раз. В частности, количество [монотонных](https://ru.wikipedia.org/wiki/%D0%9C%D0%BE%D0%BD%D0%BE%D1%82%D0%BE%D0%BD%D0%BD%D0%B0%D1%8F_%D1%84%D1%83%D0%BD%D0%BA%D1%86%D0%B8%D1%8F) неубывающих функций из множества { 1 , 2 , … , k } в множество { 1 , 2 , … , n } равно числу сочетаний с повторениями из n по k.



**20. Рекуррентные соотношения. Числа *Фибоначчи*,числа *Каталана*.**

Последовательность может быть описана *неявно*, *рекурсивно* – с помощью *рекуррентного соотношения*.

Последовательность  называется *рекуррентной порядка* *k* () , если существует формула , с помощью которой каждый последующий элемент последовательности  вычисляется через предыдущие элементы . Данная формула называется *рекуррентным соотношением порядка k* . Заметим, что *первые* *k* *элементов* последовательности должны быть заданы.

Например, последовательность чисел *Фибоначчи 0, 1, 1, 2, 3, 5, 8, 13,…*задается рекуррентным соотношением *2* порядка, данную числовую последовательность мы обозначили как :

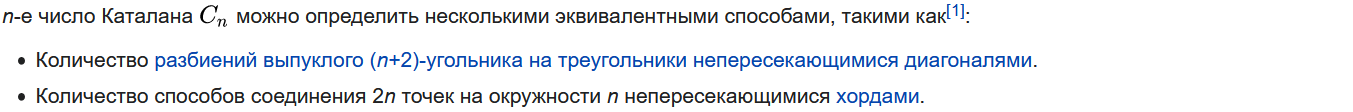
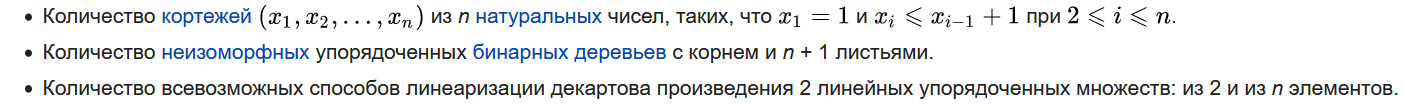
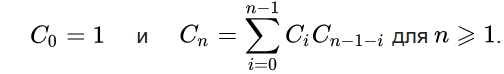
 *(1)*

Последовательность *Фибоначчи* является частным случаем *линейных однородных рекуррентных**соотношений с постоянными коэффициентами порядка k*. В общем случае они имеют вид:

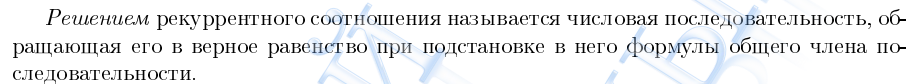
, , *(2)*

где  - постоянные коэффициенты,

– *k* начальных условий, (начальные значения последовательности ).

**21. Решение рекуррентных соотношений. Формула  *Бине****.*

  
Существует общий метод решения (т.е. отыскания  как функции *n*) данных рекуррентных соотношений. Мы рассмотрим методику решения *линейных однородных рекуррентных**соотношений с постоянными коэффициентами порядка k* на примере задачи *Фибоначчи* – *(1)*.

Решение рекуррентного соотношения будем искать в виде:

.

Подставив это решение в рекуррентное соотношение *(1)*, получим:

.

Разделив обе части этого соотношения на , имеем:



Или:

.

Это уравнение называется *характеристическим* для данного рекуррентного соотношения. Решив квадратное уравнение, получим:

.

Общее решение нашего рекуррентного соотношения имеет вид:

, *(3)*

подставив в *(3)* корни характеристического уравнения, имеем:



Решение рекуррентного соотношения *k* –*го* порядка называется *общим*, если оно зависит от *k* произвольных постоянных  и путем подбора этих постоянных можно получить любое частное решение данного соотношения. Используем начальные условия для нахождения коэффициентов :



.

Решая систему, находим:

 .

 , 

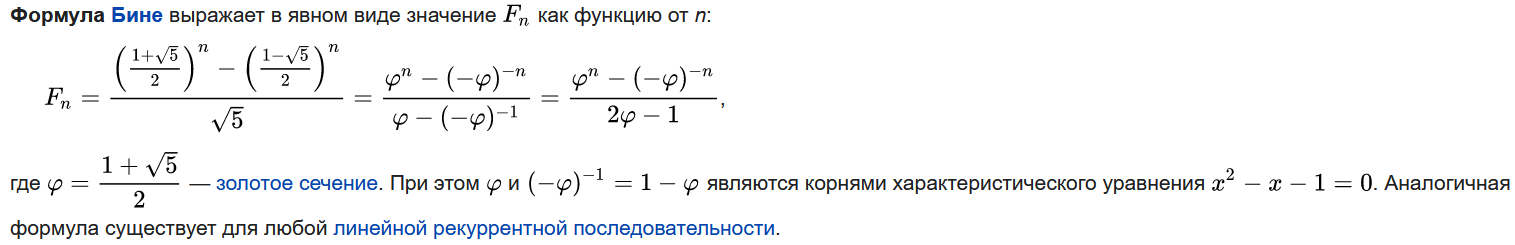
Подставив полученные коэффициенты в *(3)*, получим решение рекуррентного соотношения *(1)*для последовательности чисел *Фибоначчи*.

,

формулу *Бине* для вычислений чисел *Фибоначчи*. Это выражение при всех натуральных значениях *n* принимает целые значения.

Рассмотрев ситуацию, когда характеристическое уравнение имело два различных действительных корня, перейдем к случаю кратных корней. В этом случае общее решение для рекуррентного соотношения *2-го* порядка будет иметь вид:

.



**22. Рекуррентное соотношение для числа сочетаний. Треугольник *Паскаля*.**

img-gMd518

Основная закономерность образования строк состоит в следующем: каждое число в треугольнике Паскаля равно сумме двух чисел, стоящих над ним в предыдущей строке (5 = 1 + 4; 10 = 4 + 6; 6 = 3 = 3 и т.д.). Или то же в строгой формулировке: сумма двух соседних коэффициентов в разложении (*а* + *b*)*n* равна определённому коэффициенту в разложении (*а* + *b*)*n*+1.

Треугольник Паскаля – это треугольник из целых чисел, по боковым сторонам которого стоят единицы, а каждое число внутри треугольника равно сумме двух чисел стоящих над ним.  
Треугольник Паскаля — бесконечная таблица биномиальных коэффициентов, имеющая треугольную форму.

**Треугольником Паскаля** называется бесконечная треугольная таблица, в которой на вершине и по боковым сторонам стоят единицы, каждое из остальных чисел равно сумме двух чисел, стоящих над ним в предшествующей строке.

   1

   1 1

1 2 1

1 3 3 1

1 4 6 4 1

     1 5 10 10 5 1

     1 6 15 20 15 6 1

СВОЙСТВА ТРЕУГОЛЬНИКА ПАСКАЛЯ

Сумма чисел n-ной строки (отсчет ведется с нуля) треугольника Паскаля равна 2n. Действительно, при переходе от каждой строки к следующей сумма членов удваивается, а для нулевой строки она равна 20=1.

Все строки треугольника Паскаля симметричны. Потому что при переходе от каждой строки к следующей свойство симметричности сохраняется, а нулевая строка симметрична.

Каждое число в треугольнике Паскаля равно Cnk, где n — номер строки, k — номер (отсчет ведется с нуля) элемента в строке.

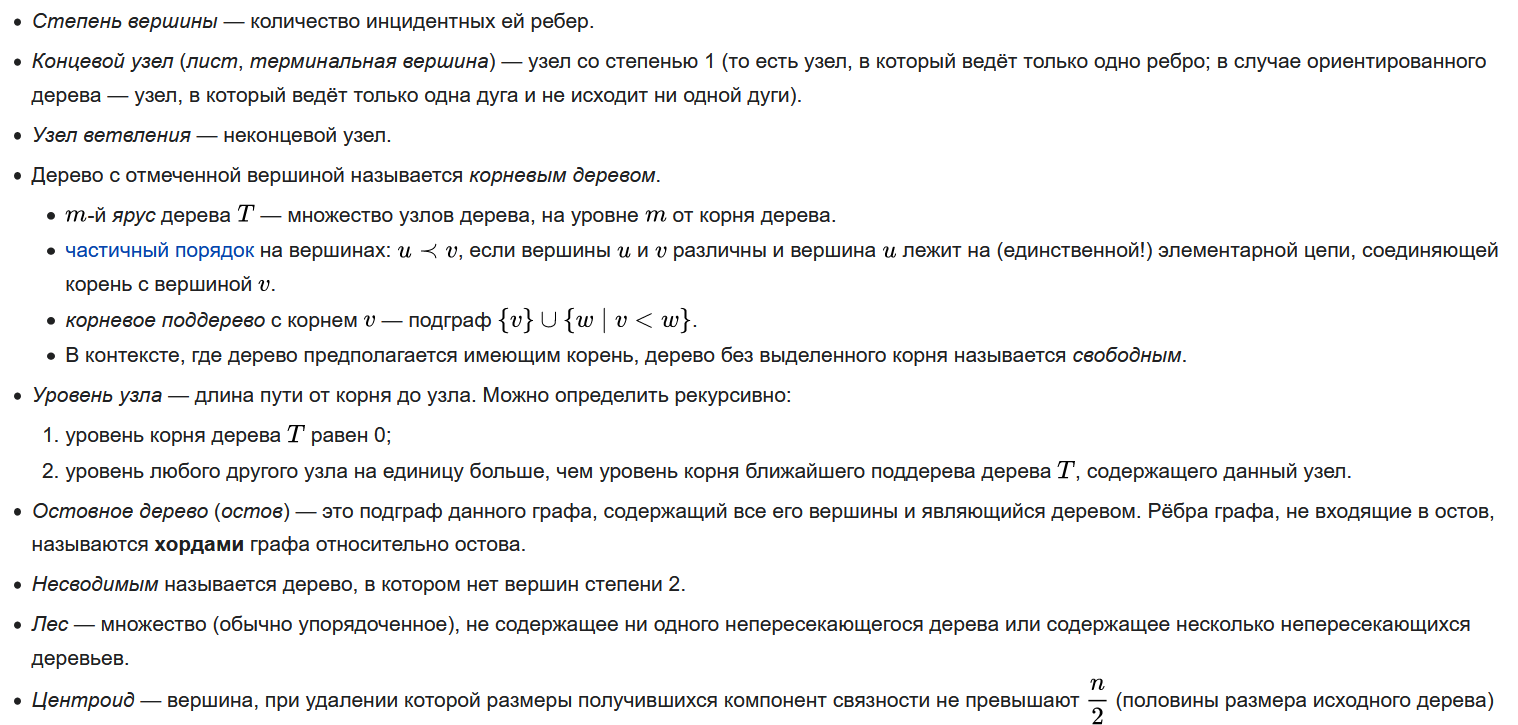
Каждое число треугольника Паскаля, уменьшенное на единицу, равно сумме всех чисел, заполняющих параллелограмм, ограниченный диагоналями, на пересечении которых находится этот элемент.

Вдоль диагоналей, параллельных сторонам треугольника, выстроены треугольные числа, тетраэдрические числа и т.д.

Если посчитать для каждой восходящей диагонали треугольника Паскаля сумму всех стоящих на этой диагонали чисел, то получится соответствующее число Фибоначчи.

**23. Понятие дерева, основные понятия. Остовные деревья, минимальный остов.**

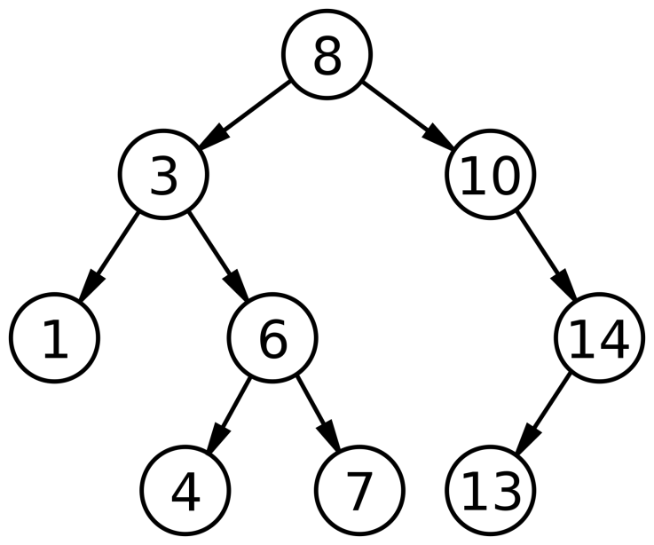
**Дерево** — это [связный](https://ru.wikipedia.org/wiki/%D0%A1%D0%B2%D1%8F%D0%B7%D0%BD%D1%8B%D0%B9_%D0%B3%D1%80%D0%B0%D1%84) [ациклический граф](https://ru.wikipedia.org/wiki/%D0%90%D1%86%D0%B8%D0%BA%D0%BB%D0%B8%D1%87%D0%B5%D1%81%D0%BA%D0%B8%D0%B9_%D0%B3%D1%80%D0%B0%D1%84).[[1]](https://ru.wikipedia.org/wiki/%D0%94%D0%B5%D1%80%D0%B5%D0%B2%D0%BE_(%D1%82%D0%B5%D0%BE%D1%80%D0%B8%D1%8F_%D0%B3%D1%80%D0%B0%D1%84%D0%BE%D0%B2)#cite_note-1) Связность означает наличие маршрута между любой парой вершин, ацикличность — отсутствие циклов. Отсюда, в частности, следует, что число рёбер в дереве на единицу меньше числа вершин, а между любыми парами вершин имеется один и только один путь.   
**Лес** —множество-деревьев.

**Высота узла n** равна длине самого длинного пути от узла n вниз до внешнего узла поддерева n. Высота двоичного дерева определяется как высота его корневого узла.  
Глубина узла равна длине уникального пути от корня к узлу.  
**Ориентированное (направленное) дерево** — ацикличный орграф ([ориентированный граф](https://ru.wikipedia.org/wiki/%D0%9E%D1%80%D0%B8%D0%B5%D0%BD%D1%82%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%BD%D1%8B%D0%B9_%D0%B3%D1%80%D0%B0%D1%84), не содержащий циклов), в котором только одна вершина имеет нулевую степень захода (в неё не ведут дуги), а все остальные вершины имеют степень захода 1 (в них ведёт ровно по одной дуге). Вершина с нулевой степенью захода называется *корнем* дерева, вершины с нулевой степенью исхода (из которых не исходит ни одна дуга) называются *концевыми вершинами* или *листьями*  
**Остовное дерево**[**графа**](https://ru.wikipedia.org/wiki/%D0%93%D1%80%D0%B0%D1%84_(%D0%BC%D0%B0%D1%82%D0%B5%D0%BC%D0%B0%D1%82%D0%B8%D0%BA%D0%B0)) — это [дерево](https://ru.wikipedia.org/wiki/%D0%94%D0%B5%D1%80%D0%B5%D0%B2%D0%BE_(%D1%82%D0%B5%D0%BE%D1%80%D0%B8%D1%8F_%D0%B3%D1%80%D0%B0%D1%84%D0%BE%D0%B2)), подграф данного графа, с тем же числом вершин, что и у исходного графа. Неформально говоря, остовное дерево получается из исходного графа удалением максимального числа рёбер, входящих в циклы, но без нарушения связности графа. Остовное дерево включает в себя все n вершин исходного графа и содержит n − 1 ребро.  
**Минимальное остовное дерево** (или **минимальное покрывающее дерево**) в связанном взвешенном [неориентированном графе](https://ru.wikipedia.org/wiki/%D0%9D%D0%B5%D0%BE%D1%80%D0%B8%D0%B5%D0%BD%D1%82%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%BD%D1%8B%D0%B9_%D0%B3%D1%80%D0%B0%D1%84) — это [остовное дерево](https://ru.wikipedia.org/wiki/%D0%9E%D1%81%D1%82%D0%BE%D0%B2%D0%BD%D0%BE%D0%B5_%D0%B4%D0%B5%D1%80%D0%B5%D0%B2%D0%BE) этого графа, имеющее минимальный возможный вес, где под весом дерева понимается сумма весов входящих в него рёбер.

**24. Бинарное дерево поиска. Основные операции с бинарным деревом поиска: вставка элемента, поиск и удаление.**

**Бинарное дерево** — это конечное множество узлов, которое либо пусто, либо состоит из корня и двух непересекающихся бинарных деревьев — левого и правого.

Из определения следует, что в бинарном дереве каждый узел может иметь максимум две связи, причем связь влево (если она имеется) ведет к младшему сыну, а связь вправо (если она имеется) ведет к ближайшему по старшинству брату.

**Бинарное дерево поиска** — это бинарное дерево, обладающее дополнительными свойствами: значение левого потомка меньше значения родителя, а значение правого потомка больше значения родителя для каждого узла дерева. То есть, данные в бинарном дереве поиска хранятся в отсортированном виде. При каждой операции вставки нового или удаления существующего узла отсортированный порядок дерева сохраняется. При поиске элемента сравнивается искомое значение с корнем. Если искомое больше корня, то поиск продолжается в правом потомке корня, если меньше, то в левом, если равно, то значение найдено и поиск прекращается.  
  


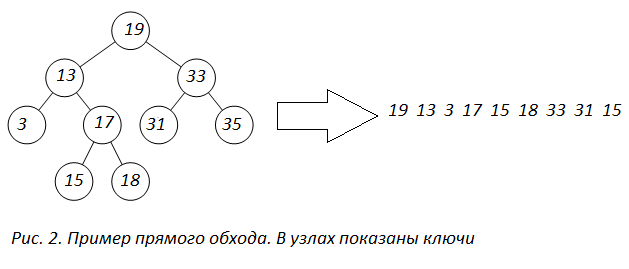
**Процедура поиска** узла по ключу заключается в том, что на каждом шаге значение искомого ключа сравнивается со значением ключа рассматриваемого узла, начиная с корня. Если значение искомого ключа меньше,чем значение ключа рассматриваемого узла, то поиск продолжается в левом поддереве, если больше —то в правом поддереве. И так, пока не будет найден узел с искомым ключом(см. Рис.1(а))или пока поиск не достигнет того узла, ниже которого этот узел не может находиться. Если при поиске мы обнаруживаем, что узел далее надо искать, например, в правом поддереве, а оно пусто(как на Рис. 1(б)),следовательно, мы можем сделать вывод, что искомого ключа в дереве нет.

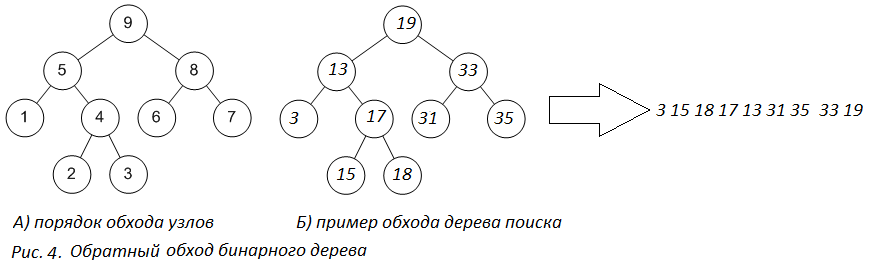
**Вставка узла** в двоичное дерево поиска выполняется после того, как найдено место, куда можно вставить новый узел так, чтобы сохранились свойства дерева поиска. Для этого выполняется поиск узла с этим ключом в дереве. Если узел с таким ключом в дереве уже имеется, то вставку выполнять не нужно. В противном случае поиск остановится на некотором узле, к которому впоследствии будет подсоединяться узел слева или справа в зависимости от значения его ключа.

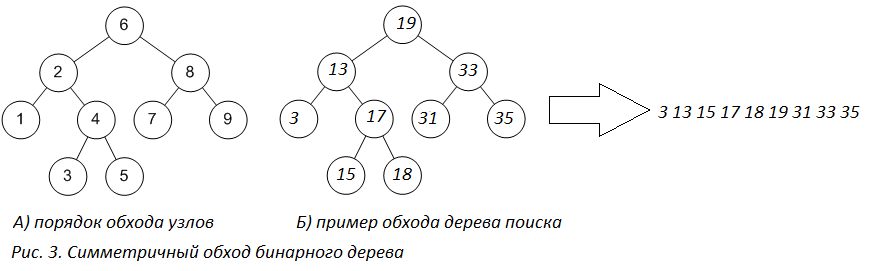
**Удаление:**  
У узла нет сыновей – узел является листовым.В этом случае узел удаляется, и соответствующее поддерево его родителя становится пустым (Рис.2(а)).122.  
У узла только один сын. Узел удаляется, и его сын переходит к его родителю.  
У узла два сына. Пусть В – удаляемый узел. Ищем его последователя С –узла с минимальным ключом в правом поддереве удаляемого узла. Переносим ключ узла C в узел В и сводим задачу к удалению узла С. Эта процедура является корректной, потому что после удаления узла B его место должен занять как раз его последователь. Время поиска последователя, как было показано выше, ограничено высотой дерева.

**25. Обходы деревьев: прямой, симметричный, обратный; обход в ширину**.

**Прямой:** заключается в том что сначала просматриваются корни (родительские узлы)а-затем-их-дочерние-листы.

  
**Обратный**: это случай когда нам нужно начать-так сказать с листов и завершить главным-узлом.

  
**Симметричный**: используется как раз когда нам надо проверять в начале детей и только-потом-подыматься-к-родительским-узлам.

  
**В ширину**: обход в ширину подразумевает, что сначала мы посещаем корень, затем, слева направо, все ветви первого уровня, затем все ветви второго уровня и т.д.   
префиксный (прямой) обход — сначала обрабатывается текущий узел, затем левое и правое поддеревья; инфиксный (симметричный) обход — сначала обрабатывается левое поддерево текущего узла, затем корень, затем правое поддерево;  
постфиксный (обратный) обход — сначала обрабатываются левое и правое поддеревья текущего узла, затем сам узел.

**26. Понятие бинарной кучи, основные операции для бинарной кучи. Реализация бинарной кучи в программе.**

Двоичная куча представляет собой полное бинарное дерево, для которого выполняется основное свойство кучи: приоритет каждой вершины больше приоритетов её потомков. В простейшем случае приоритет каждой вершины можно считать равным её значению. В таком случае структура называется max-heap, поскольку корень поддерева является максимумом из значений элементов поддерева.

Существуют также кучи, где значение в любой вершине, наоборот, не больше, чем значения её потомков. Такие кучи называются min-heap, а кучи, описанные выше — max-heap.

**Процедура Heap\_Increase\_Key** заменяет элемент кучи на новый ключ со значением, не меньшим значения исходного элемента.  
Если элемент меньше своего отца, условие 1 соблюдено для всего дерева, и больше ничего делать не нужно. Если он больше, мы меняем местами его с отцом. Если после этого отец больше деда, мы меняем местами отца с дедом и так далее. Иными словами, слишком большой элемент всплывает наверх.  
**Новый элемент добавляется** на последнее место в массиве. Возможно, что при этом будет нарушено основное свойство кучи, так как новый элемент может быть больше родителя. В таком случае следует «поднимать» новый элемент на один уровень (менять с вершиной-родителем) до тех пор, пока не будет соблюдено основное свойство кучи.

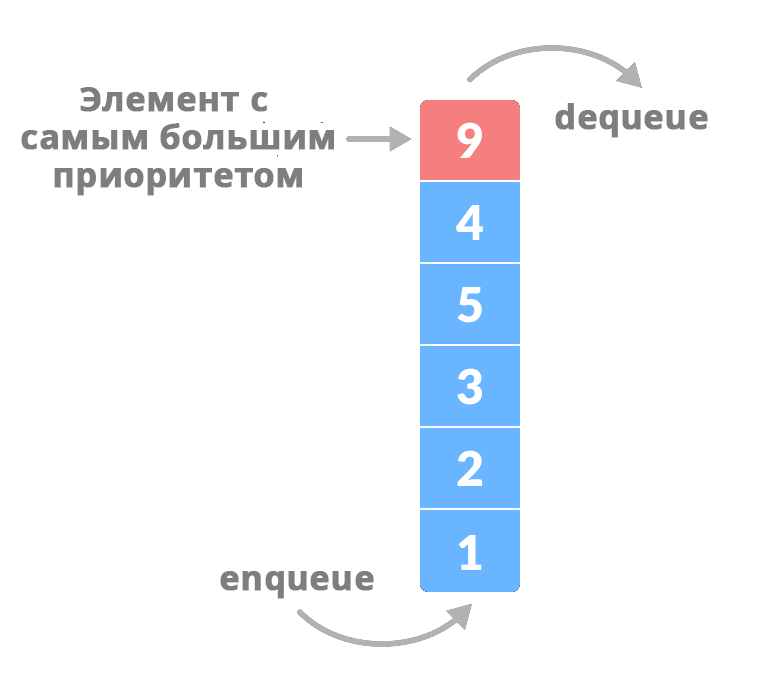
**В упорядоченном max-heap максимальный элемент** всегда хранится в корне. Восстановить упорядоченность двоичной кучи после удаления максимального элемента можно, поставив на его место последний элемент и упорядочив все дерево.  
**Для упорядочивания необходимо** «опускать» i-ую вершину (менять местами с наибольшим из потомков), пока основное свойство не будет восстановлено.

**27. Очередь с приоритетом, основные операции, виды реализации.**

Очереди с приоритетом — разновидность очередей, в которой у каждого элемента есть свой приоритет. Обслуживаются они в соответствии со своими приоритетом. Если у элементов одинаковый приоритет, то обслуживаются они по их порядку в очереди.

Значение элемента, как правило, и определяет его приоритет.

То есть, у элемента с самым большим значением самый высокий приоритет. Правда, это не всегда так. Самый высокий приоритет может быть у элемента и с самым малым значением. В остальных случаях мы можем задавать приоритеты элементам по своему усмотрению.



Обычная очередь подчиняется принципу FIFO «первый вошел — первый вышел». В очередях с приоритетом элементы удаляются в соответствии с их приоритетом. То есть, элемент с самым высоким приоритетом удаляется из очереди в первую очередь.

Очереди с приоритетом можно реализовать с помощью следующих структур данных: массив, связный список, куча и двоичное дерево поиска. Среди всех этих структур выделяется куча — это самый эффективный способ реализации очереди с приоритетом. Именно поэтому в этом руководстве мы будем использовать кучи. Конкретно — max-кучи

Основные операции очередей с приоритетом: вставка, удаление и просмотр элемента.

Вставка элементов в очередь с приоритетом (max-куча) происходит следующим образом:

* Вставьте новое значение в конец дерева.
* Отсортируйте дерево.

Алгоритм вставки элемента в очередь с приоритетом в общем виде выглядит так:

if нет узла,   
  создайте новый узел с помощью newNode.  
else узел уже есть  
  вставьте новый узел в конец дерева (последний узел слева — направо)  
отсортируйте массив

Удаление элемента из очереди с приоритетом происходит следующим образом:

* Выберите элемент, который хотите удалить.
* Поменяйте последний и удаляемый элемент местами.
* Удалите последний элемент.
* Отсортируйте дерево.

Алгоритм удаления элемента из очереди с приоритетом в общем виде выглядит так:

Если узел удаляемыйУзел = конечныйУзел  
  Удалить узел  
Иначе поменять местами удаляемыйУзел и последнийУзел  
  Удалить удаляемыйУзел  
  
отсортировать дерево

**28. Ассоциативный массив, основные операции, виды реализации**.

**Ассоциативный массив** — [абстрактный тип данных](https://ru.wikipedia.org/wiki/%D0%90%D0%B1%D1%81%D1%82%D1%80%D0%B0%D0%BA%D1%82%D0%BD%D1%8B%D0%B9_%D1%82%D0%B8%D0%BF_%D0%B4%D0%B0%D0%BD%D0%BD%D1%8B%D1%85) ([интерфейс](https://ru.wikipedia.org/wiki/%D0%9F%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D0%BD%D1%8B%D0%B9_%D0%B8%D0%BD%D1%82%D0%B5%D1%80%D1%84%D0%B5%D0%B9%D1%81) к хранилищу данных), позволяющий хранить пары вида «(ключ, значение)» и поддерживающий операции добавления пары, а также поиска и удаления пары по ключу:

* INSERT(ключ, значение)
* FIND(ключ)
* REMOVE(ключ)

Предполагается, что ассоциативный массив не может хранить две пары с одинаковыми ключами.

В паре {\displaystyle (k,v)}k,v значение {\displaystyle v}v называется значением, ассоциированным с ключом {\displaystyle k}k. Где {\displaystyle k}k — это *key*, a {\displaystyle v}v — *value*. Семантика и названия вышеупомянутых операций в разных реализациях ассоциативного массива могут отличаться.

Операция FIND(ключ) возвращает значение, ассоциированное с заданным ключом, или некоторый специальный объект UNDEF, означающий, что значения, ассоциированного с заданным ключом, нет. Две другие операции ничего не возвращают (за исключением, возможно, информации о том, успешно ли была выполнена данная операция).

Ассоциативный массив с точки зрения интерфейса удобно рассматривать как обычный [массив](https://ru.wikipedia.org/wiki/%D0%9C%D0%B0%D1%81%D1%81%D0%B8%D0%B2_(%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D0%B5)), в котором в качестве индексов можно использовать не только целые числа, но и значения других типов — например, строки.

Примером ассоциативного массива является телефонный справочник: значением в данном случае является совокупность «Ф. И. О. + адрес», а ключом — номер телефона, один номер телефона имеет одного владельца, но один человек может иметь несколько номеров.

Три основных операции часто дополняются другими, наиболее популярные расширения:

* CLEAR — удалить все записи,
* EACH — «пробежаться» по всем хранимым парам,
* MIN — найти пару с минимальным значением ключа,
* MAX — найти пару с максимальным значением ключа.

В последних двух случаях необходимо, чтобы на ключах была определена операция сравнения.

Существует **множество различных реализаций** ассоциативного массива.

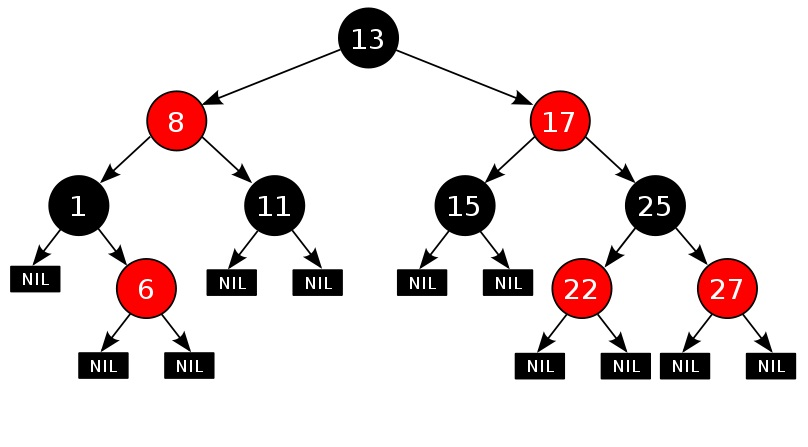
Самая простая реализация может быть основана на обычном массиве, элементами которого являются пары (ключ, значение). Наиболее популярны реализации, основанные на различных [деревьях поиска](https://ru.wikipedia.org/wiki/%D0%94%D0%B2%D0%BE%D0%B8%D1%87%D0%BD%D0%BE%D0%B5_%D0%B4%D0%B5%D1%80%D0%B5%D0%B2%D0%BE_(%D1%81%D1%82%D1%80%D1%83%D0%BA%D1%82%D1%83%D1%80%D0%B0_%D0%B4%D0%B0%D0%BD%D0%BD%D1%8B%D1%85)).  В языках [Java](https://ru.wikipedia.org/wiki/Java), [Ruby](https://ru.wikipedia.org/wiki/Ruby), [Tcl](https://ru.wikipedia.org/wiki/Tcl" \o "Tcl), [Python](https://ru.wikipedia.org/wiki/Python) используется один из вариантов [хеш-таблицы](https://ru.wikipedia.org/wiki/%D0%A5%D0%B5%D1%88-%D1%82%D0%B0%D0%B1%D0%BB%D0%B8%D1%86%D0%B0). Есть и другие реализации.

**29. Красно-черные деревья, основные операции.**

**Красно-чёрное дерево** (англ. *red-black tree*) — двоичное дерево поиска, в котором баланс осуществляется на основе "цвета" узла дерева, который принимает только два значения: "красный" (англ. *red*) и "чёрный" (англ. *black*).

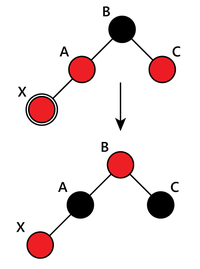
Красно-чёрным называется бинарное поисковое дерево, у которого каждому узлу сопоставлен дополнительный атрибут — цвет и для которого выполняются следующие свойства:

1. Каждый узел промаркирован красным или чёрным цветом
2. Корень и конечные узлы (листья) дерева — чёрные
3. У красного узла родительский узел — чёрный
4. Все простые пути из любого узла x до листьев содержат одинаковое количество чёрных узлов
5. Чёрный узел может иметь чёрного родителя

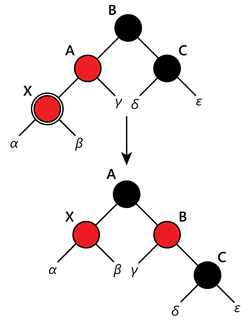


Каждый элемент вставляется вместо листа, поэтому для выбора места вставки идём от корня до тех пор, пока указатель на следующего сына не станет nilnil (то есть этот сын — лист). Вставляем вместо него новый элемент с нулевыми потомками и красным цветом. Теперь проверяем балансировку. Если отец нового элемента черный, то никакое из свойств дерева не нарушено. Если же он красный, то нарушается свойство 33, для исправления достаточно рассмотреть два случая:

1. "Дядя" этого узла тоже красный. Тогда, чтобы сохранить свойства 33 и 44, просто перекрашиваем "отца" и "дядю" в чёрный цвет, а "деда" — в красный. В таком случае черная высота в этом поддереве одинакова для всех листьев и у всех красных вершин "отцы" черные. Проверяем, не нарушена ли балансировка. Если в результате этих перекрашиваний мы дойдём до корня, то в нём в любом случае ставим чёрный цвет, чтобы дерево удовлетворяло свойству 2.

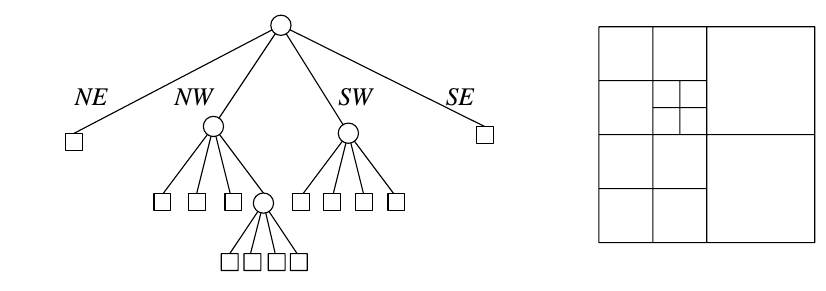
[](https://neerc.ifmo.ru/wiki/index.php?title=%D0%A4%D0%B0%D0%B9%D0%BB:Untitled-1.png)

1. "Дядя" чёрный. Если выполнить только перекрашивание, то может нарушиться постоянство чёрной высоты дерева по всем ветвям. Поэтому выполняем поворот. Если добавляемый узел был правым потомком, то необходимо сначала выполнить левое вращение, которое сделает его левым потомком. Таким образом, свойство 33 и постоянство черной высоты сохраняются.

[](https://neerc.ifmo.ru/wiki/index.php?title=%D0%A4%D0%B0%D0%B9%D0%BB:Untitled-2.png)

**30. Квадродеревья, основные операции.**

**Квадродерево** — дерево, каждая внутренняя вершина которого содержит 4 ребёнка. Любому узлу квадродерева соответствует некоторый квадрат. Если внутренней вершине v соответствует какой-то квадрат a, то её детям соответствуют четверти квадрата a (см. картинку).

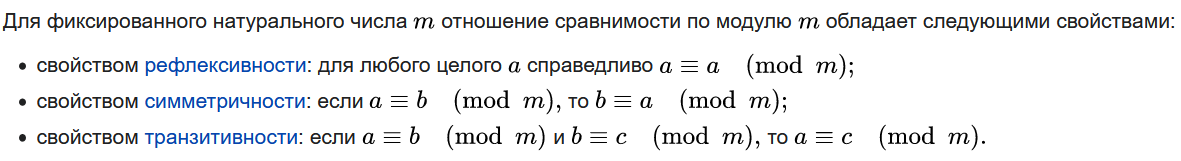


Четырехъядерное дерево - это древовидная структура, основанная на степени четыре и используемая для организации файлов в базе данных. Каждый родительский или начальный узел имеет четыре дочерних узла, и каждый дочерний узел содержит определенный объем данных. Когда лимит данных выходит за его границы, четыре узла будут сделаны из этого узла. Существует две основные структуры дерева квадрантов: область и дерево точек, каждая из которых немного отличается по дизайну. Хотя квадродерево чаще всего используется с базами данных, его также можно использовать для поиска пикселей в двумерных (2D) изображениях, поскольку пиксели в 2D-изображении всегда можно разделить на четыре части.

Все древовидные структуры создаются с родительскими или ветвистыми узлами и дочерними или конечными узлами. Родитель является отправной точкой и содержит широкие данные, основанные на категориях, в то время как ребенок хранит файлы и документы. В квадродереве каждый родитель должен иметь четверых детей. Хотя должно быть четверо детей, не все дети должны содержать данные; те, которые без, известны как нулевые узлы. Эти нулевые узлы часто остаются неизменными и ждут данных.

У каждого дочернего узла в дереве квадрантов есть предел данных. Этот предел обычно определяется общим размером базы данных. Когда информации так много, что она выходит за пределы, дочерний узел становится родительским узлом, по сути, рожая - создавая четыре дочерних узла, которые занимают все дополнительные данные. Обычно это будет один или два нулевых узла от этого создания, но это полностью зависит от того, сколько данных было в узле.

**31. Понятие сравнения. Основные свойства сравнений.**

**Сравне́ние** — Сравне́ние двух [целых чисел](https://ru.wikipedia.org/wiki/%D0%A6%D0%B5%D0%BB%D0%BE%D0%B5_%D1%87%D0%B8%D1%81%D0%BB%D0%BE) по мо́дулю [натурального числа](https://ru.wikipedia.org/wiki/%D0%9D%D0%B0%D1%82%D1%83%D1%80%D0%B0%D0%BB%D1%8C%D0%BD%D0%BE%D0%B5_%D1%87%D0%B8%D1%81%D0%BB%D0%BE)  m  — математическая операция, позволяющая ответить на вопрос о том, дают ли два выбранных целых числа при делении на m. один и тот же [остаток](https://ru.wikipedia.org/wiki/%D0%94%D0%B5%D0%BB%D0%B5%D0%BD%D0%B8%D0%B5_%D1%81_%D0%BE%D1%81%D1%82%D0%B0%D1%82%D0%BA%D0%BE%D0%BC).  


1. Два числа, сравнимые с третьим сравнимы между собой.
2. Сравнения можно почленно складывать.
3. Сравнения можно почленно перемножать
4. Обе части сравнения можно разделить на их общий делитель, если последний взаимно прост с модулем.
5. Обе части сравнения можно умножить на одно и тоже число
6. Обе части сравнения и модуль можно разделить на их общий делитель.
7. Можно почленно возводить в степень.

**32. Отношение сравнимости на множестве целых чисел *Z*. Классы вычетов.**

Возьмем натуральное целое число m, которое будем называть модулем.   
Определение. Целые числа a и b называются сравнимыми по модулю m, если разность (a–b) делится на m (m|a–b).

Множество всех чисел, сравнимых с a по модулю m , называется классом вычетов a по модулю m , и обычно обозначается [ a ] m или a ¯ m . Таким образом, сравнение a ≡ b равносильно равенству классов вычетов [ a ] m = [ b ] m   
Любое число класса называется вычетом по модулю m. Пусть для определённости  r  ― [остаток от деления](https://ru.wikipedia.org/wiki/%D0%94%D0%B5%D0%BB%D0%B5%D0%BD%D0%B8%D0%B5_%D1%81_%D0%BE%D1%81%D1%82%D0%B0%D1%82%D0%BA%D0%BE%D0%BC) любого из представителей выбранного класса на m, тогда любое число q из этого класса можно представить в виде q = m t + r, где t — [целое](https://ru.wikipedia.org/wiki/%D0%A6%D0%B5%D0%BB%D0%BE%D0%B5_%D1%87%D0%B8%D1%81%D0%BB%D0%BE). Вычет, равный [остатку](https://ru.wikipedia.org/wiki/%D0%94%D0%B5%D0%BB%D0%B5%D0%BD%D0%B8%D0%B5_%D1%81_%D0%BE%D1%81%D1%82%D0%B0%D1%82%D0%BA%D0%BE%D0%BC)  r ( q = r при t = 0) называется наименьшим неотрицательным вычетом, а вычет ρ ( q = ρ ), самый малый по абсолютной величине, называется абсолютно наименьшим вычетом.

Поскольку сравнимость по модулю m является [отношением эквивалентности](https://ru.wikipedia.org/wiki/%D0%9E%D1%82%D0%BD%D0%BE%D1%88%D0%B5%D0%BD%D0%B8%D0%B5_%D1%8D%D0%BA%D0%B2%D0%B8%D0%B2%D0%B0%D0%BB%D0%B5%D0%BD%D1%82%D0%BD%D0%BE%D1%81%D1%82%D0%B8) на ножестве целых чисел Z, то классы вычетов по модулю m представляют собой [классы эквивалентности](https://ru.wikipedia.org/wiki/%D0%9E%D1%82%D0%BD%D0%BE%D1%88%D0%B5%D0%BD%D0%B8%D0%B5_%D1%8D%D0%BA%D0%B2%D0%B8%D0%B2%D0%B0%D0%BB%D0%B5%D0%BD%D1%82%D0%BD%D0%BE%D1%81%D1%82%D0%B8); их количество равно m.

**33. Эйлеровы и гамильтоновы циклы в графе, критерии существования циклов.**

**Эйлеров путь (эйлерова цепь) в графе** — это [путь](https://ru.wikipedia.org/wiki/%D0%9F%D1%83%D1%82%D1%8C_%D0%B2_%D0%B3%D1%80%D0%B0%D1%84%D0%B5), проходящий по всем рёбрам [графа](https://ru.wikipedia.org/wiki/%D0%93%D1%80%D0%B0%D1%84_(%D0%BC%D0%B0%D1%82%D0%B5%D0%BC%D0%B0%D1%82%D0%B8%D0%BA%D0%B0)) и притом только по одному разу.

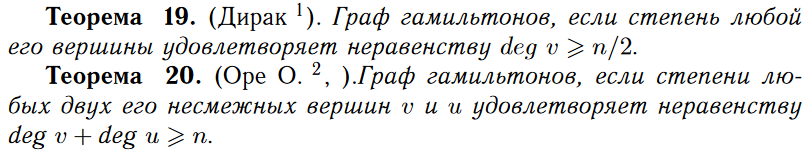
**Связный граф G является эйлеровым** тогда и только тогда, когда каждая вершина в G имеет четная степень.

**Следствие 2.1.** Связный граф является эйлеровым тогда и только тогда, когда семейство его ребер можно разбить на непересекающиеся циклы.

**Следствие 2.2.** Связный граф является полуэйлеровым тогда и только тогда, когда в нем не более двух вершин имеют нечетные степени.

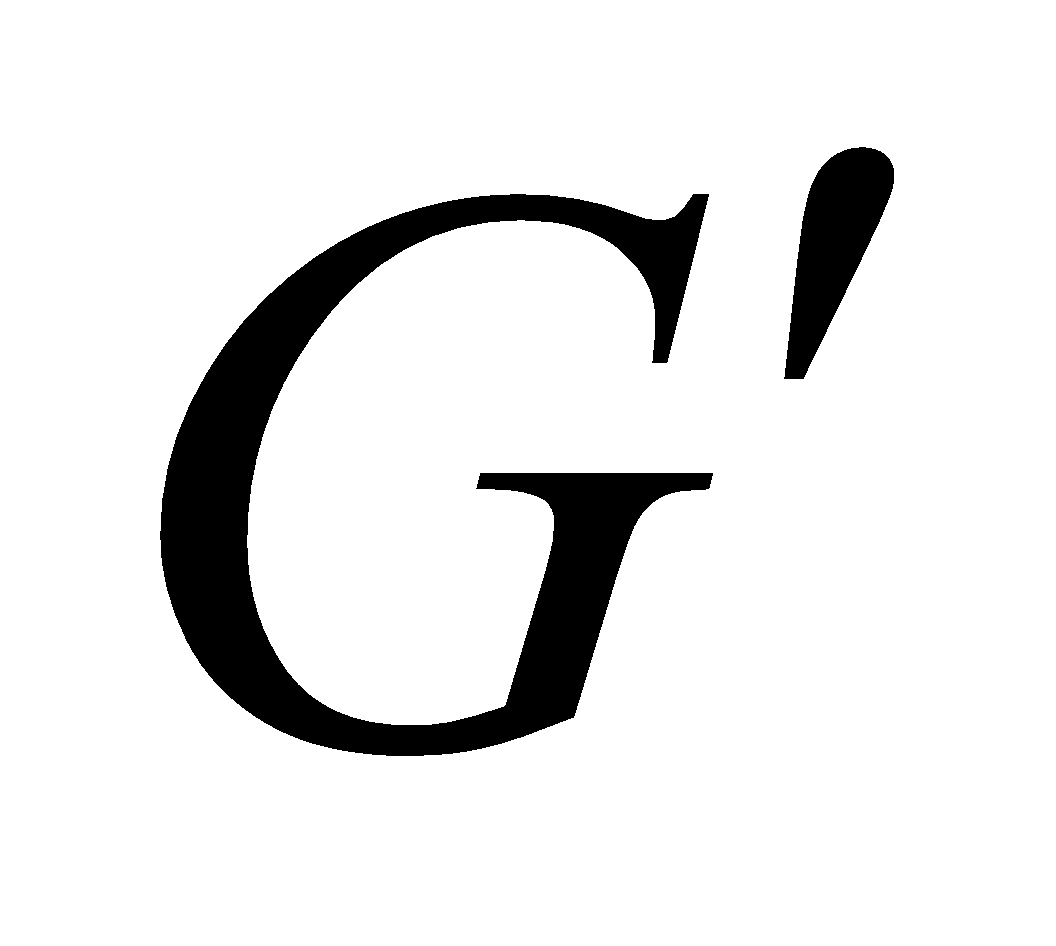
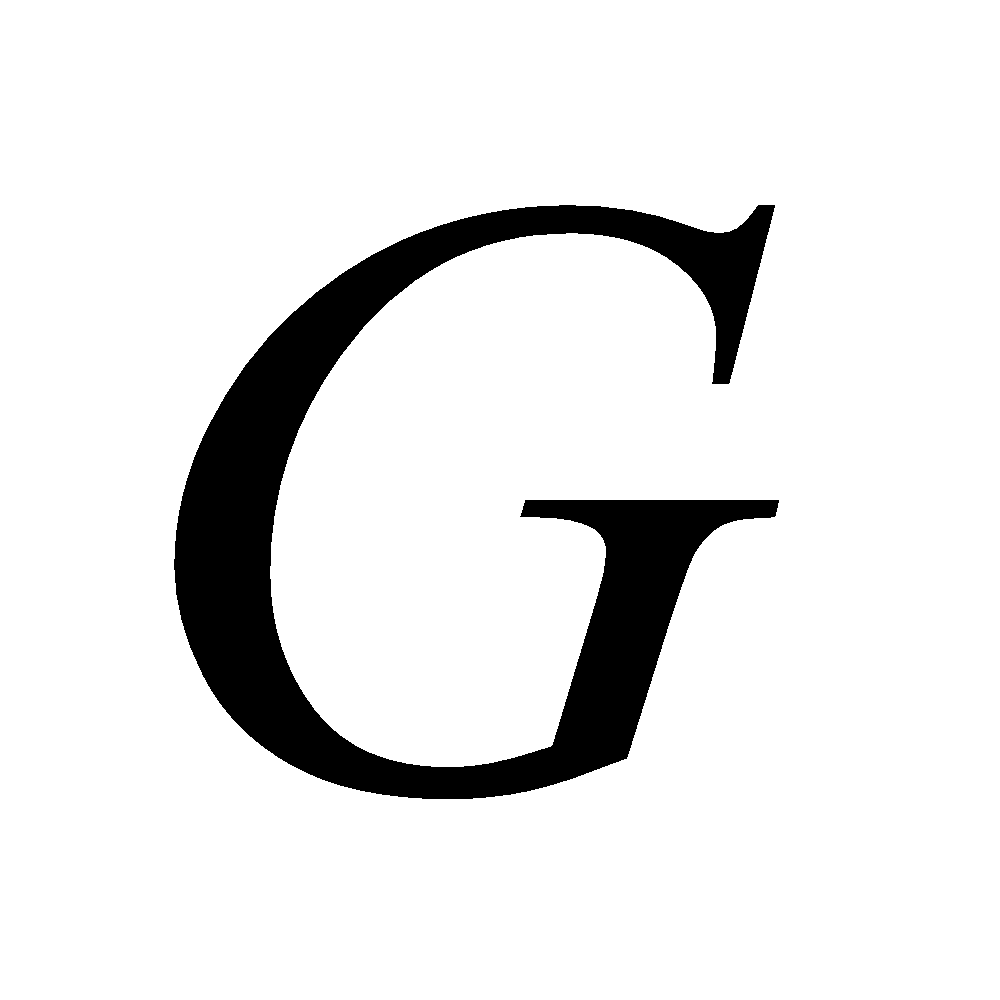
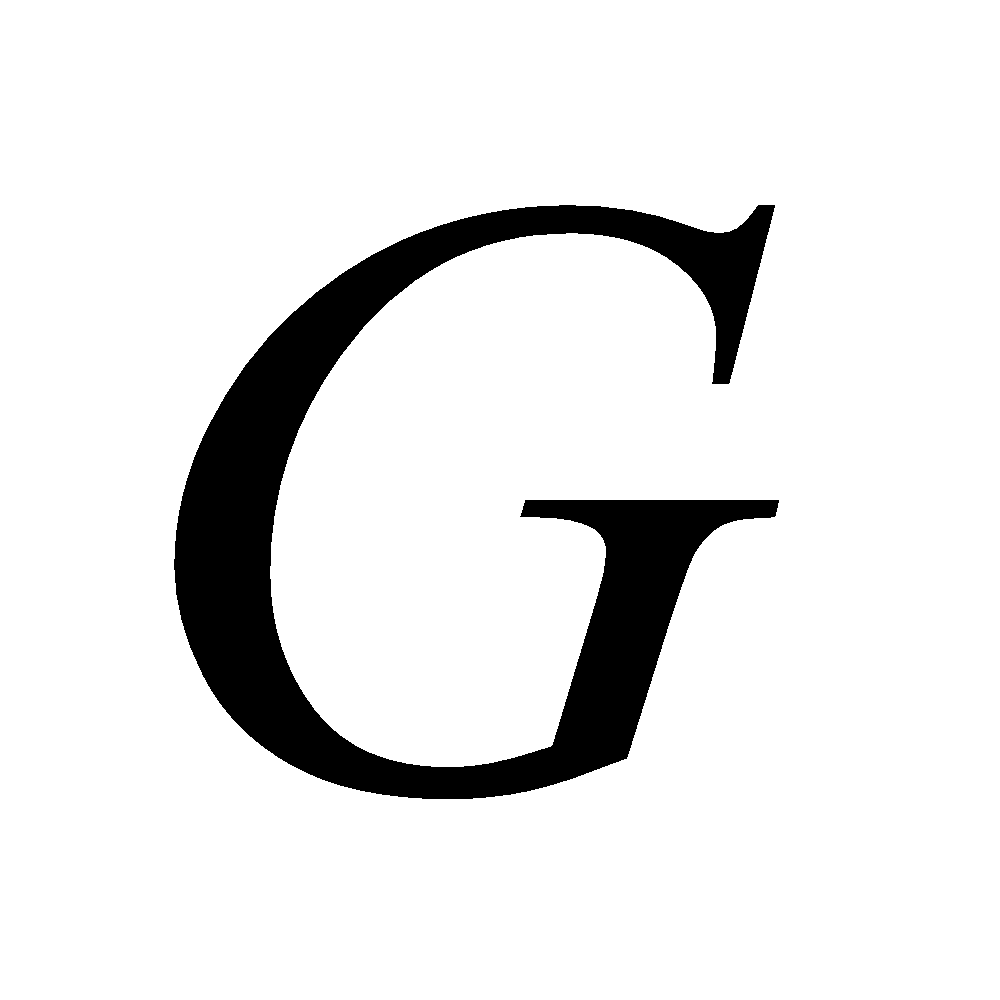
**Гамильтоновым циклом** является такой цикл (замкнутый путь), который проходит через каждую вершину данного графа ровно по одному разу[[2]](https://ru.wikipedia.org/wiki/%D0%93%D0%B0%D0%BC%D0%B8%D0%BB%D1%8C%D1%82%D0%BE%D0%BD%D0%BE%D0%B2_%D0%B3%D1%80%D0%B0%D1%84#cite_note-_9e932991424220f5-2); то есть простой цикл, в который входят все вершины графа.

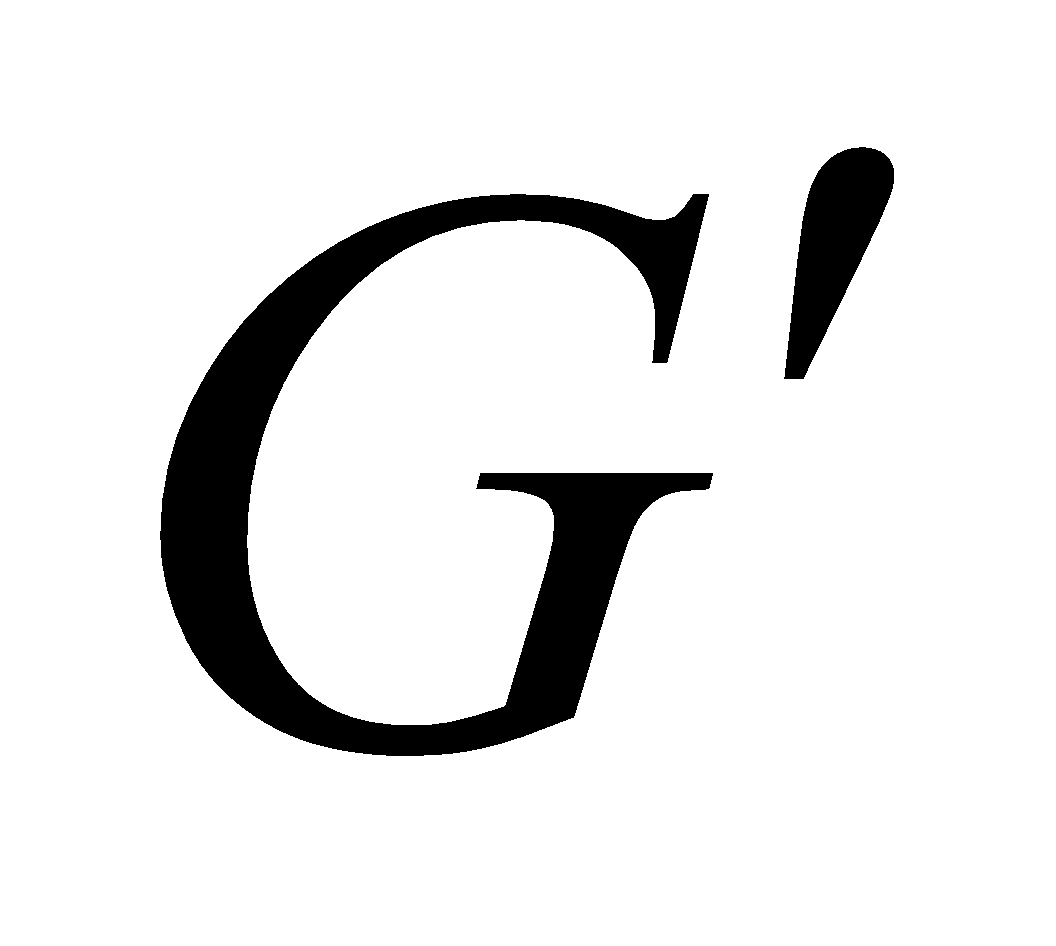
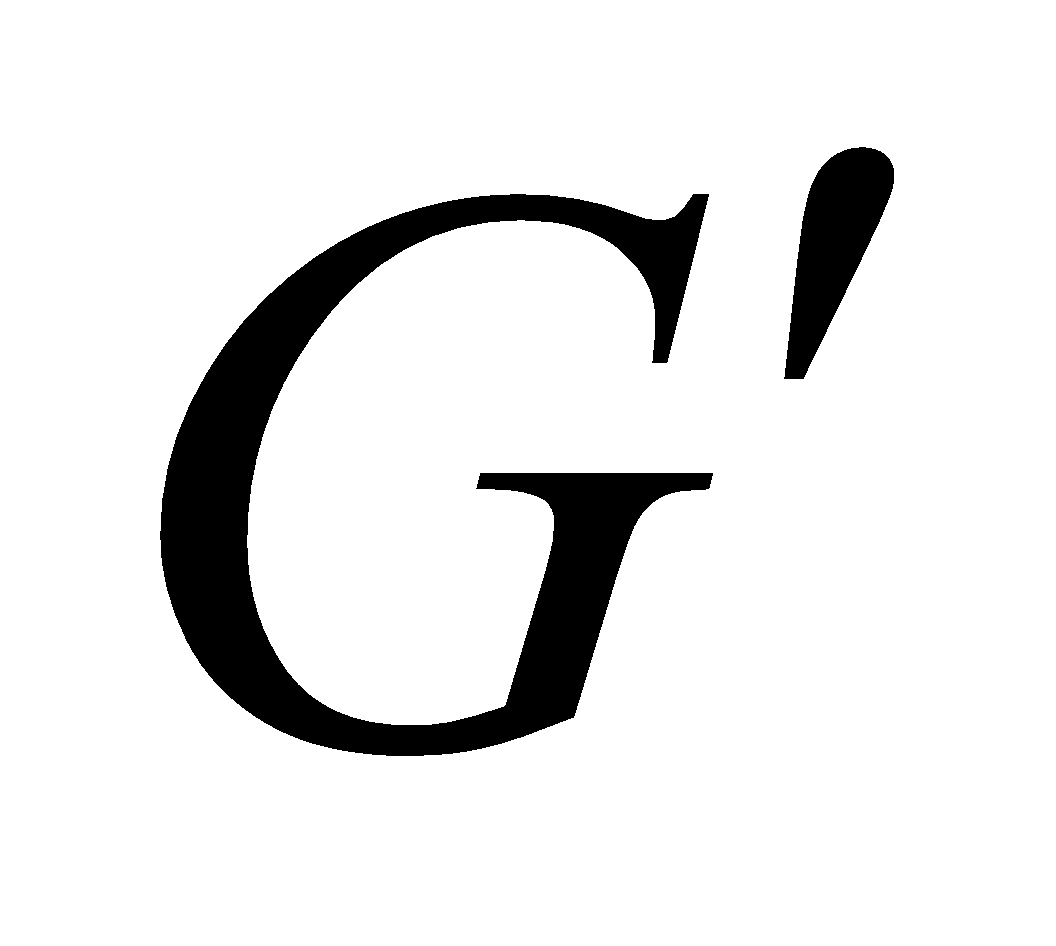
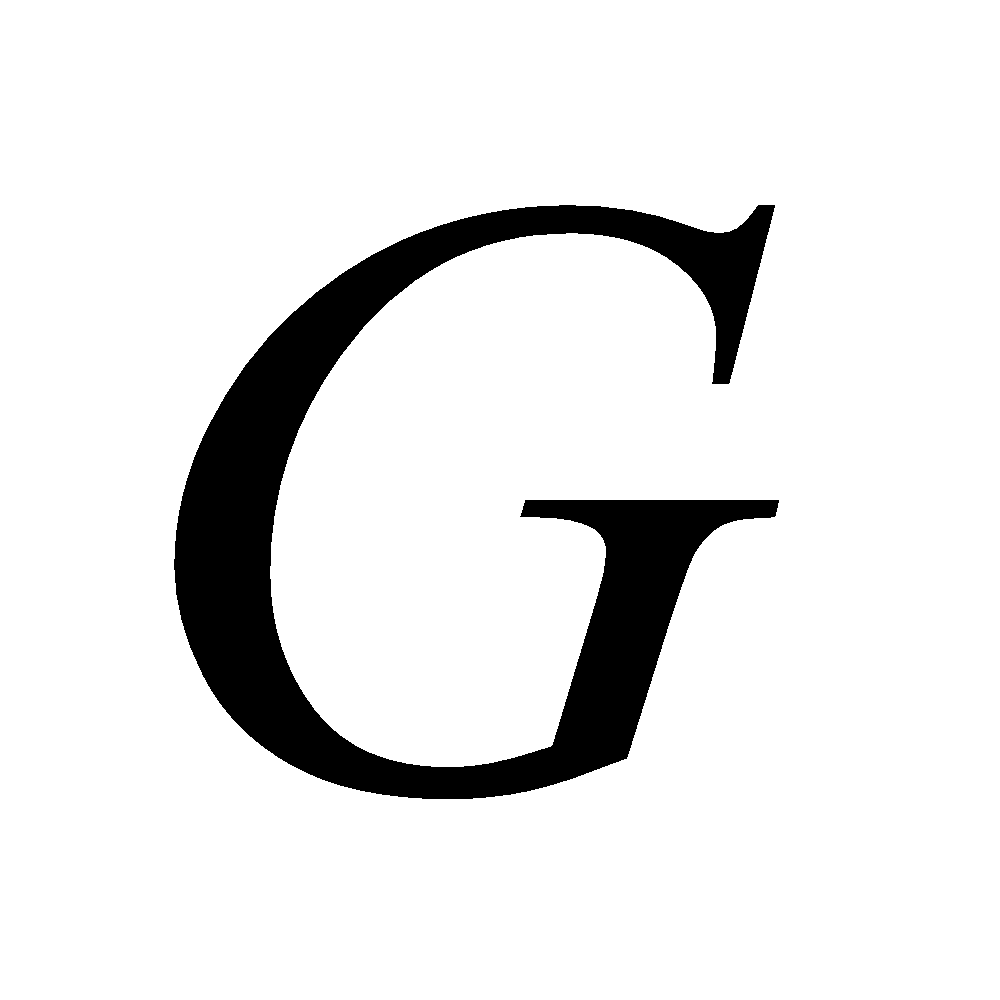
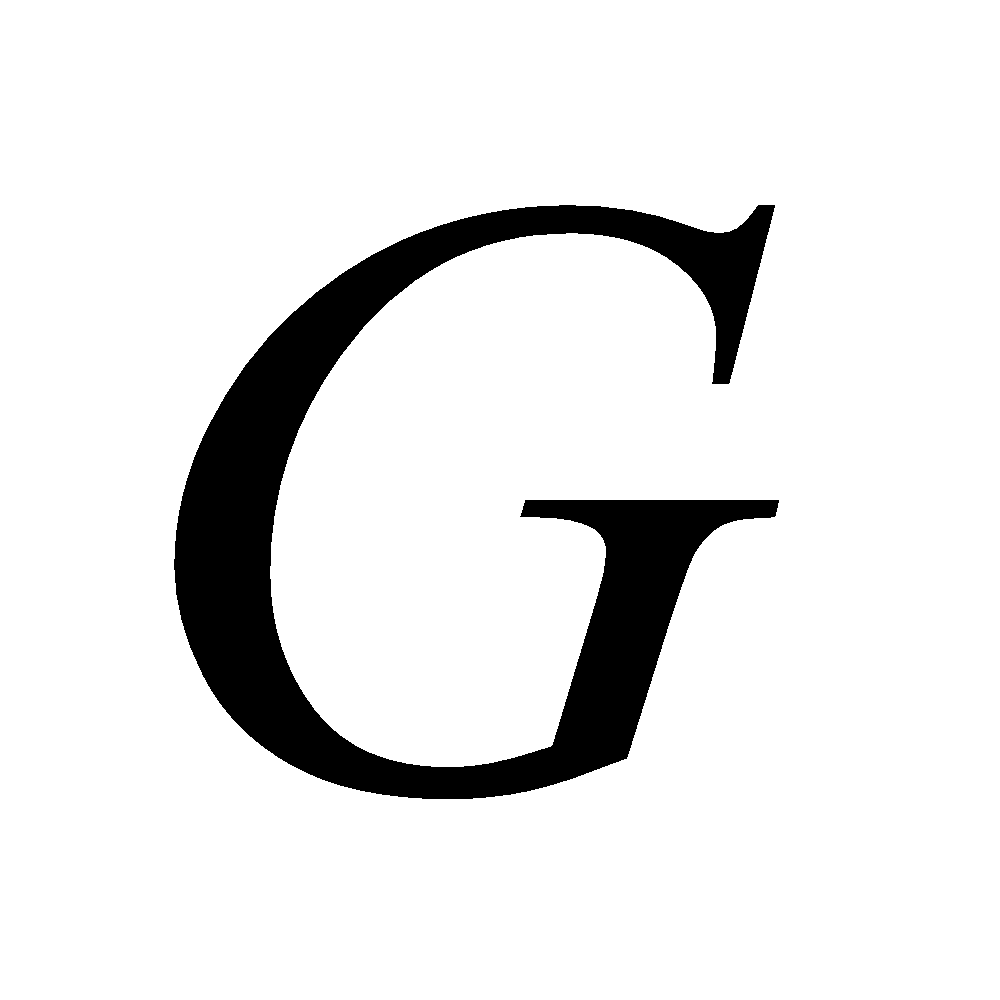
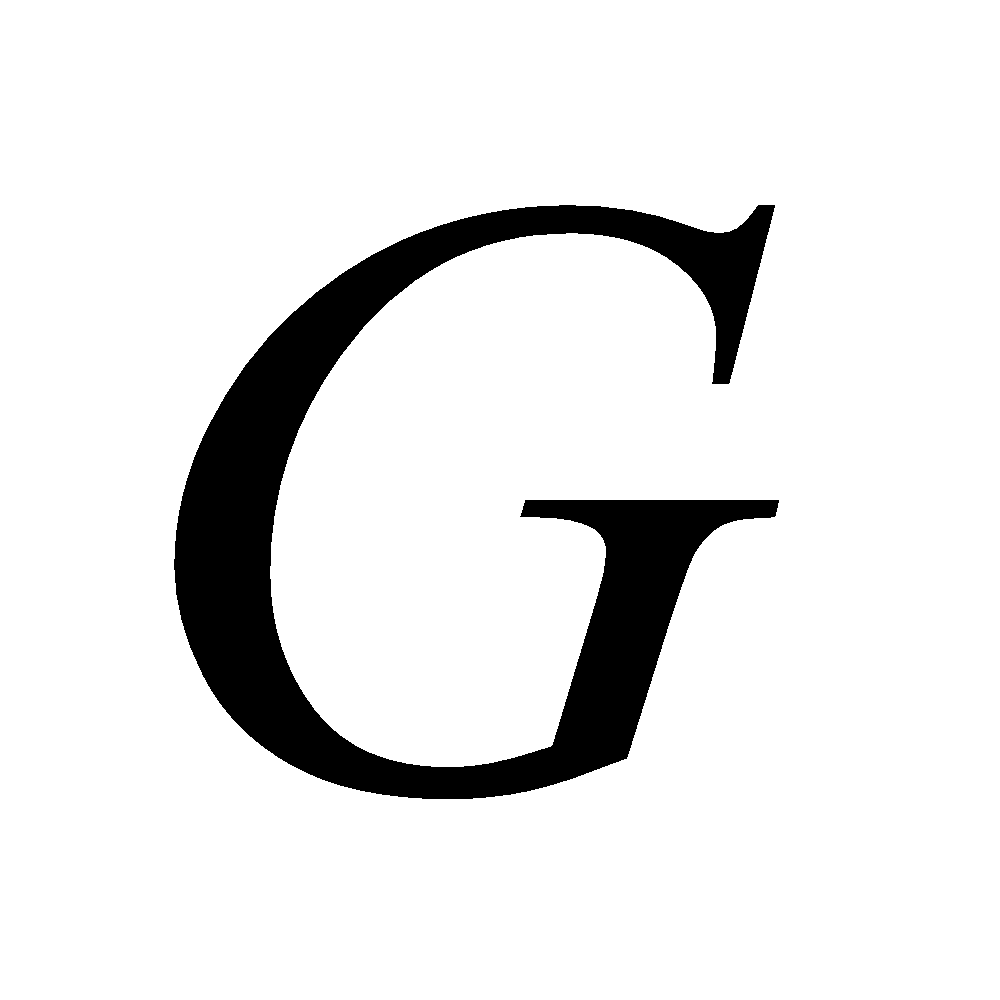
**Критерий существования гамильтонова цикла** в произвольном графе *G* еще не найден. Достаточным условием существования гамильтонова цикла является полнота графа *G.*



**34. Компоненты связности и двусвязности для неориетированных графов (точки сочленения, мосты), сильная связность для орграфа.**

Понятие компоненты связности вытекает из понятия связности графа. Попросту говоря, **компонента связности** - максимально связный подграф графа. Формально, компонента связности - набор вершин графа, между любой парой которых существует путь.

Подграф графа называется *компонентой связности* графа , если выполнены следующие два условия:

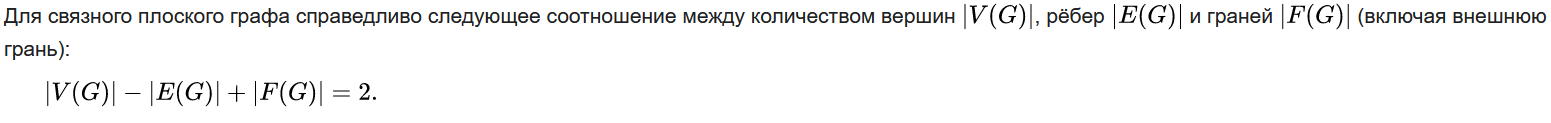
1. - непустой связный граф.
2. - максимальный связный подграф графа (т.е. компонента графа не является собственным подграфом любого другого *связанного* подграфа графа ).

**Компонентой двусвязности** графа называется такое максимальное подмножество из трех или более его вершин, в котором любые две вершины соединены, по крайней мере, двумя путями, не имеющими общих ребер. Кроме того компонента двусвязности может представлять собой просто две вершины, соединенные одним ребром. **Компонента двусвязности** - устойчивая часть графа: если в ней удалить вершину и все примыкающие к ней ребра, то любые две из оставшихся вершин по-прежнему оказываются соединенными между собой.

[Ориентированный граф](https://ru.wikipedia.org/wiki/%D0%9E%D1%80%D0%B8%D0%B5%D0%BD%D1%82%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%BD%D1%8B%D0%B9_%D0%B3%D1%80%D0%B0%D1%84) (орграф) называется ***сильно связным*** ([англ.](https://ru.wikipedia.org/wiki/%D0%90%D0%BD%D0%B3%D0%BB%D0%B8%D0%B9%D1%81%D0%BA%D0%B8%D0%B9_%D1%8F%D0%B7%D1%8B%D0%BA) *strongly connected*), если любые две его вершины s и t сильно связны, то есть если существует ориентированный путь из s в t и ориентированный путь из t в s. **Компонентами сильной связности** орграфа называются его максимальные по включению сильно связные подграфы. **Областью сильной связности** называется множество вершин компоненты сильной связности.

**35. Планарные графы, формула *Эйлера.***

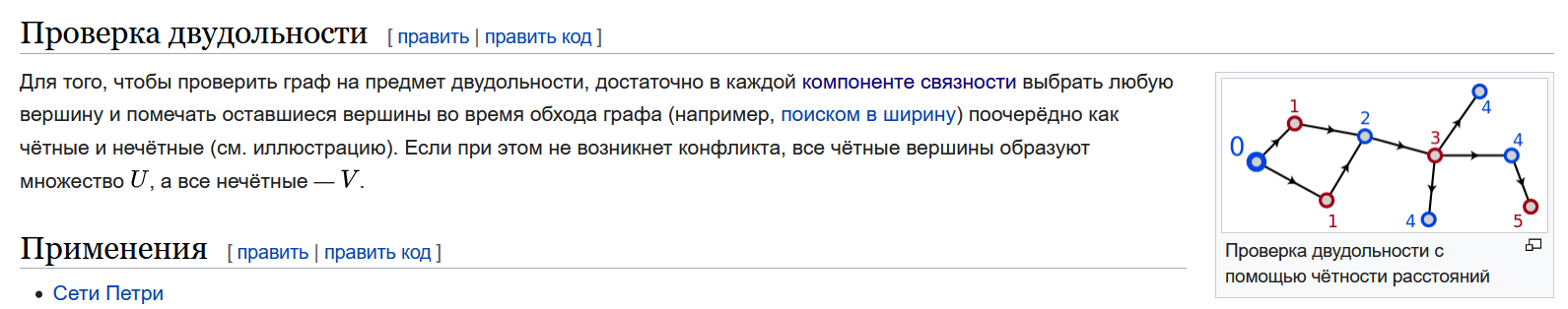
**Плана́рный граф** — [граф](https://ru.wikipedia.org/wiki/%D0%93%D1%80%D0%B0%D1%84_(%D0%BC%D0%B0%D1%82%D0%B5%D0%BC%D0%B0%D1%82%D0%B8%D0%BA%D0%B0)), который можно изобразить на [плоскости](https://ru.wikipedia.org/wiki/%D0%9F%D0%BB%D0%BE%D1%81%D0%BA%D0%BE%D1%81%D1%82%D1%8C_(%D0%B3%D0%B5%D0%BE%D0%BC%D0%B5%D1%82%D1%80%D0%B8%D1%8F)) без пересечений [рёбер](https://ru.wikipedia.org/wiki/%D0%A0%D0%B5%D0%B1%D1%80%D0%BE_(%D0%B3%D0%B5%D0%BE%D0%BC%D0%B5%D1%82%D1%80%D0%B8%D1%8F)) не по вершинам. Какое-либо конкретное изображение планарного графа на плоскости называется плоским графом. Иначе говоря, планарный граф [изоморфен](https://ru.wikipedia.org/wiki/%D0%98%D0%B7%D0%BE%D0%BC%D0%BE%D1%80%D1%84%D0%B8%D0%B7%D0%BC_%D0%B3%D1%80%D0%B0%D1%84%D0%BE%D0%B2) некоторому плоскому графу, изображённому на плоскости так, что его вершины — это [точки](https://ru.wikipedia.org/wiki/%D0%A2%D0%BE%D1%87%D0%BA%D0%B0_(%D0%B3%D0%B5%D0%BE%D0%BC%D0%B5%D1%82%D1%80%D0%B8%D1%8F)) плоскости, а рёбра — кривые на плоскости, которые если и пересекаются между собой, то только по вершинам. Области, на которые граф разбивает плоскость, называются его гранями. Неограниченная часть плоскости — тоже грань, называемая внешней гранью.



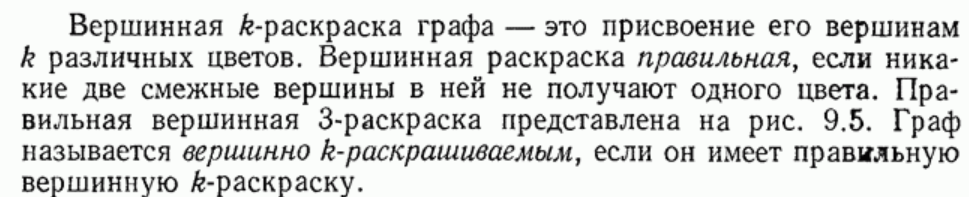
**36. Двудольные графы. Проверка графа на двудольность. Теорема *Кенига*.**

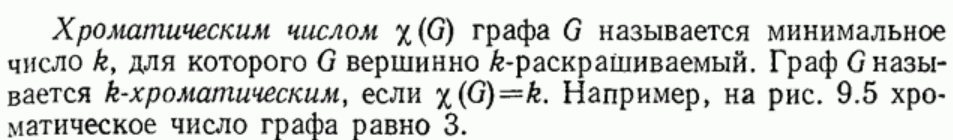
**Двудо́льный граф** или бигра́ф в [теории графов —](https://ru.wikipedia.org/wiki/%D0%A2%D0%B5%D0%BE%D1%80%D0%B8%D1%8F_%D0%B3%D1%80%D0%B0%D1%84%D0%BE%D0%B2) это граф, множество вершин которого можно разбить на две части таким образом, что каждое ребро графа соединяет какую-то вершину из одной части с какой-то вершиной другой части, то есть не существует рёбер между вершинами одной и той же части.  
**Двудольный граф - граф, вершины которого можно разбить на два множества так, что каждое ребро соединяет вершины из разных множеств.**Для проверки графа на двудольность и разбития его на доли чаще всего используется DFS.

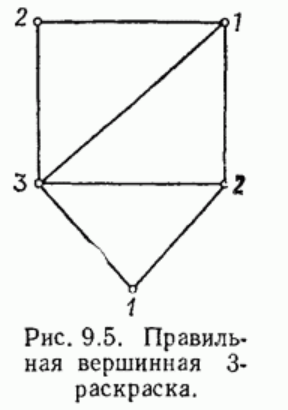
Начинаем покраску с произвольной вершины, которую красим в произвольный цвет. При прохождении по каждому ребру красим следующую вершину в противоположный цвет. Если при переборе соседних вершин мы нашли вершину, уже покрашенную в тот же цвет, что и текущая, то в графе существует нечётный цикл, а значит, он не является двудольным.

  
**Теорема Кёнига**: в любом [двудольном графе](https://ru.wikipedia.org/wiki/%D0%94%D0%B2%D1%83%D0%B4%D0%BE%D0%BB%D1%8C%D0%BD%D1%8B%D0%B9_%D0%B3%D1%80%D0%B0%D1%84) число рёбер в наибольшем паросочетании равно числу вершин в наименьшем вершинном покрытии.  
**Теорема Кёнига 2 (о необходимых и достаточных условиях разбиения множества вершин графа на два независимых подмножества).** Для того, чтобы граф был двудольным, необходимо и достаточно, чтобы он не содержал циклов нечетной длины.

**37. Вершинная раскраска графа. Хромаическое число графа.**







1. Операции над множествами. Диаграммы Эйлера-Венна.
2. Понятие прямоего произведениея множеств, его мощность, понятие булеана, мощность булеана.
3. Область определения, область значений отношений; , , обратное отношение.
4. Операции над отношениями. Операция композиции отношений.
5. Свойства отношений, замыкание отношений.
6. Отношение эквивалентности. Классы эквивалентности. Фактор-множество.
7. Отношения порядка. Отношение частичного порядка.
8. Понятие *ЧУ* – множества. Сравнимые и несравнимые элементы. Диаграммы *Хассе*. Наибольший /наименьший и максимальный/минимальный элементы *ЧУ*-множества.
9. Понятие функции. Свойства функций: инъекция, сюръекция, биекция.
10. Счетно бесконечные множества. Несчетные   бесконечные множества.
11. Понятие операции. Таблица Кэли для задания операции. Свойства операций.
12. Принцип математической индукции.
13. Классы булевых функций. Теорема *Поста-Яблонского*.
14. *СДНФ,  СКНФ* булевой функции, Полином *Жегалкина*.
15. Правило суммы. Правило произведения.Формула включений и исключений в комбинаторике.
16. Понятие выборки, примеры различных выборок.
17. Размещения.
18. Перестановки. Разупорядочения*.*
19. Сочетания.
20. Рекуррентные соотношения. Числа *Фибоначчи*,числа *Каталана*.
21. Решение рекуррентных соотношений. Формула  *Бине.*
22. Рекуррентное соотношение для числа сочетаний. Треугольник *Паскаля*.
23. Понятие дерева, основные понятия. Остовные деревья, минимальный остов.
24. Бинарное дерево поиска. Основные операции с бинарным деревом поиска: вставка элемента, поиск и удаление.
25. Обходы деревьев: прямой, симметричный, обратный; обход в ширину.
26. Понятие бинарной кучи, основные операции для бинарной кучи. Реализация бинарной кучи в программе.
27. Очерндь с приориттетом, основные операции, виды реализации.
28. Ассоциативный массив, основные операции, виды реализации.
29. Красно-черные деревья, основные операции.
30. Квадрадеревья, основные операции.
31. Понятие сравнения. Основные свойства сравнений.
32. Отношение сравнимости на множестве целых чисел *Z*. Классы вычетов.
33. Эйлеровы и гамильтоновы циклы в графе, критерии существования циклов.
34. Компоненты связности и двусвязности для неориетированных графов (точки сочленения, мосты), сильная связность для орграфа.
35. Планарные графы, формула *Эйлера.*
36. Двудольные графы. Проверка графа на двудольность. Теорема *Кенига*.
37. Вершинная раскраска графа. Хромаическое число графа.