

$$f: \mathbb{N} \rightarrow \mathbb{R}$$

$f \in O(g) \Leftrightarrow \exists M > 0, \exists N \in \mathbb{N}, \forall n > N, f(n) \leq M g(n)$

$$\min_{n \geq 1} n^3 \approx n^{1.582^2} \approx n$$

$$\lim_{x \rightarrow \infty} \frac{f(x)}{g(x)} = 0 \Rightarrow f \in O(g)$$

$$\forall \varepsilon > 0, \exists N > 0, x > N \Rightarrow \left| \frac{f(x)}{g(x)} \right| < \varepsilon$$

$$\Rightarrow |f(x)| < \varepsilon |g(x)|$$

potom je

$$\begin{array}{c} \exists \varepsilon' > 0, \exists N, \forall n > N, \\ \parallel \\ f(n) \leq \varepsilon' g(n) \end{array} \Rightarrow f \in O(g)$$

$$O(1) \subset O(\ln(\ln(n)))$$

$$O(\log_2(n)) = O(\underbrace{\log_{10}(n)})$$

$$\log_2(n) = \frac{\log_{10} n}{\log_{10} 2} \in O(\log_{10}(n))$$

$$\log_{10}(n) = \frac{\log_2 n}{\log_2 10} \in O(\log_2(n))$$

koleme

- že celo podjetje (producent po kolosijenjih)
- zdej je po kolece řešit na fázy
- seznám se s obecnou kritikou

$$\begin{aligned} O(1) &\subset O(\log(\log n)) \subset O(\log_2(n)) = O(\log_{10}(n)) \\ &\subset O(\sqrt{n}) \subset O(\log(n!)) \subset O(n \log n) \subset O(3^{n^n}) \end{aligned}$$

$$O(\log_2(n)) \subset O(n^\varepsilon) \quad \begin{matrix} \subseteq O(n^2) \subseteq O(2^n) \\ \vdots \\ \subseteq (2^{\log n}) \end{matrix}$$

$$\log(n!) \leq \log(n^n) = n \log n$$

$$O(\log(\log(n))) \leq O(\log n)$$

$$\log(\log(n)) \leq \log(n) \quad \begin{matrix} \log(n) < n \\ \text{in log} \\ \text{je nezávazné} \end{matrix}$$

$$\log(n) \in O(\sqrt{n})$$

$$\lim_{n \rightarrow \infty} \frac{\log n}{\sqrt{n}} \stackrel{H}{=} \frac{\frac{1}{n} \cdot \frac{1}{2}}{n^{\frac{1}{2}}} = \frac{1}{2\sqrt{n^3}} = 0 \Rightarrow \log(n) \in O(\sqrt{n})$$

$$3^{\log(n)} = 3^{\frac{\log_2(n)}{\log_2 3}} = n^{\frac{1}{\log_2 3}} = n^{\varepsilon} > 1$$

$$\sqrt{n} < n < \log n$$

$$\mathcal{O}(n \log n) = \mathcal{O}(\log n!)$$

$$\log n! = \sum_{i=1}^n \log i$$

$$\log n! > \log\left(\frac{n}{e}\right)^{\frac{n}{2}} = \frac{n}{2}(\log n - \log 2) \approx n(\log n - \frac{1}{2})$$

$$\underline{n \log n} < \underline{n^2}$$

$$n \log n \leq n \cdot n \leq n^2$$

$$\mathcal{O}(n^2) \subseteq \mathcal{O}(2^n) \subseteq \mathcal{O}(2^{n \log n} \leq \mathcal{O}(n!))$$

$$2^{\log_2 n} = 2^{\frac{\log_2 n}{\log_2 10}} = n^{\log_2 10}$$

seznam: n

append: $O(1)^*$

delete(i): $O(n)$ ($n - i$)

copy: $O(n)$

update(i): $O(1)$

get(i): $O(1)$

find(a): $O(n)$

add: $O(n)$

tipične zemlje	čas
add(a)	$O(n)$
add(O, n)	$O(n)O(1)$
delete();	$O(n)$
delete(a, n)	$O(n), O(1)$
get();	$O(1)$
search(x)	$O(n)$

tipične zemlje	čas	prostov
add(a)	$O(n)$	$O(1)$
add(O, n)	$O(n)O(1)$	$O(1)$
delete();	$O(n)$	$O(1)$
delete(a, n)	$O(n), O(1)$	$O(1)$
get();	$O(1)$	$O(1)$
search(x)	$O(n)$	$O(1)$

Slovnik
 $O(1)$ vse

algorithm za iskanje maxima

$m = \text{seznam}[0]$

for c in $\text{seznam}[1:]$
if $c > m \Rightarrow m = c$

$O(n), O(1)$

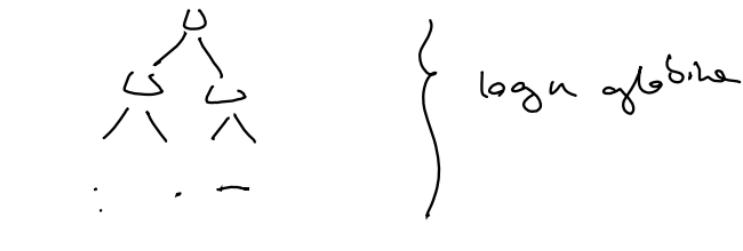
return m

seznam^n
 $\text{seznam}.sort$
 $\text{seznam}[-1]$

$O(n \log n)$ $O(1)$

$$\sum_{i=1}^n i = O(n^2)$$

rekurzivno: seznam na dve deli



↳ ↳ ↳ ↳ ↳

čas je lahko $O(\log n) \approx \frac{n}{\log n}$ (prostovanje n
koda se je tudi $O(\frac{n}{\log n})$ preberi inje
se vedno hkrati)

Bitwise operacije

- bitand &
- bitor |
- bitxor ^
- bitnot !, ~
- bitshiftleft <<
- bitshiftright >>

v jazyku
 &
 ||
 !=
 not

$$z: 111$$

$$g: 1001$$

$$\begin{array}{r} 111 \\ 1001 \\ \hline 1 \end{array} \&$$

$$\begin{array}{r} 0111 \\ 1001 \\ \hline 1111 \end{array} |$$

$$\wedge \begin{array}{r} 111 \\ 1001 \\ \hline 1110 \end{array}$$

$$! \begin{array}{r} 111 \\ 0 \\ \hline \end{array} ! \begin{array}{r} 1001 \\ 110 \\ \hline \end{array}$$

$$<< \begin{array}{r} 1001 \\ 10010 \\ \hline \end{array} \quad g \gg 2 \begin{array}{r} 1001 \\ 10 \\ \hline \end{array} :$$

$$\underline{a \oplus b} = \underline{a}$$

a	b	\oplus	c
0	0	0	1
0	1	1	0
1	1	0	1

a...b, i

$$\& \frac{a_n \dots a_2 a_1 a_0}{[0 \dots 010 \dots 0 \dots 0]} \leftarrow \text{mask}$$

if $\underbrace{(a \& \text{mask})}_{>0 \text{ zero}} \text{ True}$

mask = $1 \leq i$

$$\begin{array}{l} 1.) [0 \dots 0] \\ \sim [1 \dots 1] \\ \gg n-i [0 \dots 1 \dots 1] \end{array} \quad \begin{array}{l} \sim((\sim 0) \ll i) \\ (1 \sim i)-1 \end{array}$$

funkcija spremi a, i

$$a_n \dots a_0 \rightarrow a_n \dots ^1 \dots a_1 a_0$$

def f(a, i):

 mask := $1 \ll i$

 return $a \mid mask$

def f'(a, i)

 mask := $\sim(1 \ll i)$

 return $a \& mask$

	uuu	
uuu	uuu	uuu
	uuu	
uuu	uuu	uuu
	uuu	

$$n = 0010;1011;0\underline{00}0;0110;1010 \quad \begin{array}{l} \swarrow \text{no mask} \\ \searrow \end{array}$$

leva: $n-1$

dava: $n+1$

gav: $n-u$

del: $n+u$

def levo_prij(a)

 maskp = $(1 \ll u) - 1$

 p = $a \& maskp$

 mask = $a(1 \ll 20 - p_{10} + 1)$

 return $(a \& mask) - 1$

$\left. \begin{array}{l} a = a \& mask \\ a = (a \gg u) \ll u \end{array} \right\}$

 return $a \mid p$

def fib(n):

prv = 1

drv = 1

for i in range(n)

 drv = drv'

 drv += prv

 prv = drv'

return drv

$$\sum_{i=1}^n 3O(i) = O(n^2)$$

$$O(drv) = 1,6^i = 1,6^n$$

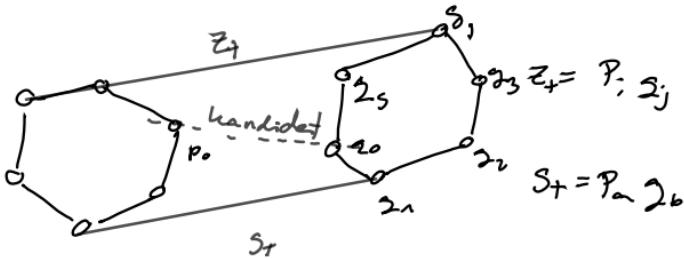
at least

$$\text{at bitav za } drv = \log_2 1,6^i \\ = O(i)$$

assign = $O(n)$

TRIE - prefix tree

V
españa



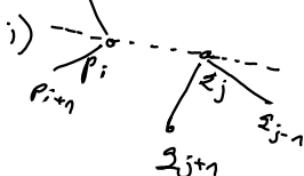
Združevava vrednost bo rešitev

Kako do z_j

$$z_j = p_0 z_0$$

while z_j ni rezanja tangent

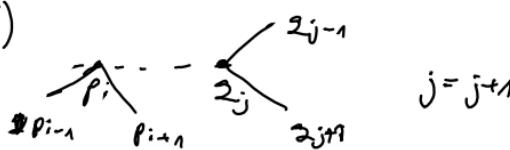
$$p_{i-1}$$



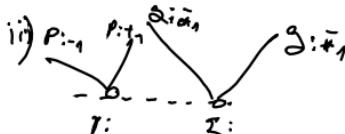
V tem primeru nadavimo

$$p_i = p_{i-1} \quad (i = i-1)$$

ii)



$$j = j+1$$



to je spodnja tangent
divergencija oba
 $i+1$ $j+1$

Bla bla bla

čr posicija n_i - - - heredis $+1$

Predstava grafa

- Matrica sosednjosti $A = [a_{ij}]$; $a_{ij} = \begin{cases} 1 & i \text{ je sosed} \\ 0 & \text{sicer} \end{cases}$
- slavar $\{ \text{vozlišče : } [\text{povezave do vozlišča}] \}$
 $\{ v : [u \text{ for } u \text{ in next}(v)] \text{ for } v \text{ in } G \}$

opomba: ce vozlišče označimo od 0 do $n-1$, namesto
sloverja uporabimo seznam

ideja DFS:

- ~~1. izberemo začetno vozlišče in ga shranimo v obiskane~~
~~2. izberemo sosede, ki se niso v obiskanih~~
~~3. ponavljamo se vsakega~~

ideja BFS:

1. izberemo začetno vozlišče in ga denu v obiskane
2. obiskano vozlišče sosedne v njih denju obiskane,
ter naredimo seznam vseh sosedov v sosedov
3. obiskano vse sosedove ... stokat 2.

from collections import deque ← double ended que

def DFS(G, s):

n = len(G)

obiskeni = [False]*n

sklad = deque([s])

while sklad:

v = sklad.pop()

if not obiskeni[v] ← standardne struktura
for u in G[v]:

if obiskeni[u] == False:

sklad.append(u)
obiskeni[v] = True

def BFS(G, s):

n = len(G)

obiskeni = [False]*n

rrsta = deque([s])

while rrsta:

v = rrsta.popleft()

if not obiskeni[v]
for u in G[v]:

if obiskeni[u] == False:

rrsta.append(u)
obiskeni[v] = True

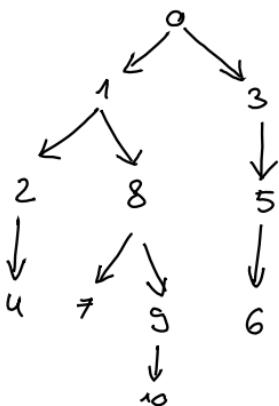
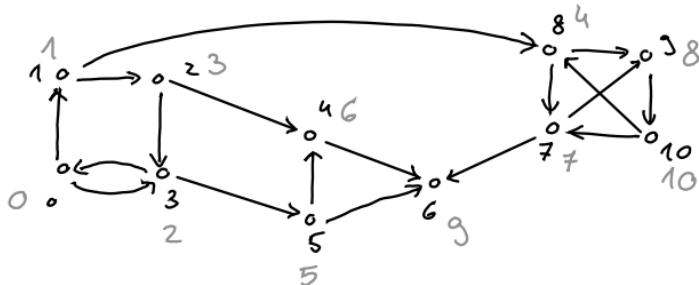
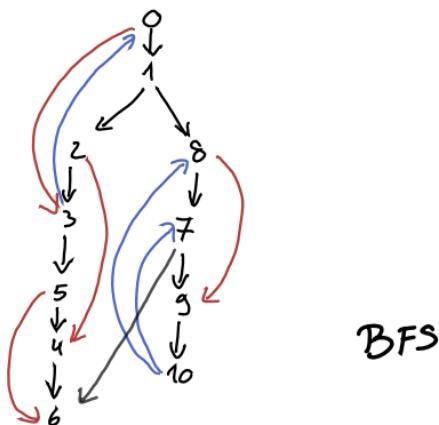
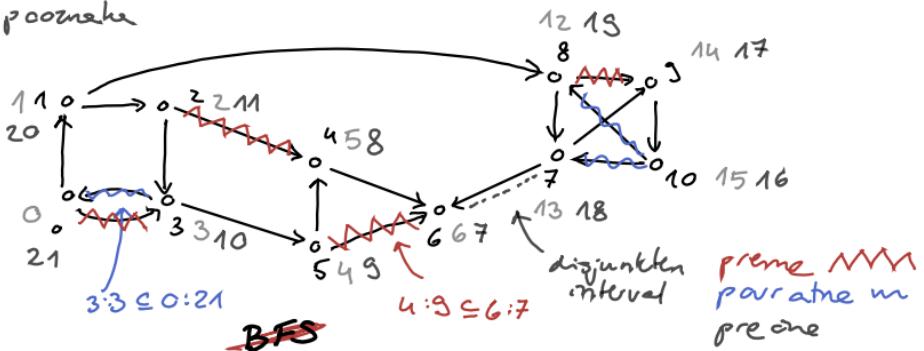
O(1) add(x) (denad)

O(1) pop(x) (vzam: desm)

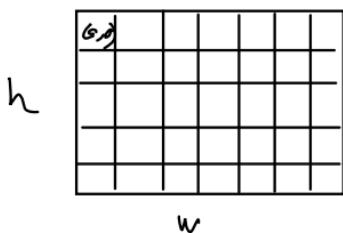
O(1) popleft(x) (vzam: lewo)

-Predomäne
-pooromäne

DFS



Razvij pristop za delo zgrabi, ki so mreže



$$\# \text{vred} \cdot \bar{s} = h \cdot w \quad (\text{celice})$$

u -sosednost

$$(i,j) \sim (i \pm 1, j)$$

$$(i,j) \sim (i, \pm j)$$

$$\text{comeri} := [(1,0), (-1,0), (0,1), (0,-1)]$$

def get_sosed(i,j) :

return [(i+a, j+b) for (a,b) in comeri]

G ne usmerjen graf

Algoritam da ē graf modeln, ne sicer

$V(G) = A \cup B$ - biparticija na dodeli, tako da
med



ko katerih pa barvamo $V(G)$ tako, da vokišča vedno
dobjo drug barvo.



from collections import deque

def je_dodeli(G)

n = len(G)

barva = [None] * 2 \leftarrow = obiskani

vrsta = deque([0])

barva[0] = 0

while not vrsta.empty:

u = vrsta.popleft()

trb = barva[u]

for v in G[u]

if barva[v] == None

vrsta.push(v)

barva[v] = trb $\% 2$

elif barva[v] == trb

return False

return True

def pot_do_najk(G)

n = len(G)

razdelje = [None]*n

p-ecd = [None]*n

razdelje[0] = 0

rrsta.add(0)

while not rrsta.empty

3 Mreža 4×4 s rdećim i matim poljima prema

	M	m	m
M	M	m	m
M	M	m	m
M	M	m	m

m	m	m	m
M	M	m	m
M		M	m
M	M	m	m

Zatim ne najmanje dve pozicije iz A v B
pojavljuju se u poziciji

Vsaka konfiguracija je svoje vrste

pozicija
boje
u biti

rdeće in
matre
16 bitov

Sosed: razen ne rabi
pozicija boje ($\pm 1 \vee \pm 4$) A parsal ista kren
razen v pozicijah boje, tada parsel
biti ista boje

V 28. 11

Wen: pomembro

Dijkstruv algoritam

Vred: G usmerjen/neusmerjen graf.

povezno učenje, učenje so nenegativne,
 $w: E(G) \rightarrow \mathbb{R}_+$

za četno vozlišče s (oprejša skupina vozlišč)

izhod \Rightarrow razdelje nekajje podit od s od vseh ostalih vozlišč

- začnemo v s $d(s)=0$
- Pogledamo sosede od s in izberemo tistega, ki je "naj blžji" $d(u)=w(u)$
- ostale sosede damo ne prioriteta vrsto v obliki $(d(s)+w(v), v)$

- meto iz vrste

\leftarrow za drugiško kopico
 $\log(E)$ $\log(E)$
 $O(|V| + |E| + |E|(\text{zah temost delovanja} + \text{zah temost brisanja}))$

\nearrow
 katerih je
 ne prioritetskih

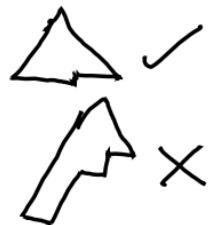
$O(1)$ $O(E)$
 \nearrow \nearrow
 PQ seznam

$$= O(|V| + |E| + E(\log E)) = O(E \log E + V)$$

Lahko tudi $O(V^2)$

Dvojiska kopica

- levo paravzano dvojisko drevo
- lastnost kopice: vrednost vsakega starša \leq min vrednosti oben otrok



V korenu je najmanjši element, ampak to n: izkcelno drevo

Osnovne operacije

dodaj (x)

brisi (izbrisemimum)

dodej;

dodemo ga na zadnje mesto in dvigujemo ga, dokler ne zedajo lastnost; kopice

brisi:

vzamemo koren in ga izbrisemo.

Najbolj spodnji desni element prestavimo v koren in potem ga bubljamo dol

