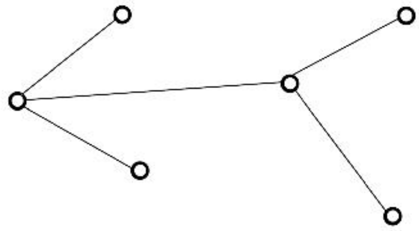
Лекция 12
Деревья, лес
Остовные деревья, остовной лес
Поиск остовных деревьев с минимальным весом
Обход графа
ПУТИ И ЦИКЛЫ ЭЙЛЕРА
ПЛОСКИЕ И ПЛАНАРНЫЕ ГРАФЫ

План лекции
1. Определение дерева, свойства деревьев
2. Теорема Кэли
3. Процедура построения остовного леса
4. Свойства циклического ранга
5. Фундаментальная система циклов графа
6. Обход графов
6.1. Обход в глубину
6.2. Обход в ширину
6.3. Программа обхода графов в глубину
7.Пути и циклы Эйлера.
7.1. Эйлеров цикл в ориентированном графе.
8. Гамильтоновы циклы (Основные определения) .
9. Плоские и планарные графы.
9.1. Общие понятия о плоском графе.
9.2. Укладка графа на поверхности.
9.3. Теорема Жордана.
9.4. Теорема об укладке графа в трехмерном пространстве.
10. Непланарные графы.
11. Грани плоского графа.
12. Теорема Эйлера.
13. Гомеоморфные графы.
14. Теорема Понтрягина-Куратовского.
15. Операция стягивания.
16. Теорем Вагнера.

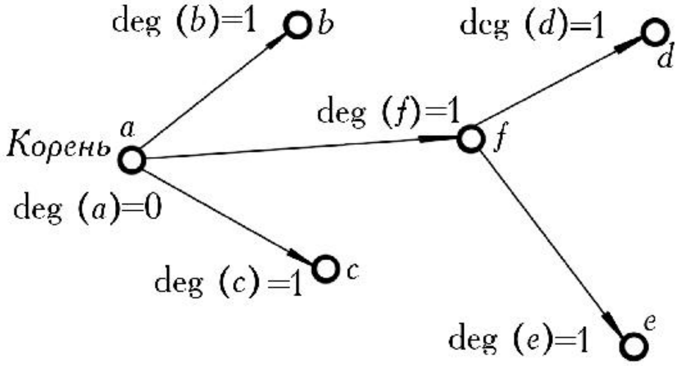
Определение дерева. Свойства деревьев
Неориентированным деревом называется связный неориентированный граф без циклов.



Корневым деревом называется такое дерево, в котором выделена вершина, называемая *корнем.*

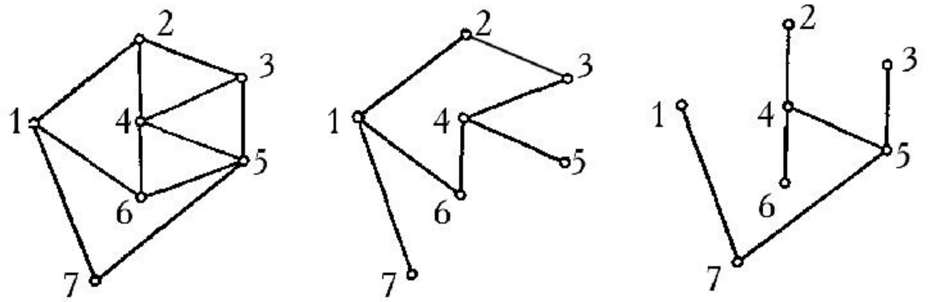
Корень в неориентированном графе – это одна из вершин, выбранная по желанию наблюдателя.

Ориентированным деревом называется связный ориентированный граф без циклов, в котором полустепень захода каждой вершины, за исключением корневой, равна единице, а полустепень захода корневой вершины равна 0.



Остовным деревом графа *G* называется остовной подграф графа *G*, являющийся деревом.

Остовным подграфом называется такой подграф, в котором множество его вершин совпадает с множеством вершин самого графа.

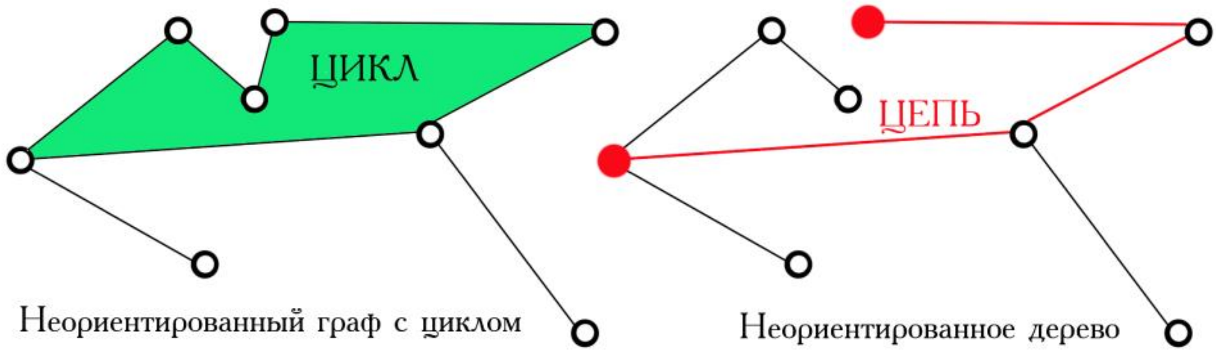


Исходный граф **Остовной подграф** **Остовное дерево**

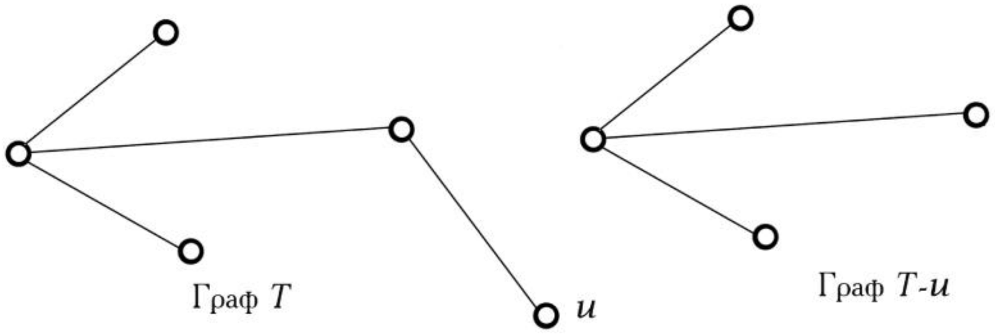
Теорема 1. **Граф является деревом тогда и только тогда, когда любые две его вершины связаны единственной цепью.**

Доказательство. Пусть граф является деревом. Если допустить существование более одной цепи, связывающей любые две его вершины, то в таком графе существует цикл, то есть граф не может быть деревом.

Наоборот, поскольку любые две вершины графа соединены цепью, то граф связный, а в силу того, что эта цепь единственная, он не имеет циклов. Значит, граф является деревом. Теорема доказана.



Следствие 1. Если T – дерево и u – его концевая вершина, то граф $T-u$ $(T$ *минус* $u)$ – дерево.
Действительно, граф $T-u$ – подграф дерева T, для которого выполняются все условия теоремы.

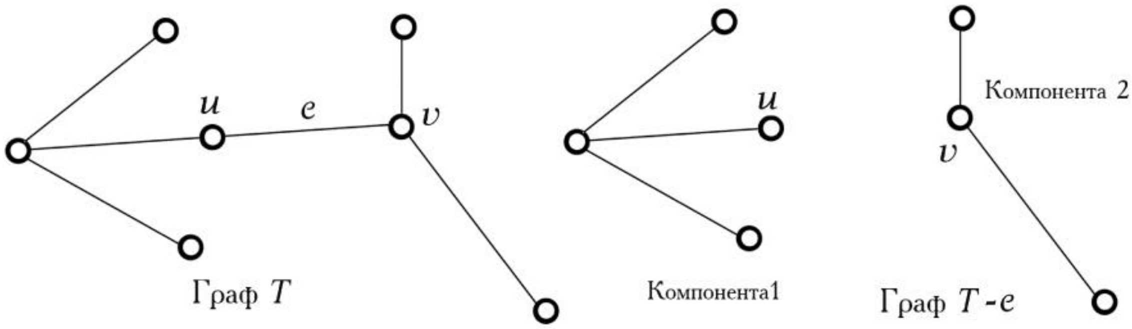


Следствие 2. Всякое непустое дерево имеет по крайней мере две висячие вершины и одно висячее ребро.
1. **Висячая вершина в неориентированном граф**е – это вершина степени 1.
2. **Висячая вершина в орграфе** – вершина с полустепенью захода, равной 1, и полустепенью исхода, равной 0.
3. **Висячее ребро** – это ребро, инцидентное вершине со степенью 1.

Определение. Ребро связного графа называется *существенным*, если его удаление ведет к нарушению связности этого графа.

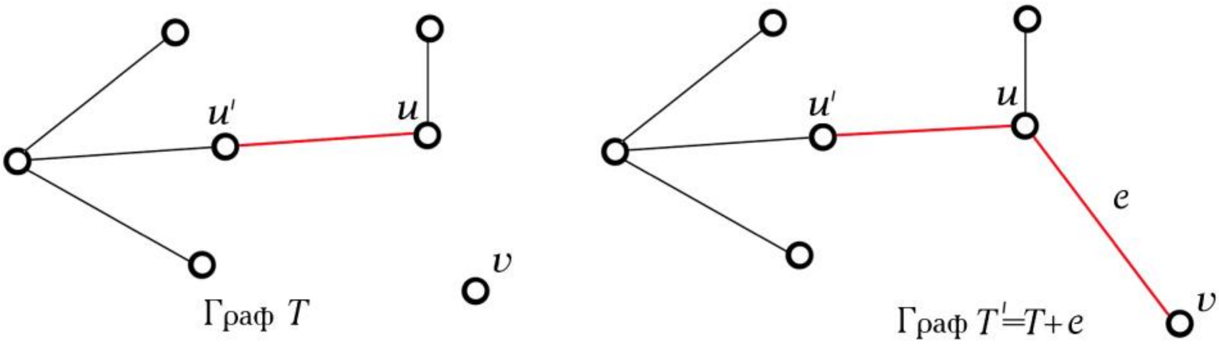
Следствие. В неориентированном графе существенным ребром является мост.
Теорема. В дереве каждое ребро существенное.
Доказательство. Доказательство вытекает из того, что удаление ребра $e=(u,v)$ в дереве T в силу единственности цепи, соединяющей вершины u и v, ведет к появлению двух компонент связности: одна содержит вершину u и другая – вершину v.

Следовательно, граф $T-e$ не является связным.



Теорема 2. *Если* $T=(V,E)$ *– дерево и вершина* $v \notin V$, *то граф* $T'=\left(V\cup\{v\}, E\cup\{(u,v)\}\right)$, *где* u *– произвольная вершина из* V, *тоже является деревом.*

Доказательство. Поскольку T – дерево, то существует единственная цепь, соединяющая любую вершину u' с вершиной u. Поскольку вершина $v \notin V$, то добавление одного концевого ребра (u,v) приводит к тому, что из каждой вершины u' имеем лишь единственную цепь, которая соединяет вершины u' и v. В силу теоремы 1 граф T' является деревом.



Теорема 3. *Пусть граф* T *имеет* n *вершин. Тогда эквивалентными являются такие утверждения:*

1. T является деревом.

2. T не имеет циклов и имеет $n-1$ ребро.

3. T – связный граф и имеет $n-1$ ребро.

4. T – связный граф и каждое его ребро является мостом.

5. Любые две вершины графа T соединены ровно одной простой цепью.

6. T не имеет циклов, но добавление любого нового ребра в T способствует возникновению ровно одного цикла.

Определение. Лес – несвязный n-граф без циклов;
1. Связные компоненты леса являются деревьями.
2. Любая часть леса или дерева также не имеет циклов.
3. Любая часть леса является лесом или деревом.

Теорема. Пусть лес G содержит n вершин и k компонент. Тогда лес G имеет $n-k$ ребер.

Доказательство. Из условия 2 теоремы 3 вытекает, что каждая компонента G_i имеет (n_i-1) ребро. Но тогда число ребер в G равно

$$(n_1-1)+(n_2-1)+\ldots+(n_k-1)=n_1+n_2+\ldots+n_k-k=n-k,$$

что и следовало доказать.

Теорема Кэли. Число разных деревьев, которые можно построить на n вершинах, равно n^{n-2}.

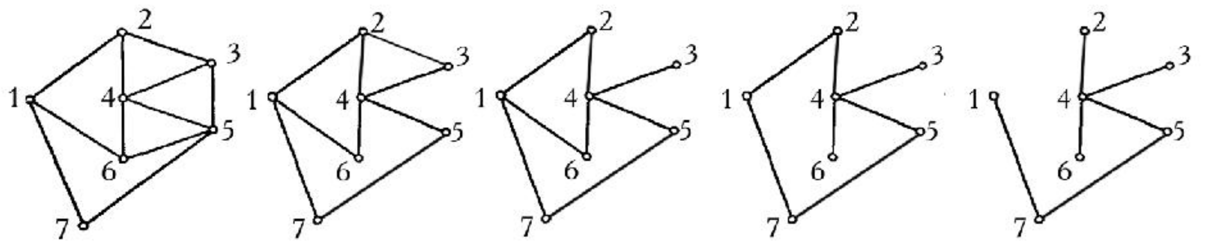
Процедура построения остовного дерева

1. Удаление из связного графа G одного ребра, принадлежащего некоторому циклу, не нарушает связности графа G.

2. Применим процедуру удаления ребра к одному из циклов в графе G.

3. Будем повторять удаление до тех пор, пока в G не останется ни одного цикла.

4. В результате получим дерево, содержащее все вершины графа G. Это дерево называется *остовным деревом графа* G.



T $\quad$ $T1=T-(3,5)$ $\quad$ $T2=T1-(2,3)$ $\quad$ $T3=T2-(1,6)$ $\quad$ $T4=T3-(1,2)$

Процедура построения остовного леса

Пусть G представляет собой граф с n вершинами, m ребрами и k компонентами.

1. Применим процедуру удаления ребра к одному из циклов в каждой компоненте связности графа G.

2. Будем повторять удаление до тех пор, пока в каждой компоненте G не останется ни одного цикла.

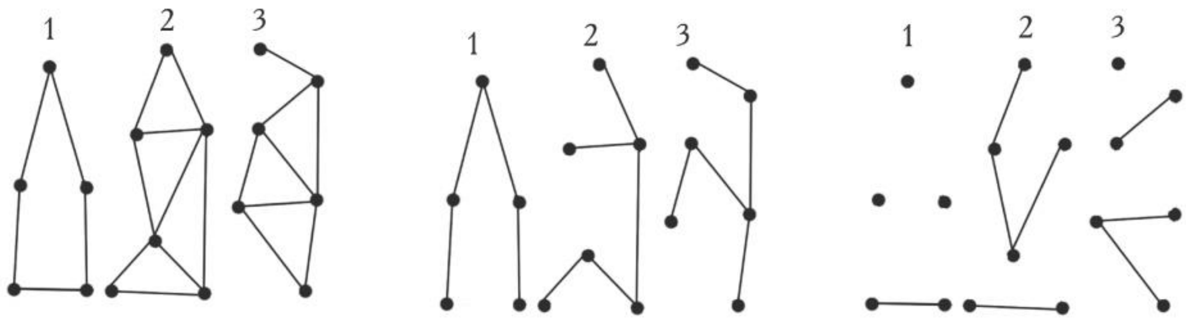
3. В результате получим граф, который называется *остовным лесом.*

4. Число ребер, которые при этом удаляются, называется *цикломатическим числом* или *циклическим рангом графа* G и обозначается $C(G)$. Таким образом, цикломатическое число является мерой связности графа.

Свойства циклического ранга

1. **Циклический ранг дерева** равен нулю.
2. **Циклический ранг циклического графа** равен единице.

Циклический граф – связный регулярный граф степени 2, единственная компонента связности циклического графа является простым циклом.
Остовное дерево графа – это дерево, содержащее все вершины графа.
Остовным лесом называется несвязный граф, состоящий из компонент, являющихся остовными деревьями.
Теорема. Пусть T – остовной лес графа G . Граф G', полученный из графа G путем удаления всех ребер графа T, называется *дополнением* остовного леса T графа G.

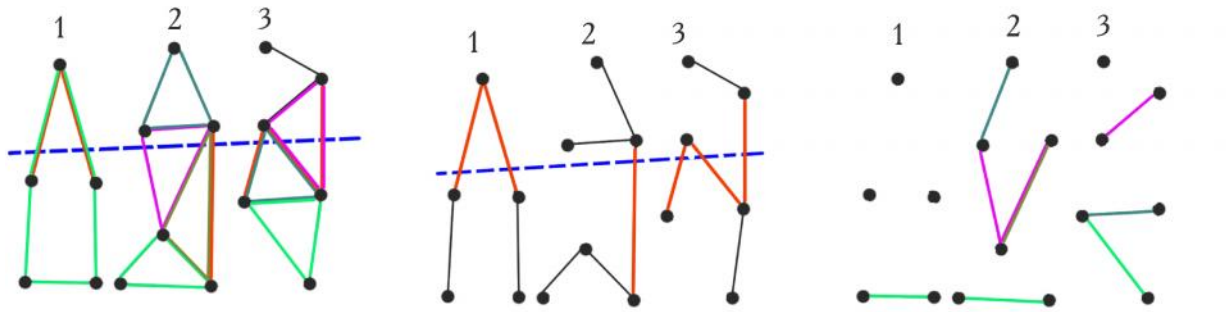


Несвязный граф G | **Остовной лес** T | **Дополнение остовного леса** G'

Теорема 4. Если T – остовной лес графа G , то

а) всякое сечение в G имеет общее ребро с T ;

б) всякий цикл в G имеет общее ребро с дополнением T **.**



Несвязный граф G | **Остовной лес** T | **Дополнение остовного леса** G'

Фундаментальная система циклов графа

Пусть T – остовной лес графа G.

Если добавить к T любое не содержащееся в нем ребро графа G, то по предложению п. 6 теоремы 3 получим единственный цикл.

Определение. Множество всех циклов, получаемых путем добавления по отдельности каждого ребра из G, не содержащегося в T, называется фундаментальной системой циклов, ассоциированной с Т.

Циклы данной фундаментальной системы будут различными, и их число равняется циклическому рангу графа G.

На рисунке показаны графы, демонстрирующие фундаментальную систему циклов.

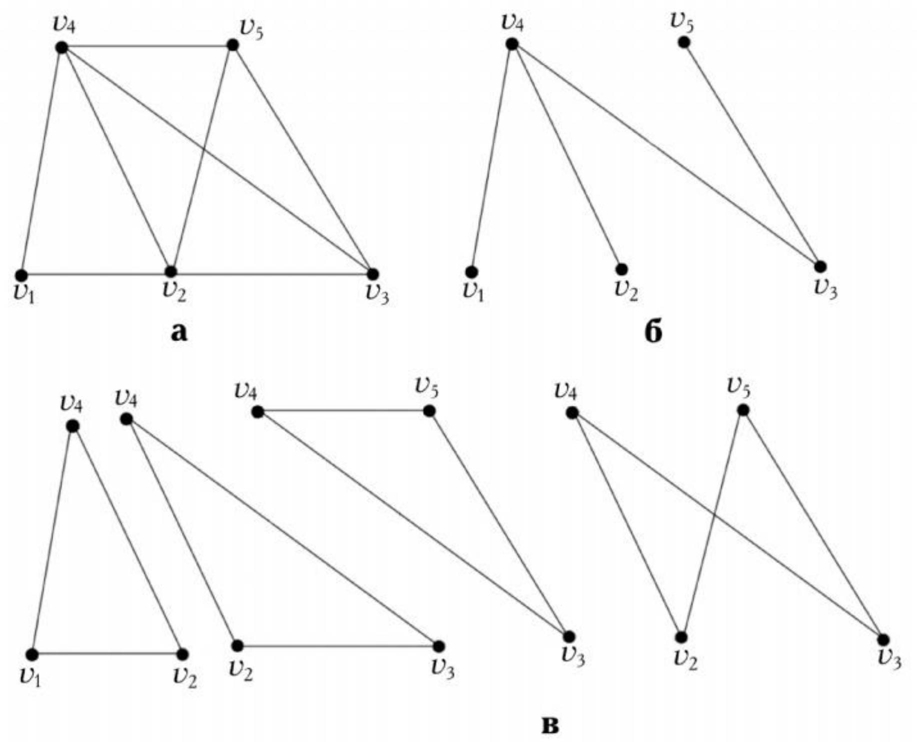


Рисунок: **а** – граф G; **б** – остовное дерево графа G; **в** – фундаментальная система циклов, ассоциированная с остовным деревом графа G.

Теорема 5. Пусть $G=(V,E)$ – произвольный конечный граф. Число ребер графа G, которые необходимо удалить для получения остовного леса T, не зависит от порядка их удаления и равно $C(G)=|E|-|V|+k$, где k – число компонентов связности графа G.

Следствие. Граф G представляет собой остовной лес тогда и только тогда, когда $C(G)=0$.

Следствие. Граф G имеет единственный цикл тогда и только тогда, когда $C(G)=1$.

Следствие. Граф G, у которого число ребер превышает число вершин, имеет цикл.

Следствие. Всякое дерево порядка $n \geq 2$ имеет по крайней мере две концевые вершины.

Остов наименьшего веса

Постановка задачи

Пусть каждому ребру графа $G=(V,E)$ поставлено в соответствие некоторое число

$$d_i = d(e_i), \quad e_i \in E, \quad i = 1,2,...,|E|.$$

Необходимо в графе G найти остовное дерево, сумма чисел d_i в котором наименьшая.

$$S = \min \sum_{e_i \in E} \left(d(e_i) \right)$$

Число d_i в этом случае называется *весом ребра* e_i, а сам граф G – таким, что имеет *вес*.

Следовательно, задача состоит в том, чтобы найти остовное дерево с наименьшим весом.

Практические приложения задачи поиска остовного дерева с минимальным весом

Эта задача возникает при проектировании дорог, линий электропередач, трубопроводов и пр., когда необходимо соединить заданные точки некоторой системой связи так, чтобы общая длина линий связи была минимальной.

Рассмотрим несколько методов решения этой задачи.

Решение, следующее из теоремы Кэли

1. Рассмотрим все возможные остовные деревья полного графа G с n вершинами.

Согласно теореме Кэли ч*исло разных деревьев, которые можно построить на n вершинах, равно n^{n-2}.*

2. При выборе каждого дерева определим сумму его весов.

3. Применив сортировку по величине суммы весов, в начале списка получим те остовные деревья, которые имеют минимальную сумму весов.

Но поскольку число n^{n-2} даже при небольшом n будет большим, то такой путь решения данной задачи является достаточно трудоемким.

Поэтому актуальным является использование более эффективных алгоритмов.

Предварительные замечания

1. **Порядок графа** – число, соответствующее количеству вершин графа.
2. **Пустой граф** – граф, не содержащий ребер или регулярный граф степени 0.

Алгоритм Краскала

Алгоритм Краскала состоит в выполнении такой последовательности действий:

1. Берем пустой граф O и строим граф $T_1 = O + e_1$,

где e_1 – ребро графа $G = (V, E)$ минимального веса.

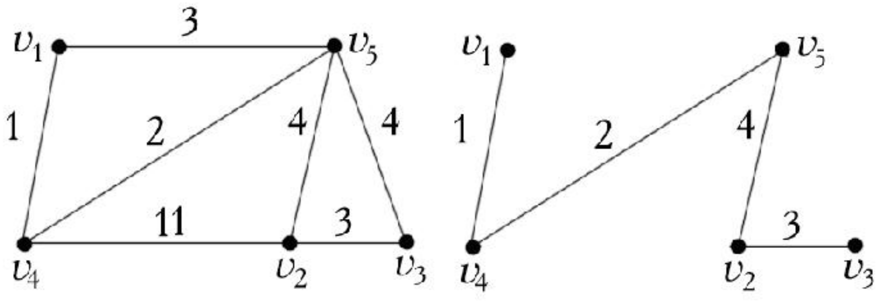
2. Если граф T_k уже построен и $k < n-1$, то строим граф $T_{k+1} = T_k + e_{k+1}$, где e_{k+1} – ребро минимального веса среди ребер графа G, не вошедших в граф T_k, которое не составляет цикла с ребрами графа T_k.

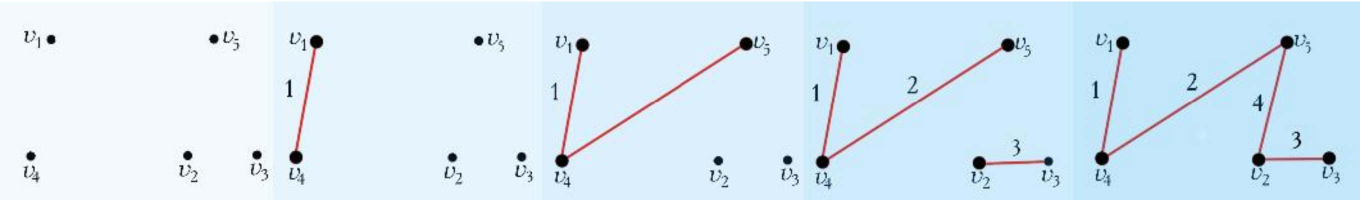
Этот алгоритм базируется на специальной теореме.

Теорема 7. Пусть G – связный граф порядка n и T – подграф графа G, полученный в результате выполнения шагов 1 и 2 алгоритма Краскала. Тогда подграф T – остовное дерево графа G минимального веса.

Пример. Пусть дан граф G, показанный на рисунке слева. Необходимо найти остовное дерево минимального веса этого графа с помощью алгоритма Краскала.

Решение.





Свойства алгоритма Краскала

1. Алгоритм используется для взвешенных неориентированных графов.

2. Алгоритм может быть использован для построения остовного леса в многокомпонентных несвязных графах.

В этом случае **структуры данных**, описывающие каждую из компонент связности, **должны быть отдельными**.

3. Вычислительная сложность алгоритма Краскала $O(e \cdot \log e)$, где e – количество ребер в данном графе.

Алгоритм Прима

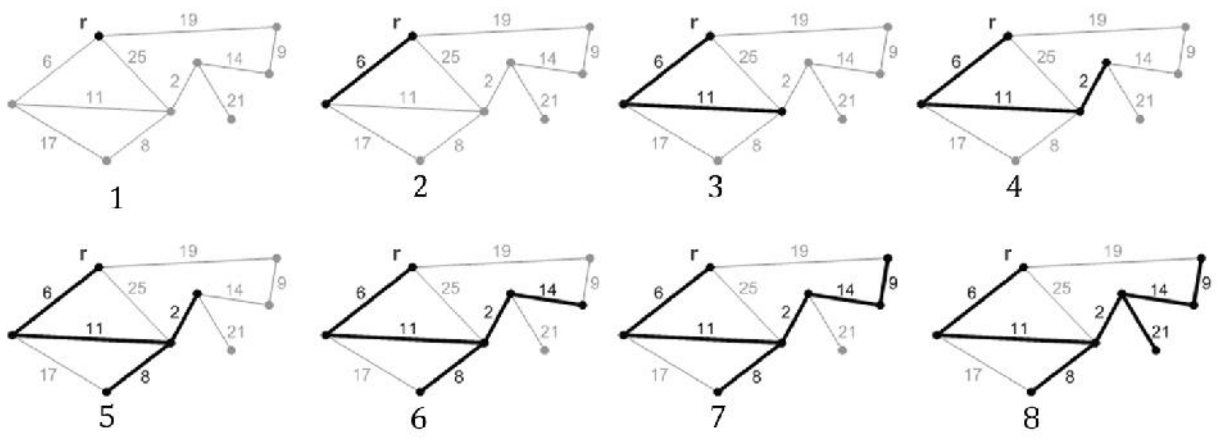
Алгоритм Прима состоит в выполнении такой последовательности действий:

1. В пустом графе O выбираем произвольную **вершину**.
2. Выбираем ребро e_1 минимального веса к смежной вершине и строим подграф $T_1 = O + e_1,$.
3. Если граф T_k уже построен и $k < n-1$, то строим граф $T_{k+1} = T_k + e_{k+1}$, где e_{k+1} – ребро минимального веса, соединяющее концевую вершину графа T_k со смежной вершиной, не включенной в множество вершин графа T_k.

Пример. Пусть дан граф G. Необходимо найти остовное дерево минимального веса этого графа с помощью алгоритма Прима.

Решение. 1. Выбираем произвольную вершину r и проводим инцидентное ребро минимального веса.

2. Затем, просматривая инцидентные ребра на каждой из концевых вершин, находим ребро минимального веса.



Свойства алгоритма Прима

1. Алгоритм используется для взвешенных неориентированных связных графов.

2. Вычислительная сложность алгоритма Прима $O\left(n^2\right)$, где n – количество вершин графа. Если значение n достаточно большое, то использование этого алгоритма не рационально.

3. Если e значительно меньше n^2, то алгоритм Краскала предпочтительнее, но если e близко к n^2, рекомендуется применять алгоритм Прима.

Структура алгоритма Прима

Дан граф $G\left(V,E\right)$, где множество вершин $V=\{1,2,...,i,...,n\}$

$U=\varnothing$ – множество вершин остовного дерева.

$T=\varnothing$ – множество ребер остовного дерева.

procedure Prim (*G*: граф; ***var*** *T*: множество ребер;
var *U*: множество вершин;
u ,*v*: вершина;
begin
T := $\varnothing$; *U* := {i};
while *U* ≠ *V* ***do***
begin
Нахождение ребра (*u* ,*v*) наименьшей стоимости и такого, что $u\in U$ и $v\in V\setminus U$
T:= *T* ∪ {(*u* ,*v*)};
U:= *U* ∪ {*v*}
end
end.

Обход графов

При решении многих задач с использованием графов необходимо уметь обходить все вершины и ребра (дуги) графа. Обойти граф — это побывать во всех вершинах точно по одному разу. Существует два классических алгоритма обхода графов: обход графа в глубину и обход графа в ширину.

Работа всякого алгоритма обхода состоит в последовательном посещении вершин и исследовании ребер. Какие именно действия выполняются при посещении вершины и исследовании ребра – зависит от конкретной задачи, для решения которой производится обход. В любом случае факт посещения вершины запоминается, так что с момента посещения и до конца работы алгоритма она считается посещенной.

Вершину, которая ещё не посещена, называют новой. В результате посещения вершина становится открытой и остается такой, пока не будут исследованы все инцидентные ей ребра. После этого она превращается в закрытую.

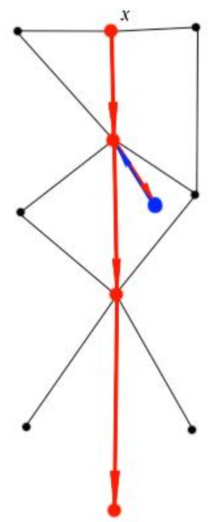
Поиск в глубину

Обход в глубину – это обход графа по следующим правилам:

(1) находясь в вершине x, нужно двигаться в любую другую, ранее не посещенную вершину (если таковая найдется), одновременно запоминая дугу, по которой мы впервые попали в данную вершину;

(2) если из вершины x мы не можем попасть в ранее не посещенную вершину или таковой нет, то мы возвращаемся в вершину z, из которой впервые попали в x, и продолжаем обход в глубину из вершины z.

При выполнении обхода графа по этим правилам мы стремимся проникнуть "вглубь" графа так далеко, как только это возможно, затем отступаем на шаг назад и снова стремимся пройти вперед, и так далее.



Таким образом, допустим, что мы находимся в вершине графа v. Далее, выбираем произвольную, но еще не просмотренную вершину u, смежную с вершиной v. Эту новую вершину рассматриваем теперь уже как v и, начиная с нее, продолжаем обход. Если не существует ни одной новой (еще не просмотренной) вершины, смежной с v, то говорят, что вершина v просмотрена, и возвращаются в вершину, из которой мы попали в v.

Главное отличие от поиска в ширину состоит в том, что в качестве активной выбирается та из открытых вершин, которая была посещена последней. При обходе в глубину чем позднее будет посещена вершина, тем раньше она будет использована.

Поиск в глубину ищет любой возможный путь – то есть первый попавшийся.

Пример обхода графа в глубину

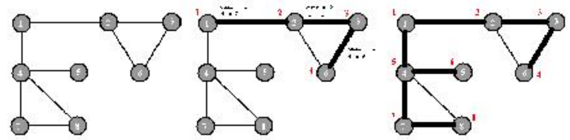
Поиск в глубину начинаем от какой-либо вершины. Рекурсивно применим ко всем вершинам, в которые можно попасть из текущей, такую последовательность действий.

1. Зададим граф из N вершин матрицей смежности A размером $N \times N$ в виде массива $A[N,N]$.

2. Зададим одномерный массив $Visited[N]$ и заполним его нулями, считая, что ни одна вершина до начала выполнения алгоритма не была посещена.

3. Зададим рекурсивную процедуру обхода графа в глубину.

```
Procedure Go(Curr:Integer);
begin
 Visited[Curr]:=1; {Помечаем текущ. верш. как пройденную}
 For i=0 to N do
 begin
  If Visited[i]=0 AND (A[Curr,i]=1) then Go(i);
 end;
end;
Program Depth
Begin Go(Start) end.
```



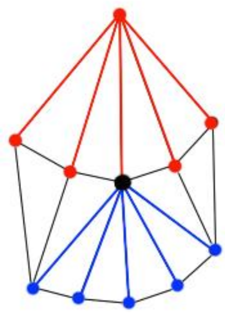
Программа, описывающая обход графов в глубину (Pascal).

```
Program graff;
Var n, v, u: integer;
        fr: text;
        gr: array[1..30, 1..30] of integer;
        nov: array[1..15] of boolean;

  procedure dfs (v: integer);
  Var u: integer;
  begin
    readln;
    write (v, ‘ ‘);
    nov [v]:=false;
    for u:=1 to n do
    if (gr[v,u]>0) and (nov[u]) then dfs (u);
  end;

begin
    n:=3; (*Задаем количество вершин графа*)
   for v:=1 to n do
   begin
      nov [v]:=true;
      writeln;
      for u:=1 to n do
```

```
      begin
        nov[u]:=true;
         (*Ввод инциденции для вершин u и v*)
        write (‘ gr[‘,v,u,’]=’);
        read (gr [v, u] );
      end;
  end;
  for v:=1 to n do
  begin
    (*Вызов процедуры обхода*)
    if nov[v] then dfs (v);
  end;
  readln;
end.
```

Поиск в ширину

Основная особенность поиска в ширину, отличающая его от других способов обхода графов, состоит в том, что в качестве активной вершины выбирается та из открытых, которая была посещена раньше других. Именно этим обеспечивается главное свойство поиска в ширину: чем ближе вершина к старту, тем раньше она будет посещена.



Использование вершины происходит с помощью просмотра сразу всех еще не просмотренных вершин, смежных этой вершине. Таким образом, поиск ведется во всех возможных направлениях одновременно.

Сначала посещается сама вершина a, затем все вершины, смежные с a, находящиеся от нее на расстоянии 1, затем вершины, находящиеся от a на расстоянии 2, и т. д. Чем ближе вершина к стартовой вершине, тем раньше она будет посещена. Поиск в ширину ищет наиболее короткий возможный путь.

Пример обхода графа в ширину

При поиске в ширину вместо стека рекурсивных вызовов **используется очередь**, в которую записываются вершины в порядке удаления от начальной.

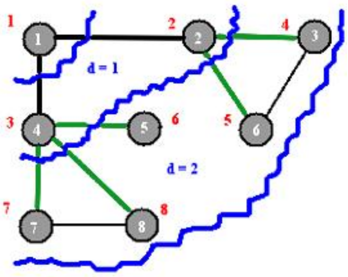
1. **Зададим массив** $Visited[N]$ и заполним его нулями, считая, что ни одна вершина до начала выполнения алгоритма не была посещена.

2. **Создадим очередь** для хранения вершин в виде массива $Queue[N]$. В начало очереди запишем начальную вершину.

3. **Переменная** r указывает на позицию в очереди, из которой мы **читаем** данные.

4. **Переменная** w указывает на позицию в очереди, куда мы будем **писать** данные.
5. Зададим процедуру обхода в ширину.

```
r:= 0, w:=1;

While (r < w) do
begin
  r:=r+1;
  Curr = queue[r]; {выбрали активную вершину}
  For i:= 1 to N do {посещаем все смежные вершины}
  begin
   if ( (Visited[i]=0) AND (A[curr,i]=1) do
   begin
    Visited[i]:= 1; {отмечаем посещенную вершину}
    w:=w+1;
    queue[w]:= i; {ставим ее в очередь активных вершин}
   end;
  end;
end;
```



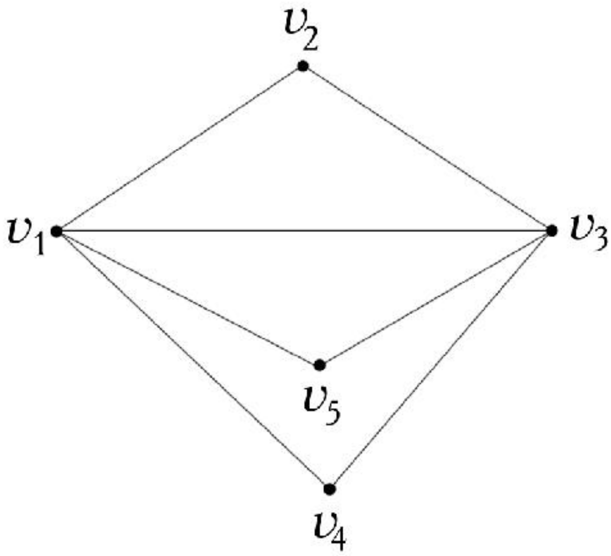
Пути и циклы Эйлера

Определение. Пусть $G=(V,E)$ — граф. Цикл, который включает все ребра и вершины графа G, называется *эйлеровым циклом.* Если это условие выполняется, говорят, что граф G имеет эйлеров цикл.

Если теперь вернуться к задаче о кенигсбергских мостах, легко убедиться, что она сводится к попытке определить, содержит ли граф, изображающий задачу, эйлеров цикл. Для этой цели нам потребуется приведенная ниже теорема. Эта теорема справедлива также для мультиграфов. Более того, доказательство теоремы остается тем же. Для ясности будем использовать термин граф, понимая, что каждое утверждение справедливо для мультиграфов.

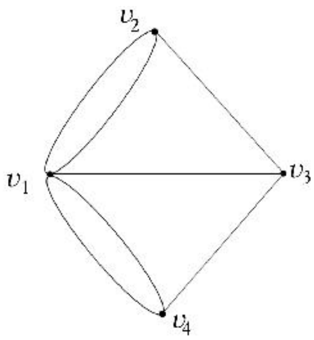
Теорема. Граф с более чем одной вершиной имеет эйлеров цикл тогда и только тогда, когда он связный и каждая его вершина имеет четную степень.

Пример. Приведенный на рисунке граф имеет эйлеров цикл, поскольку степень каждой его вершины четная.

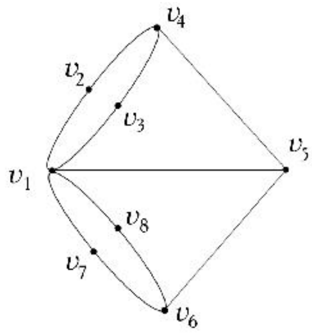


Примеры эйлеровых циклов: $(v_1, v_2, v_3, v_4, v_1, v_5, v_3, v_1)$,
$(v_4, v_3, v_2, v_1, v_{3,} v_5, v_1, v_4)$ и т. д.

Возвращаясь к задаче о кенигсбергских мостах, обнаруживаем, что мультиграф, построенный Эйлером для описания кенигсбергских мостов, имеет нечетные степени всех его вершин. Итак, этот мультиграф не имеет эйлерова цикла, поэтому невозможно пройти каждый мост по одному разу и вернуться в исходную точку пути.



Как видно из рисунка, задача о кенингсберских мостах была решена с использованием мультиграфа. Однако эту задачу можно решить, используя простой граф. Для этого исходный мультиграф необходимо привести к простому графу путем введения дополнительных вершин:



Если поставить задачу не возвращаться в исходную точку, то предыдущая задача сводится к поиску пути в графе.

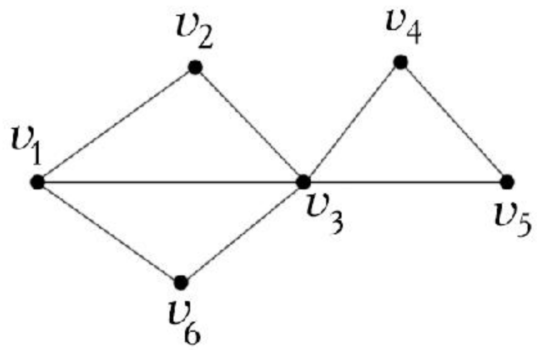
Определение. Пусть $G(V,E)$ — граф. Путь, который включает каждое ребро графа G только один раз, называется *эйлеровым путем*. В этом случае говорят, что граф G имеет эйлеров путь.

В случае, когда эйлеров путь не является эйлеровым циклом, его называют собственным эйлеровым путем.

Теорема. Граф или мультиграф имеет собственный эйлеров путь тогда и только тогда, когда он связный и ровно две его вершины имеют нечетную степень.

Поскольку граф для кенигсбергских мостов имеет четыре вершины с нечет− ными степенями, можно сделать вывод о невозможности пройти каждый мост по одному разу, даже если не нужно возвращаться в исходную точку маршрута.

Пример. На рисунке показан граф, обладающий эйлеровым путем, поскольку две его вершины имеют нечетную степень.



Пример эйлерова пути: $v_1, v_2, v_3, v_1, v_6, v_3, v_4, v_5, v_3$.

Эйлеров цикл в ориентированном графе.

Определение. Пусть $G=\left(V,E\right)$ — ориентированный граф. *Ориентированным циклом* называется ориентированный путь ненулевой длины из вершины в ту же вершину без повторения ребер.

Определение. Пусть $G=\left(V,E\right)$ — ориентированный граф. Ориентированный цикл, который включает все ребра и вершины графа G, называется *эйлеровым циклом*. В этом случае говорят, что ориентированный граф G имеет эйлеров цикл.

Теорема. Ориентированный граф имеет эйлеров цикл тогда и только тогда, когда он связный и полустепень входа каждой вершины равна ее полустепени выхода.

Алгоритм построения эйлерова цикла

1. Выходим из произвольно выбранной вершины v и каждое пройденное ребро зачеркиваем.
2. Никогда не идем по тому ребру e, которое в настоящий момент является мостом (т. е. при удалении которого граф, образованный незачеркнутыми ребрами, распадается на две компоненты связности, имеющие хотя бы по одному ребру).
3. Не выбираем ребра, ведущего в v, пока есть другие возможности.

Гамильтоновы циклы (Основные определения)

Определение. *Гамильтоновой цепью* графа называется его простая цепь, которая проходит через каждую вершину графа точно один раз.

Определение. Цикл графа, проходящий через каждую его вершину, называется *гамильтоновым циклом*.

Определение. Граф называется *гамильтоновым*, если он обладает гамильтоновым циклом.

Определение. Граф, который содержит простой путь, проходящий через каждую его вершину, называется *полугамильтоновым.*

Это определение можно распространить на ориентированные графы, если путь считать ориентированным. Гамильтонов цикл не обязательно содержит все ребра графа. Ясно, что гамильтоновым может быть только связный граф и что всякий гамильтонов граф является полугамильтоновым. Заметим, что гамильтонов цикл существует далеко не в каждом графе.

Замечание: Любой граф G можно превратить в гамильтонов граф, добавив достаточное количество вершин. Для этого, например, достаточно к вершинам

$v_1,...,v_p$ графа G добавить вершины $u_1,...,u_p$ и множество ребер $\{v_i,u_i\}$ и $\{u_i,v_{i+1}\}$.

Степенью вершины v называется число ребер $d(v)$, инцидентных ей, при этом петля учитывается дважды. В случае ориентированного графа различают степень $do(v)$ по выходящим дугам и $di(v)$ – по входящим.

Теорема. Пусть G имеет $p \geq 3$ вершин. Если для всякого n, $1 \leq n \leq \frac{p-1}{2}$, число вершин со степенями, не превосходящими n, меньше чем n, и для нечетного p число вершин степени $\frac{p-1}{2}$ не превосходит $\frac{p-1}{2}$, то G – гамильтонов граф.

Следствие 1. Если $p \geq 3$ и $\deg u + \deg v \geq p$ для любой пары u и v несмежных вершин графа G, то G – гамильтонов граф.

Следствие 2. Если $p > 3$ и $\deg v \geq \frac{p}{2}$ для любой вершины v графа G, то G – гамильтонов граф.

Теорема. В полном графе $G(V,E)$ всегда существует гамильтонов путь.

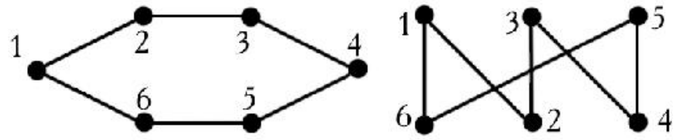
Плоские и планарные графы

Общие понятия о плоском графе

В определении графа как геометрической фигуры до этого мы не накладывали никаких ограничений на расположение этой фигуры в пространстве. Теперь же будем говорить, что граф изображен на поверхности (плоскости, сфере, и т. п.), если все его **вершины и ребра принадлежат этой поверхности**.

Определение. Граф, изображенный на плоскости, называется *плоским,* если его ребра не пересекаются в точках, отличных от вершин графа.

Отметим, что свойство графа ***быть или не быть плоским*** *— это свойство геометрического изображения, а не алгебраического объекта*



G_1 и G_2-изоморфны, G_1 - плоский, а G_2 - нет.

Общие понятия о планарном графе

Таким образом, термин «плоский граф» всегда относится к **конкретному** (одному из многих) геометрическому изображению графа.

Один и тот же граф (как множество вершин + множество ребер) может иметь **как плоские, так и не плоские изображения**.

В то же время, принципиальный вопрос, на который нужно отвечать при решении задач типа прокладки коммуникаций: «Имеет ли данный граф хотя бы одно плоское изображение?» Определим класс графов, для которых ответ на этот вопрос положителен.

Определение. Граф называется ***планарным***, если он изоморфен плоскому графу.

Иными словами. *Планарным графом* называется граф, который может быть изображен на плоскости, так что его ребра не пересекаются.

Укладка графа на поверхности

О планарных графах говорят, что они имеют **плоскую укладку**, или что они **укладываются на плоскости**.

Можно определить укладку графов не только на плоскости, но и на **других поверхностях** в пространстве.

Свойство графа укладываться на поверхности безусловно **зависит от вида этой поверхности**.

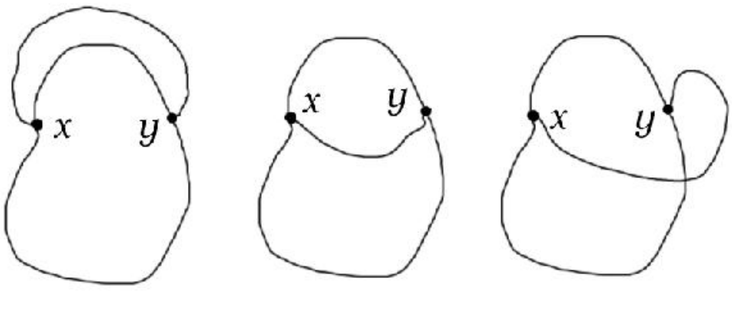
Однако многие поверхности с точки зрения укладки графов ничем **не отличаются от плоскости**.

Введем понятие жордановой кривой.

Жордановой кривой на плоскости называется непрерывная кривая линия без самопересечений.

Замкнутой жордановой кривой называется жорданова кривая, начало и конец которой совпадают.

Теорема Жордана. Если S - замкнутая жорданова кривая на плоскости, а x и y - две разные точки этой кривой, то всякая жорданова кривая, соединяющая точки x и y, либо полностью лежит внутри S, кроме точек x и y, либо вне кривой S, кроме точек x и y, либо пересекает кривую S в некоторой точке, отличной от точек x и y.

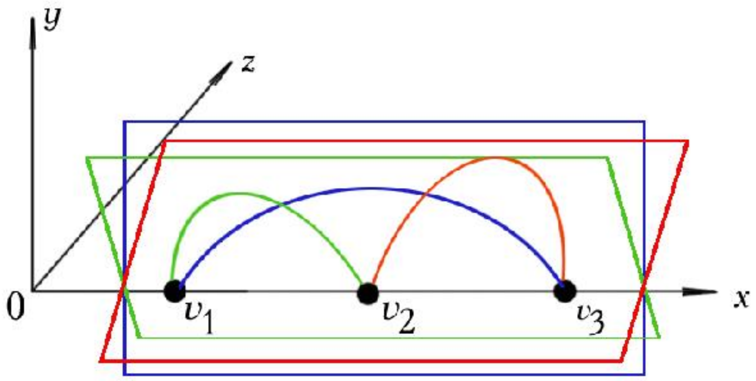


На рисунке показаны три варианта взаимного расположения жордановых кривых.

Будем говорить, что граф G укладывается в пространстве L, если существует взаимно однозначное отображение графа G вершин в точки и ребер в жордановы кривые этого пространства такое, что кривые, соответствующие разным ребрам, пересекаются в инцидентных данным ребрам вершинах. Изображенный таким образом граф G в пространстве L называется укладкой графа G.

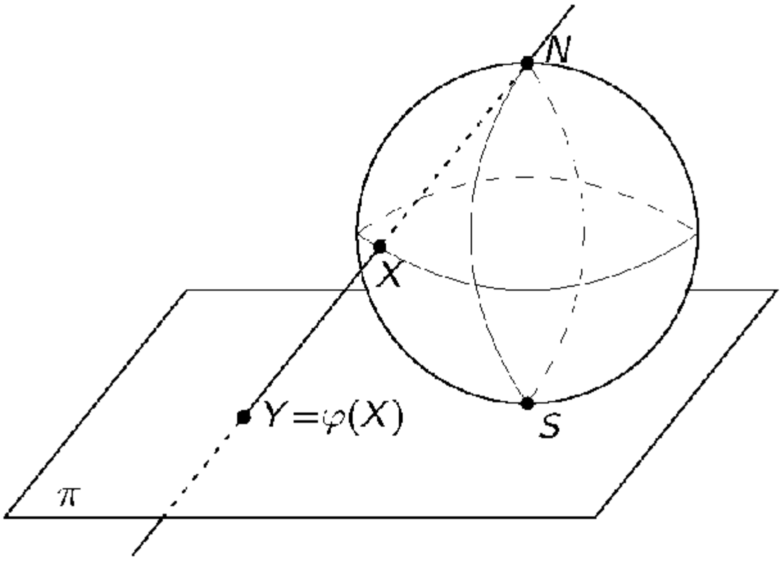
Теорема. Всякий граф укладывается в трехмерном пространстве.

Доказательство. Разместим все вершины графа $G=(V,E)$ на оси OX. Из пучка плоскостей, проходящих через эту ось, выберем $|E|$ разных плоскостей. Далее каждое ребро $(u,v)\in E$ изобразим в соответствующей плоскости полуокружностью, проходящей через вершины u и v. Ясно, что в результате получим укладывание графа G в трехмерное пространство, поскольку все ребра лежат в разных плоскостях и потому не пересекаются ни в каких точках, кроме вершин.



Теорема. Граф укладывается на сфере тогда и только тогда, когда он планарный.

Доказательство. Необходимость. Пусть граф G уложен на сфере; построим его изоморфизм на плоский граф. Для этого выберем на сфере точку N, не принадлежащую G, а в диаметрально противоположной к точке N точке S проведем к сфере касательную плоскость ® (см. рис.).

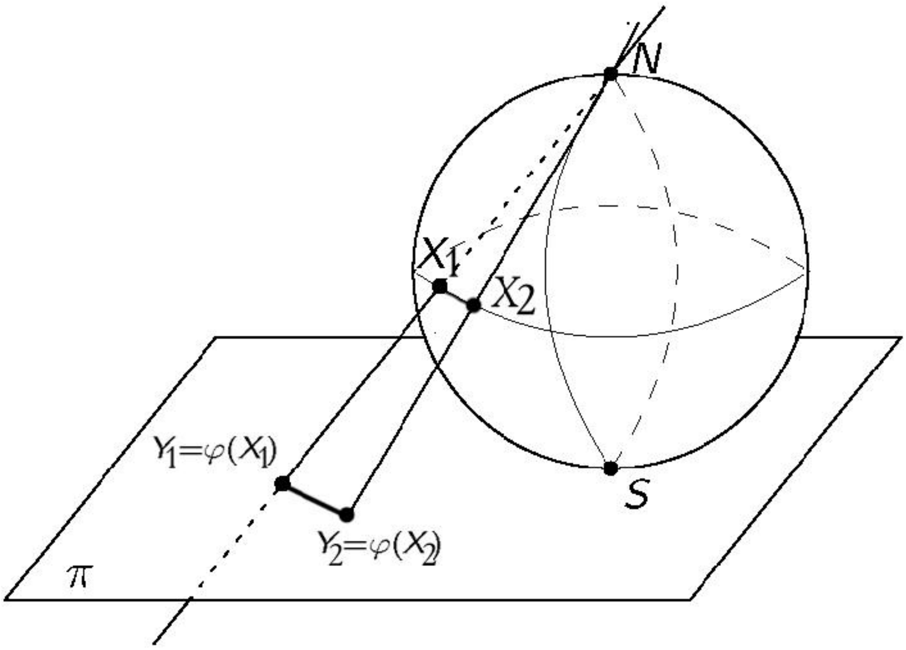


Рассмотрим произвольную прямую, проходящую через N не параллельно ® . Эта прямая не является касательной к сфере, поскольку касательные плоскости к сфере в точках N и S параллельны. Значит, такая прямая имеет ровно одну общую точку X со сферой, отличную от N, и ровно одну общую точку Y с плоскостью. Определим функцию ¨ , которая переводит любую точку X сферы, не совпадающую с N, в точку Y плоскости ®, лежащую на прямой NX

Функция ¨ называется *стереографической проекцией сферы на плоскость* ® *из точки* N .

Очевидно, что

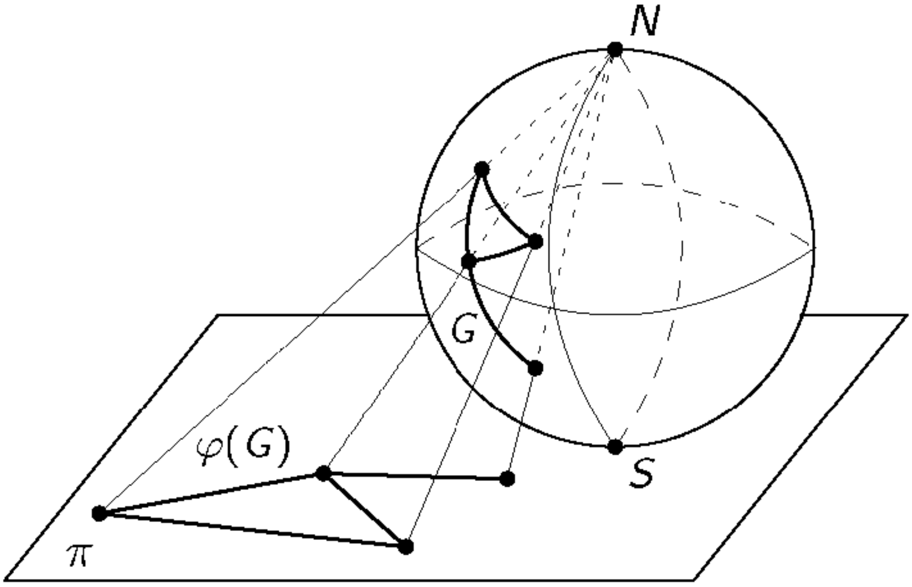
1. ¨ — биекция (разные точки сферы переходят в разные точки плоскости, а для любой точки $Y \in$ ® можно найти ее прообраз, проведя прямую YN).



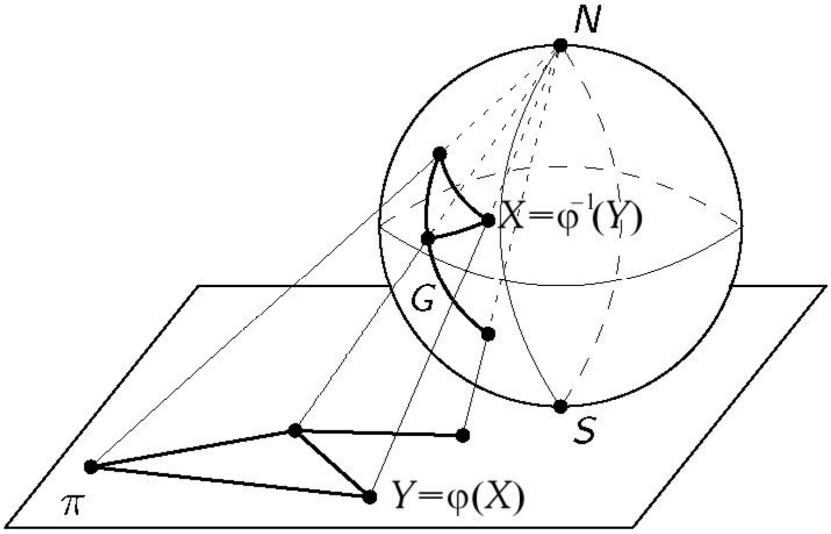
2. ¨ непрерывна (стандартными средствами математического анализа легко показать, что близкие точки на сфере переходят в близкие точки на плоскости), а значит, образом отрезка непрерывной линии на сфере является отрезок непрерывной линии на плоскости.

Из непрерывности ¨ следует, что геометрическая фигура $¨(G)$, т. е. образ графа G при функции ¨, сама является графом, как показано на рисунке (вершины и ребра $¨(G)$ являются образами вершин и ребер G).

Граф $¨(G)$ изображен на плоскости. Проверим, что он плоский.

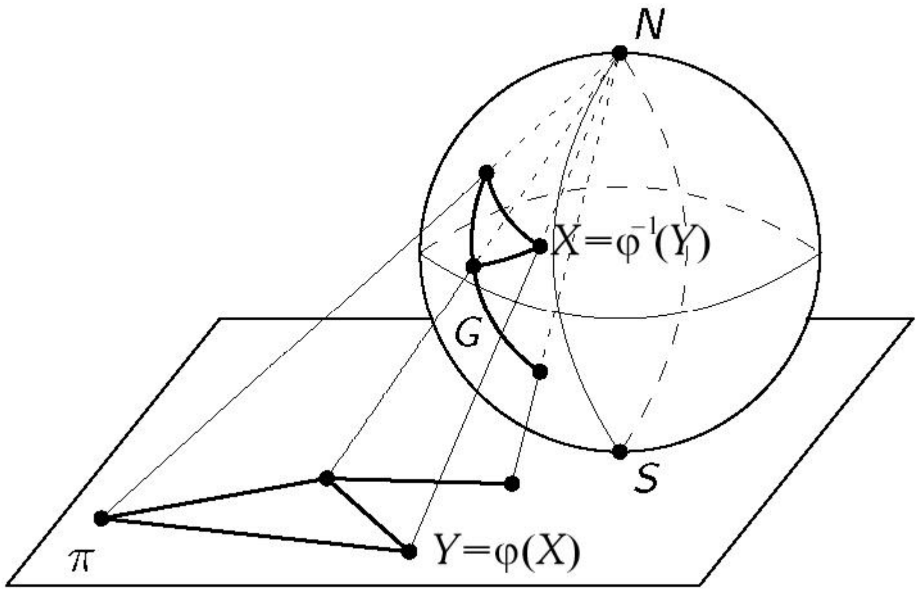


Пусть точка Y принадлежит двум ребрам $¨(G)$. Тогда точка $X = ¨^{-1}(Y)$ принадлежит двум соответствующим ребрам графа G, т. е. по условию является вершиной в G. Но тогда $Y = ¨(X)$ — вершина в $¨(G)$, что, собственно и было нужно. Осталось заметить, что функция ¨, рассматриваемая только на множестве вершин графа G, является изоморфизмом G на $¨(G)$. Тем самым, мы доказали, что граф G — планарен. Значит, планарен и любой граф, изоморфный G. Необходимость доказана.



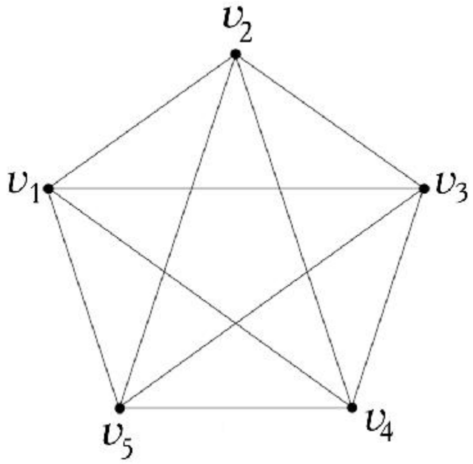
Доказательство теоремы об укладке на сфере (достаточность)

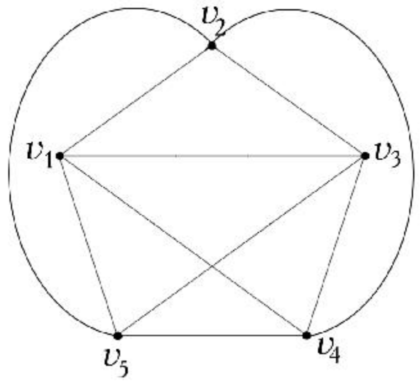
Достаточность легко доказывается с помощью той же самой биекции ¨ (точнее, с помощью биекции ¨ $^{-1}$). Меняется только начало построения: на плоскость, содержащую заданный граф, произвольным образом «ставим» сферу, после чего за N берем точку сферы, противоположную точке касания с плоскостью.



Непланарные графы

Теорема. *Графы* K_5 *и* $K_{3,3}$ *не являются планарными.*





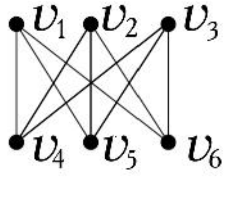
Доказательство. Предположим противное, что граф K_5 планарный. Поскольку он имеет цикл длины 5, например, v_1, v_2, v_3, v_4, v_5, то без ограничения общности можно считать, что при всякой укладке этого графа на плоскости этот цикл изображается правильным пятиугольником. По теореме Жордана ребро (v_1, v_3) должно лежать либо целиком внутри этого пятиугольника, либо целиком вне него. Третью возможность (когда ребро имеет общую точку с пятиугольником) мы не рассматриваем, поскольку имеем дело с укладкой на плоскости.

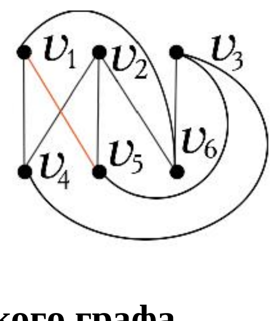
1. Рассмотрим сначала случай, когда (v_1, v_3) лежит внутри пятиугольника. Поскольку ребра (v_2, v_4) и (v_2, v_5) не должны пересекать ребро (v_1, v_3), то оба эти ребра лежат вне пятиугольника. Эта ситуация изображена на рисунке ниже.

Ребро (v_1, v_4) не должно пересекать ребро (v_2, v_5) и потому оно должно лежать внутри пятиугольника. Аналогично ребро (v_3, v_5) должно лежать внутри пятиугольника, поскольку не должно пересекаться с ребром (v_2, v_4). Однако ребра (v_1, v_4) и (v_3, v_5) обязательно пересекаются, и поэтому согласно теореме Жордана, одно из них должно лежать вне пятиугольника.

Таким образом, мы приходим к противоречию с нашим предположением. Наличие данного противоречия доказывает ошибочность первоначального предположения, т. е. граф K_5 не является планарным.

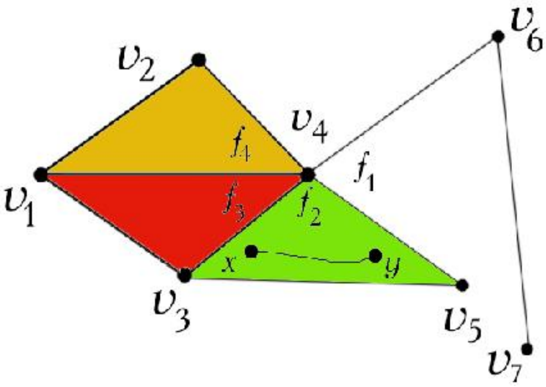
Аналогично доказывается, что граф $K_{3,3}$ не является планарным.





Грани плоского графа

Точка x плоскости S, на которой размещен граф G, называется *дизъюнктной* с G, если эта точка не соответствует никакой вершине графа G и не лежит ни на одном ребре этого графа. Две точки x и y плоскости S называются эквивалентными, если они дизъюнктны с G и их можно соединить такой жордановой кривой, все точки которой дизъюнктны с G.



Введенное отношение эквивалентности точек плоскости разбивает множество всех точек плоскости S на классы эквивалентности, которые называются гранями графа G. Например, граф G, изображенный на рисунке, имеет 4 грани: f_1, f_2, f_3, f_4. Причем грань f_1 - бесконечная.

Теорема Эйлера

Теорема. Пусть G – связный плоский граф, в котором
n - число вершин,
m - число ребер
и f - число граней.

Тогда справедливо равенство:

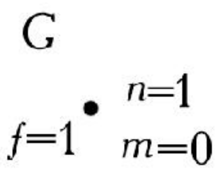
$$n + f = m + 2 .$$

Доказательство. План доказательства основывается на выполнении индукции по числу ребер в графе G.

1. Пусть $m = 0$, то $n = 1$
(поскольку по условию теоремы G - связный) и $f = 1$ (бесконечная грань).

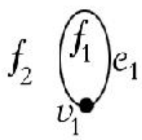
В этом случае теорема справедлива, поскольку

$$n + f = 1 + 1 = 2 \text{ и } m + 2 = 0 + 2 = 2 .$$



2. Пусть граф G представлен такими параметрами:

ребро e_1 является петлей, и в этом случае число граней увеличивается на одну, а число вершин остается неизменным



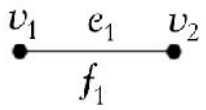
Для рассматриваемого графа $G=(V,E)$ получаем

$$V=\{v_1\},\ E=\{e_1\},\ F=\{f_1,f_2\}.$$

Поэтому $n=|V|=1$, $m=|E|=1$, $f=|F|=2$.

Отсюда $n+f=1+2=3$ и $m+2=1+2=3$.

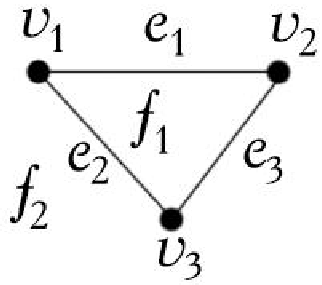
3. Пусть граф G представлен такими параметрами:
ребро e_1 соединяет две разные вершины в G.



В данном графе $G=(V,E)$ получаем $V=\{v_1,v_2\}$, $E=\{e_1\}$, $F=\{f_1\}$. Поэтому $n=|V|=2$, $m=|E|=1$, $f=|F|=1$.

Отсюда $n+f=2+1=3$ и $m+2=1+2=3$.

4. Пусть граф G представлен такими параметрами:
- связный граф G содержит несколько вершин и ребер



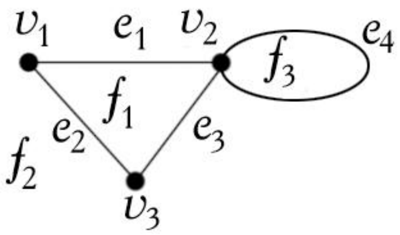
Граф $G=(V,E)$ характеризуется параметрами $V=\{v_1,v_2,v_3\}$, $E=\{e_1,e_2,e_3\}$, $F=\{f_1,f_2\}$. Поэтому

$$n=|V|=3,\ m=|E|=3,\ f=|F|=2.$$

Отсюда $n+f=3+2=5$ и $m+2=3+2=5$.

4. К графу, показанному на предыдущем рисунке, добавим петлю. В результате получим

$$V=\{v_1,v_2,v_3\},\ E=\{e_1,e_2,e_3,e_4\},\ F=\{f_1,f_2,f_3\}.$$



Тогда $n=|V|=3$, $m=|E|=4$, $f=|F|=3$.

Следовательно,

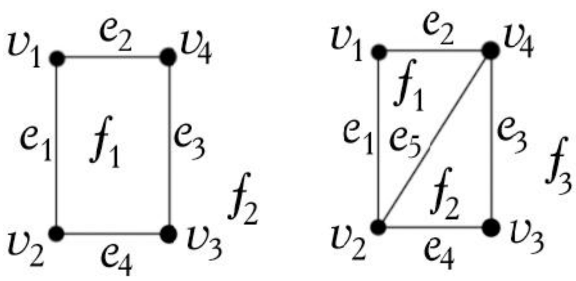
$$n+f=3+3=6,\ m+2=4+2=6.$$

5. Если к связному графу

$$V=\{v_1,v_2,v_3,v_4\},\ E=\{e_1,e_2,e_3,e_4\}\ F=\{f_1,f_2\}$$

добавить ребро, соединяющее две его вершины, получим граф

$V=\{v_1,v_2,v_3,v_4\},\ E=\{e_1,e_2,e_3,e_4,e_5\}\ F=\{f_1,f_2,f_3\}$



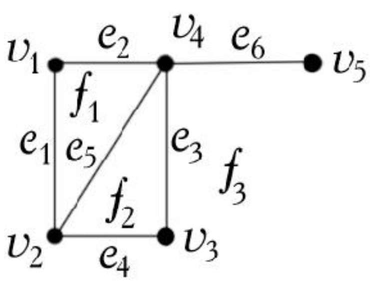
Параметры нового графа $n=4,\ m=5,\ f=3$. Следовательно,

$$n+f=4+3=7\ ,\ m+2=5+2=7\,.$$

6. Если к связному графу

$V=\{v_1,v_2,v_3,v_4\},\ E=\{e_1,e_2,e_3,e_4,e_5\}\ F=\{f_1,f_2,f_3\}$

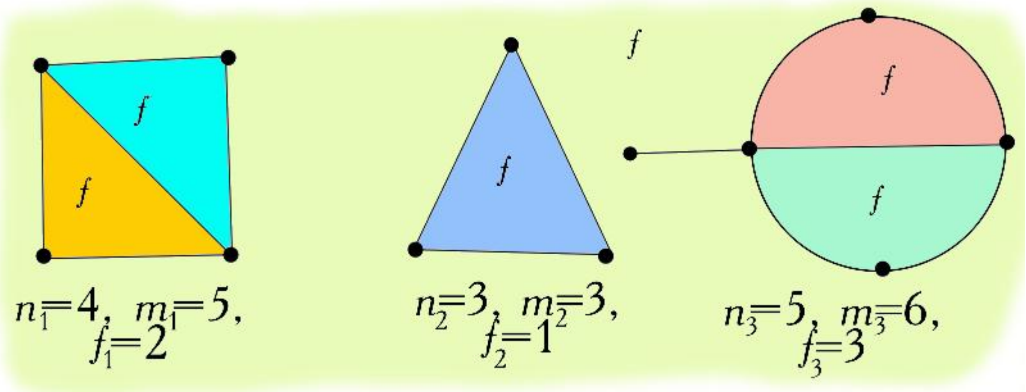
прибавить ребро, инцидентное только с одной из его вершин, получим граф
$V=\{v_1,v_2,v_3,v_4,v_5\},\ E=\{e_1,e_2,e_3,e_4,e_5,e_6\}\ F=\{f_1,f_2,f_3\}$



Получаем $n+f=5+3=8,\ m+2=6+2=8$.

Следствие. Пусть G - плоский граф с n вершинами, m ребрами, f гранями и k компонентами связности; тогда

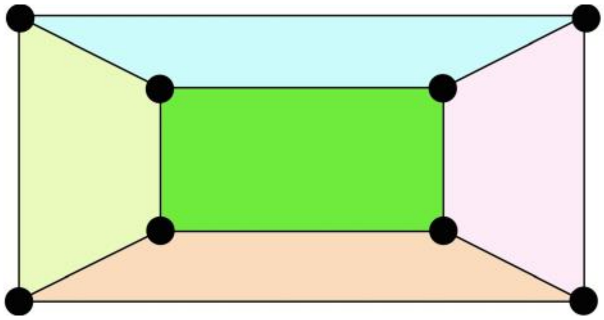
$$n+f=m+k+1\,.$$



$$n=n_1+n_2+n_3=4+3+5=12$$

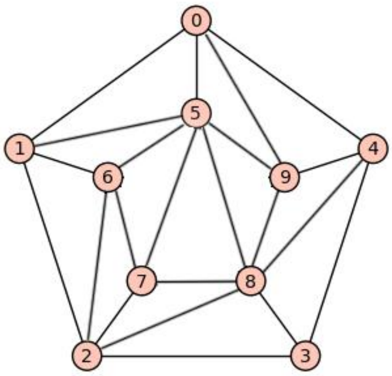
$$m = m_1 + m_2 + m_3 = 5 + 3 + 6 = 14$$
$$f = f_1 + f_2 + f_3 = 2 + 1 + 3 = 6$$
$$n + f = 12 + 6 = 18$$
$$m + k + 1 = 14 + 3 + 1 = 18$$

Следствие. Если G - связный планарный граф с m ребрами и $n \geq 3$ вершинами, то $m \leq 3n - 6$.

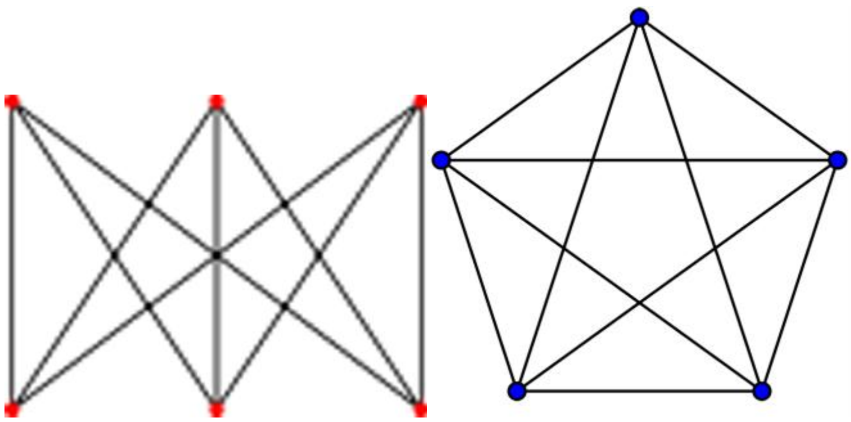


$$n = 8 > 3 \quad m = 12$$
$$m < 3n - 6, \; m < 3 \cdot 8 - 6, \quad m < 24 - 6, \; 12 < 18$$

Следствие. В любом планарном графе существует вершина, степень которой не больше пяти.



Утверждение. Всякий подграф планарного графа тоже планарный.
Утверждение. Если граф включает непланарный подграф, то и сам граф непланарный.

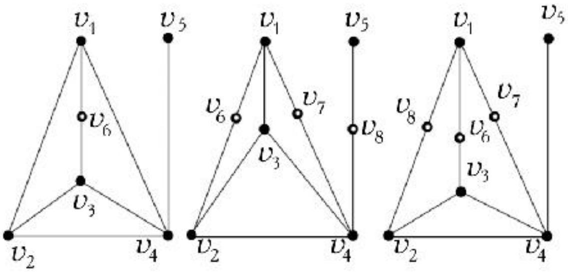


Следствие. Всякий подграф, включающий граф K_5 или $K_{3,3}$ не будет планарным.
Оказывается, что графы K_5 и $K_{3,3}$ – **единственные непланарные** графы в том смысле, что всякий непланарный граф включает один из них как подграф.

Для формулировки этого результата введем понятие *гомеоморфности графов.*

Гомеоморфные графы

Определение. Два графа называются *гомеоморфными* или *тождественными* с точностью до вершин степени 2, если они могут быть получены с одного и того же графа с помощью операции введения вершины степени 2 в ребро.
Например, графы, изображенные на рисунке гомеоморфны.

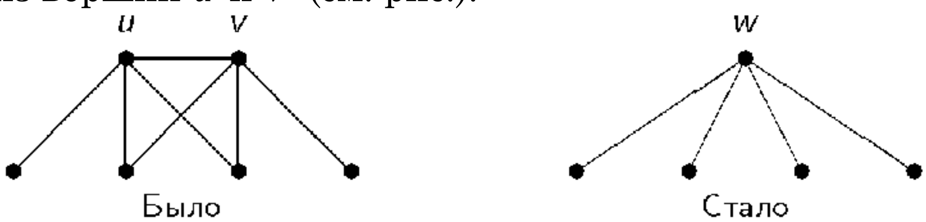


Нетрудно убедиться, что гомеоморфизм является отношением эквивалентности.
Теорема Понтрягина-Куратовского. Граф планарный тогда и только тогда, когда он не имеет подграфов, гомеоморфных K_5 и $K_{3,3}$.

Доказательство. С геометрической точки зрения, добавление вершины степени 2 — это добавление точки на ребре, а стирание такой вершины объединяет два ребра с общим концом в одно. Очевидно, что любая из этих операций, примененная к плоскому графу, снова даст плоский граф. Значит, по следствиям из теоремы Эйлера, никакой плоский (а, следовательно, и планарный) граф не гомеоморфен графам K_5 и $K_{3,3}$. С учетом замечания о непланарных подграфах, теорема доказана.

Операция стягивания

Пусть (u,v) — ребро графа G. Удалим из графа G вершины u и v. После этого добавим в граф G новую вершину w и соединим ее ребрами со всеми вершинами, с которыми была смежна хотя бы одна из вершин u и v (см. рис.).



Обозначим полученный граф через G'. Говорят, что граф G' получен из G *стягиванием ребра* (u,v).

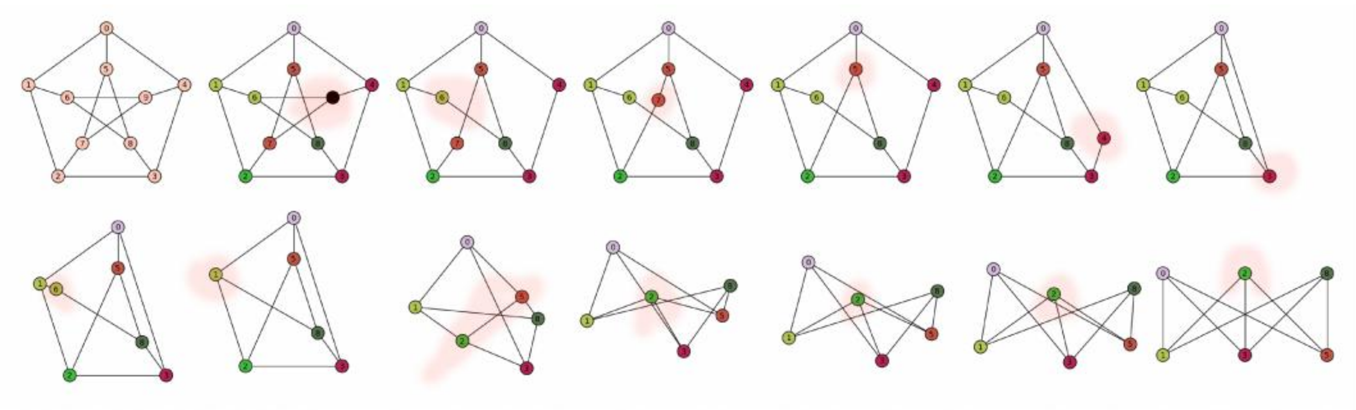
Если конечным (возможно нулевым) числом операций стягивания ребра из графа G можно получить граф G', то говорят, что G *стягивается* к G' .

Теорем Вагнера

Теорема. Граф планарный тогда и только тогда, когда он не имеет подграфов, которые стягиваются графом K_5 и $K_{3,3}$.

Доказательство необходимости. Если стянуть ребро в плоском графе, он, очевидно, останется плоским. Значит, по следствиям из теоремы Эйлера, никакой плоский (а следовательно, и пленарный) граф не стягивается ни к графу K_5, ни к графу $K_{3,3}$. С учетом замечания о непланарных подграфах, необходимость доказана.

Процесс стягивания к $K_{3,3}$



Процесс стягивания к K_5