300. Нека $G = \langle V, E \rangle$ е свързан мг и $w \in V$ е негов срязващ/критичен връх (след неговото отстраняване на w и ребрата, инцидентни с него, броят на свързаните компоненти на G се увеличава).
Нека G_1 е графът, получен от G чрез отстраняване на w и ребрата, инцидентни с него.
Докажете, че графът $\overline{G_1}$ е свързан.

Нека $C_1, \dots, C_k$ са свързаните компоненти на G_1.
Нека $u \in C_i$, $v \in C_j$ за $i \neq j$. $i, j \in \{1, \dots, k\}$
Тогава $\{u, v\} \notin E_1 \Rightarrow \{u, v\} \in \overline{E_1}$ и
$\{u, v\} \in E_1 \Rightarrow \{u, v\} \notin \overline{E_1}$ т.е.
всеки два върха от различни компоненти на свързаност в G_1

са свързани с ребро в $\overline{G_1}$.

Нека сега u и v са върхове от C_i, $i \in \{1,\dots,k\}$.

Ако $\{u,v\} \in E_1 \leadsto \{u,v\} \notin \overline{E_1}$ и $\{u,v\} \notin E_1 \leadsto \{u,v\} \in \overline{E_1}$.

Нека сега вземем $q \in C_j$ за $j \neq i$, $j \in \{1,\dots,k\}$.

Веднага имаме, че $\{u,q\} \in \overline{E_1}$ и $\{v,q\} \in \overline{E_1}$, т.е. имаме път м/у u и v през q в $\overline{G_1}$.

Т.е. току-що показахме, че $\overline{G_1}$ е свързан!

def | **Клика**

Пълен граф т.е. $G = \langle V, E \rangle$

и $|E| = \binom{n}{2} = \frac{n.(n-1)}{2}$

За произволен граф имаме, че

$0 \le |E| \le \binom{n}{2}$.

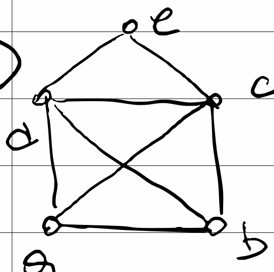
Антиклика е граф без ребра

т.е. няма два върха, които да са свързани с ребро $\rightarrow |E| = 0$.

def | **Кликово число**

Кликово число на граф е броят на върховете на най-голямата клика в графа!

Зад



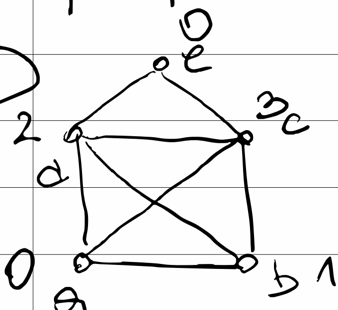
Какво е кликовото число на графа?

Една максимална клика е $\{a, b, c, d\} \rightarrow$ кликовото число е 4.

def | <u>Върхово хроматично число</u>

Това е най-малкият брой цветове необходими за оцветяване на върховете на граф, така че никои два едноцветни върха да не са свързани с ребро.

зад



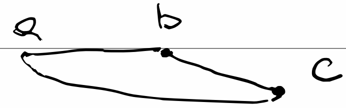
Какво е върховото хром. число на графа?

4 цвята са достатъчни, т.к. има клика с 4 върха. 3 цвята няма да стигат, т.к. заради кликата ще ни трябват поне 4. => 4 е върховото хром. число.

Th | Върховото хроматично число е по-голямо или равно на кликовото число.

(322) Постройте пример за th-мата, за който се получава ">".

C_3:



Върхово хр. число = 3

Кликово = 3

Още едно

C_5:

Върхово хр. число = 3

Кликово = 2

Става ☺.

цикъл с 5 върха

<u>Th на Кьониг!</u>

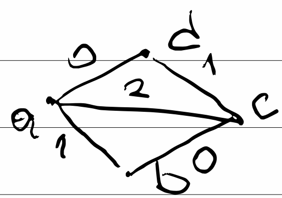
Един граф е двуцветен/двуделен $\longleftrightarrow$ графът не съдържа цикъл с <u>нечетна дължина</u>!

↳ т.е. с нечетен брой ребра

Да минем към оцветяване на ребра.

def| Реброво хром. число

Минимален брой цветове необходими за оцветяване на ребрата, така че никои две едноцветни ребра да нямат общ връх.

зад  Какво е ребровото хром. число на графа?

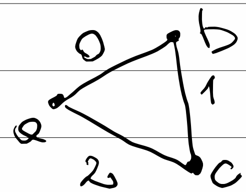
3 е, защото:

1) 3 е достатъчно.
2) 3 са необходими т.к. $d(a) = 3$

th| Ребровото хроматично число е $\geq \max\limits_{v \in V} d(v)$.

* В последната задача имахме "=".

(зад) Да построй граф, за който е изпълнено „<".

C_3: 

Реброво хром. число $= 3$, а $\max\limits_{v \in V} d(v) = 2$

(всички цикли с нечетни дължина стават)

(зад) Нека $G = \langle V, E \rangle$ е дърво.
Какви са:

a) кликовото число;

b) върховото хром. число;

c) Реброво хром. число?

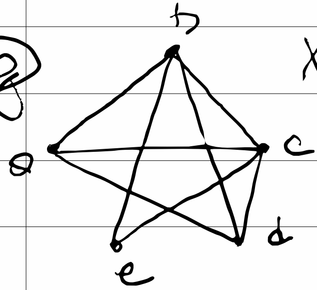
a) при $|V| = 1 \rightsquigarrow$ е 1
при $|V| > 1 \rightsquigarrow$ е $\underline{2}$ (ацикличен)

b) при $|V| = 1 \rightsquigarrow$ е 1
при $|V| > 1 \rightsquigarrow 2$ (в дървото няма цикли с нечетна дължина, значи по th Кьониг го получаваме, че е двуделен.)

c) Тъй като реброво хром. число е $= \underline{\max\limits_{v \in V} d(v)}$

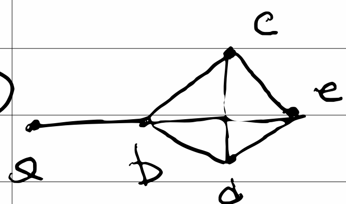
def 1. • Хамилтонов път – минаване точно веднъж през всеки връх на графа (първи ≠ последен).

• Хамилтонов цикъл – хамилтонов път, в който първи = последен.

302) 

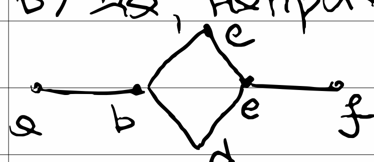
Хамилтонов ли е графа?

Да, един такъв цикъл е acebda.

303) 

a) Хамилтонов ли е?
b) Има ли хамил. път?

a) Не, защото има връх от степен 1.
b) Да, например abced.

304) 

a) Хамилтонов ли е?
b) Има ли хамил. път?

a) Не, защото както преди има върхове от степен 1.
b) Няма и хамилтонов път, защото

abcef → не съд. d
abdef → не съд. c } → няма

други възможности от това и
$d(a) = d(f) = 1$

Th на Dirac

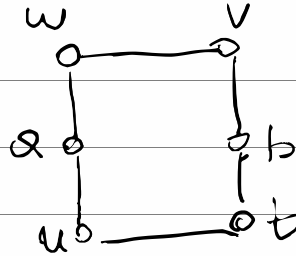
Ако $G = \langle V, E \rangle$ е граф, т.ч.
$n = |V| \geq 3$ и всеки връх $v \in V$, то
$d(v) \geq \frac{n}{2}$, то G е Хамилтонов

Th на Оре

Ако $G = \langle V, E \rangle$ е граф, т.ч.
$n = |V| \geq 2$ и за всяка двойка
несъседни върхове $u, v \in V$
е изпълнено, че $d(u) + d(v) \geq n$, то
G е Хамилтонов.

Зад Условието $d(u) + d(v) \geq n$ е
достатъчно, но не е необходимо,
за да бъде графа Хамилтонов.

прим 1



$d(u)+d(v) = 2+2 = 4 \not\geq 5 = n$

Също $(\forall v' \in V)\,[d(v') = 2 \not\geq 3]$, $3 = \frac{6}{2}$

Th на Dirac също не става

Но има Хамилтонов цикъл!

(Зад) На прием били поканени 20 гости. Всеки от тях познава поне 10 от другите. Д-же, че можем да ги наредим на масата, така че всеки да си познава двамата съседи

Разгл. неориент. граф
- V – гостите
- E – познанството м/у гостите

$d(u) \geq 10$, $d(v) \geq 10$, $n = 20$,

т.е. $d(u) + d(v) \geq 20$ за кои да е гости $u, v \in V$ по условие (в частност и ако u и v са непознати един за друг).

По Th на Ore, значи графът е Хамилтонов, с други думи можем да ги наредим на масата.

def | Път – не може да се повтарят върхове и ребра.
Верига – не може да се повтаря ребро, но може да се повтарят върхове.

def | **Ойлерови вериги**
Минават през всяко ребро точно веднъж. Има два вида: отворени и затворени.

Th | Нека G е свързан мултиграф. Ако броят на върховете от нечетна степен е:

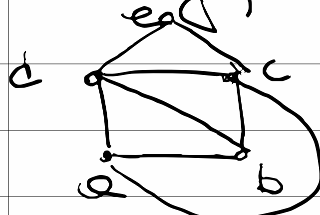
a) 0, то има затворена Ойлерова верига и не същ. отворена.

b) 2, то същ. отворена Ойлерова верига с краища двата върха от нечетна степен и не същ. затворена

c) ≠ 0 и 2, то не същ. нито отворена, нито затворена Ойлерова верига.

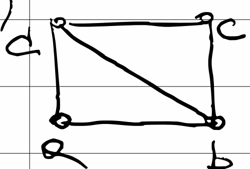
392 Има ли Ойлерова верига? Ако да, то каква е тя?

a)



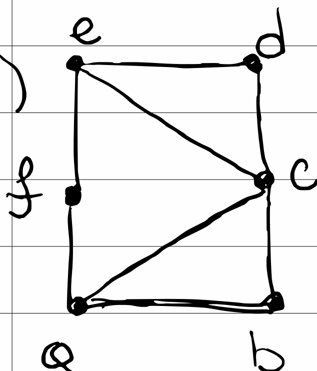
Има 2 върха от неч. степен $d(a)=d(b)=3$. Има отворена: adecdbacb.

b)



Има 4 на брой от нечетна степен => не съществува нито отворена, нито затворена

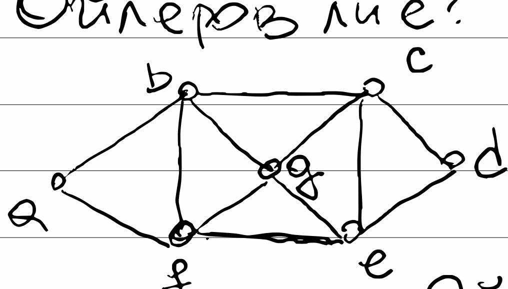
c)



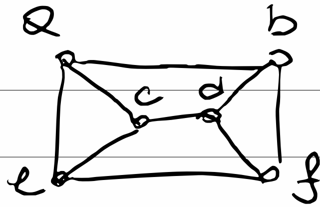
Няма върхове от нечетна степен → има затворена: abcdefaeca.

393 Какви са следните графи?

a) Ойлеров ли е?

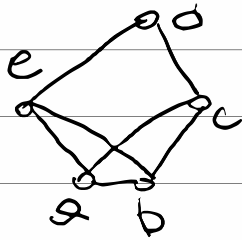


Точно два върха с нечетна степен $d(a)=d(b)$. → Ойлеров е.

b) 

Ойлеров или Хамилтонов?

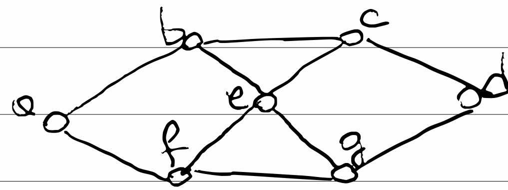
Не е Ойлеров т.к. имаме повече от 2 върха от нечетна степен.
Съгласно th на Dirac с $n = 6$ и $(\forall v \in V)[d(v) = 3 = n/2]$, значи G е Хамилтонов.

c) 

Ойлеров или Хамилтонов?

Не е Ойлеров: повече от два върха от нечет. степен.
По Th на Ore и т.к. $n=5$ и $d(a)=d(b)=d(c)=d(e)=3$ и $d(d)=2$, то имаме, че за кои да е върха от графа, сумата на степените им е поне 5.
Значи е Хамилтонов.

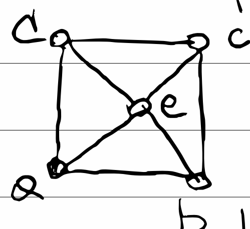
d) Ойлеров и Хамилтонов?



Не е Ойлеров. <u>Хамилтонов е</u>:

abcdgefa

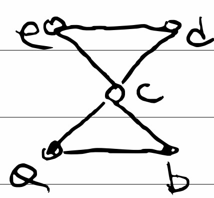
e) Хамилтонов, но не е Ойлеров



Не е Ойлеров, но е Хамилтононов по <u>Th на Ore</u>.

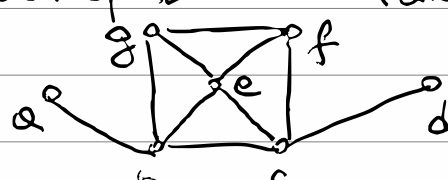
$|V|=5$, $d(a)=d(b)=d(c)=d(d)=3$, $d(e)=4$, $(\forall v\in V)(\forall v'\in V)[d(v)+d(v')\geq 6>\underline{5}]$

f) Ойлеров, но не Хамилтонов.



Ойлеров е, но не е Хамилтонов, защото c е срязващ връх.

g) Не Ойлеров и не Хамилтонов



Не е Ойлеров и не е Хамилтонов.

Покриващи дървета (ST)
(Минимални/максимални покриващи дървета OST)

def Покриващо дърво

$T = \langle V, \mathcal{E}' \rangle$ е покриващо дърво на $G = \langle V, \mathcal{E} \rangle$, ако:

- $\mathcal{E}' \subseteq \mathcal{E}$
- G е свързан (иначе T е гора);
- T е дърво.

Ако графът има тегло по ребрата, можем да говорим за optimal spanning tree (OST) било то максимални или минимално.
Минимално покриващо дърво съдържа всички върхове на даден свързан тегловен неориент. граф и сбора от теглата на ребрата е минимален.

Основни алгоритми строящи покриващи дървета на НГ/ОГ.
- bfs – в ширина
- dfs – в дълбочина

Имаме теорема за НГ:

Th| Всяко кореново дърво е дърво както и:

Th| Графът G има покриващо дърво <=> G е свързан

bfs и dfs обхождат граф и строят покриващо дърво. С тях може да проверявате дали е свързан граф.
С dfs е удобно да проверите дали граф съдържа цикъл.
Bfs строи дърво на най-

късите пътища (когато G няма тегло по ребрата) от един начален връх до всички останали.

BFS (G, s)

for all v ∈ V:

масив - пази разстояние от s до останалите

dist(v) = ∞

масив пази покриващото дърво

parent(v) = null

dist(s) = 0 // тривиален път от s до s

queue = initQueue(s) // в опашката са сиви върхове, а извън нея са или бели, или черни. Белите имат dist = ∞, а черните dist < ∞ /

while queue is not empty:

v = dequeue(queue) // вземане главата на опашката

for all edges {v, u} ∈ E:

if dist(u) = ∞: // бял връх

enqueue(queue, u)

dist(u) = dist(v) + 1

parent(u) = v

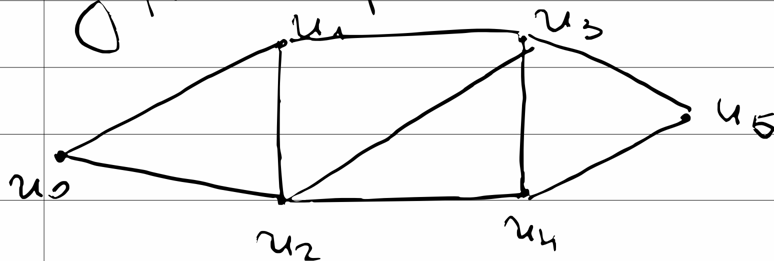
$O(|V|+|E|)$ сложност по време.

```
DFS(G)
  forall v ∈ V:
    visited(v) = false   // само дели в началото
    parent(v) = null
  forall v ∈ V
    if not visited(v):
      explore(G, v)

explore(G, v)
  visited(v) = true   // сив връх е v
  for all edges {v, u} ∈ E:
    if not visited(u):
      parent(u) = v
      explore(u)
  // вече тук v е черен.
```

Пак $O(|V|+|E|)$ сложност по време. Всъщност такова е и сложността по памет.

(322) Да се намери покриващо дърво на графа:



a) с bfs;

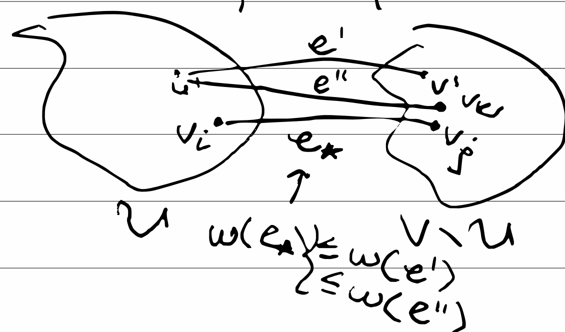
b) с dfs.

Optimal spanning tree

```
kruskal(G, w):
    E_T = {}
    Sort edges E by inc. weight using w.
    for all edges {u,v} ∈ E:
        if adding {u,v} to E_T does not
                form a cycle:
            add edge to E_T
    return E_T
```

Най-проста имплементация.
Опозвава се на th за съгласуваното мн-во.

$\boxed{\text{Тм}}$ Нека $G=\langle V, E\rangle$ е НГ, свързан с ценова ф-я на ребрата $w: E \to \mathbb{R}$ и $\emptyset \neq U \subsetneq V$. Нека $e=\{v_i, v_j\} \in E$ е т.е. $v_i \in U$, $v_j \notin U$ и има мин. тегло измежду всички такива ребра. Тогава съществува MST $T_0 = \langle V, E_0\rangle$, т.е. $e \in E_0$.



```
Prim-Jarnik(G, w, s)
    forall v ∈ V
        cost(v) = ∞
        parent(v) = null
    cost(s) = 0
    PQ = makePQ(V, cost)  // строи приор. опашка спрямо cost
    while PQ is not empty:
        u = popMin(PQ)  // връща върха с най-малък ключ
```

намерили сме връх, за който можем да подобрим цената му

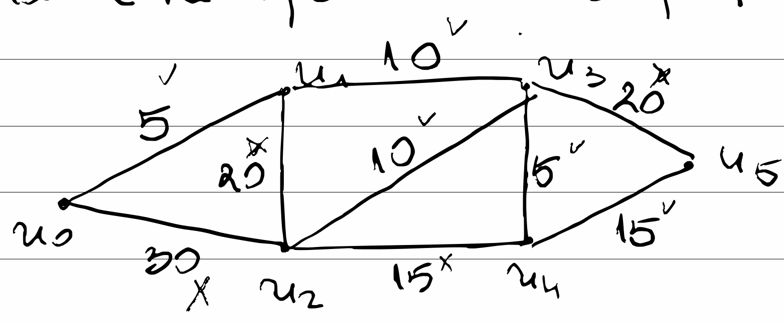
```
for each {u, v} ∈ E:
    if cost(v) > w(u, v):
        cost(v) = w(u, v)
        parent(v) = u
        decreaseKey(PQ, v)
```

изпраща сигнал към опашката, че cost ст-та на v е била намалена и предизвиква пренареждане на PQ спрямо обновения масив cost.

relax(u, v, w, cost, parent, PQ)

---

392 Да се намери MST на графа.



a) с Kruskal:

$u_0u_1 - 5$ ✓

$u_3u_5 - 5$ ✓

$u_1u_3 - 10$ ✓

$u_2u_3 - 10$ ✓

$\cancel{u_2u_4 - 15}$ → става цикъл

$u_4u_5 - 15$ ✓

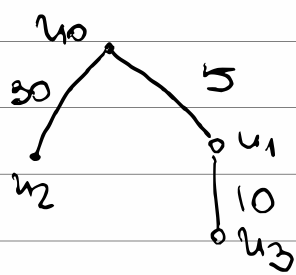
$u_3u_5 - 20$

$u_1u_2 - 20$

$u_0u_2 - 30$

min тегло = 45

b) Прим-Ярник:



SPST

Дървета на най-късите пътища

* Ако няма тегла на ребрата, казахме, че bfs строй от даден връх до всички други.
* Ако има положителни тегла и
  * всички тегла са равни -> bfs;
  * има поне две различни -> Dijkstra.
* Ако графът е DAG (directed acyclyc graph), има по-хубави алгоритми.
* Ако сред теглата има отрицателни, има други алгоритми, които строят SPT.
* Dijkstra строи дърво на най-късите пътища от даден връх, до всички други (работи за ЧОГ и КНГ).

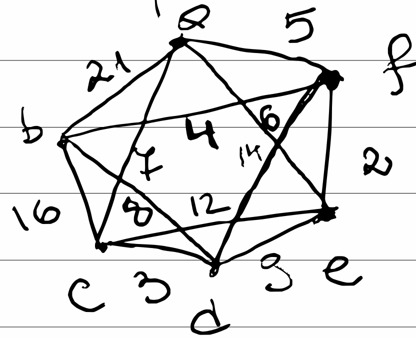
! MST ≠ SPST !

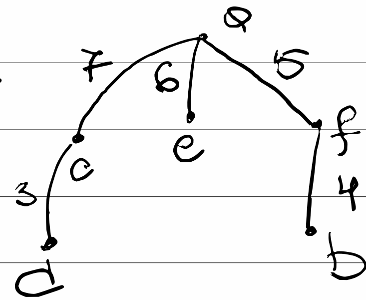
```
Dijkstra(G, w, s)
  for all v ∈ V:
      dist(v) = ∞
      parent(v) = null
  dist(s) = 0  // тривиален път от s до s
  PQ = makePQ(V, dist)
  while PQ is not empty:
      u = deleteMin(PQ)
      for all edges {u,v} ∈ E:
          if dist(v) > dist(u) + w(u,v):
              dist(v) = dist(u) + w(u,v)
              parent(v) = u
              decreasekey(PQ, v)
```

взимаме върха, до който стигаме най-бързо

намерили сме по-лек път до v през u

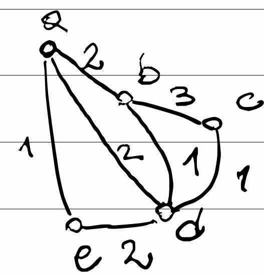
Тв | Всеки начален отрез на най-лек път от s строен от Dijkstra е също най-лек път.

(зад) Постройте MST на  с корена a.



(зад) КНГ $G = \langle V, R \rangle$ с м-во върхове $V = \{a, b, c, d, e\}$ е зададен с матрица M:

$$M = \begin{array}{|c|c|c|c|c|} \infty & 2 & \infty & 2 & 1 \\ \hline 2 & \infty & 3 & 1 & \infty \\ \hline \infty & 3 & \infty & 1 & \infty \\ \hline 2 & 1 & 1 & \infty & 2 \\ \hline 1 & \infty & \infty & 2 & \infty \end{array}$$



с елементи $m_{ij} = \begin{cases} \text{"}\in N\text{"}, & \text{ако } \langle i,j \rangle \in R \\ \infty, & \text{ако } \langle i,j \rangle \notin R \end{cases}$

a) По Kruskal MST без корен;

b) По Prim-Jarnick MST с корен r=3.

c) SPST с Dijkstra с корен r=1

d) SPST с корен r=3, ако теглата на всички ребра е 1.