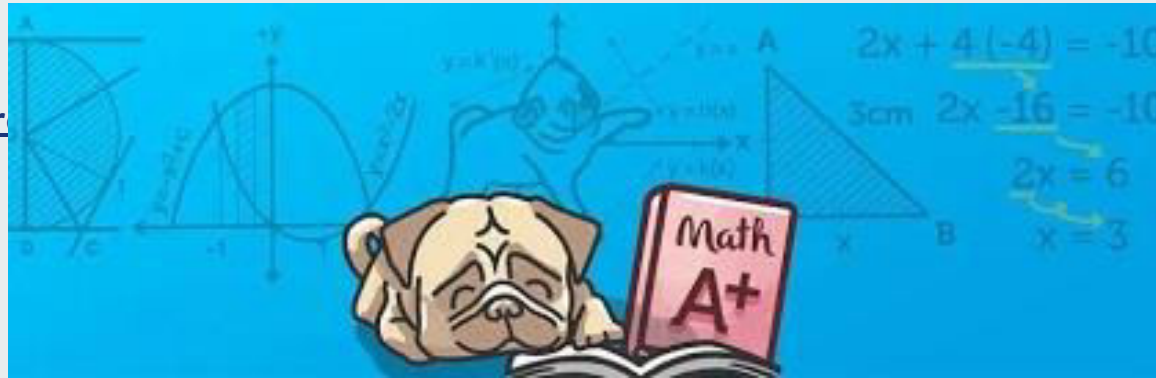
Algoritmi Avansați 2021
c-7
Debriefing
Randomized Data Structures: Skip Lists

Lect. Dr. Ștefan Popescu

Email: stefan.popescu@fmi.unibuc.r

Grup Teams:



Debriefing

- Smalltalk about second half of the course
- presentation schedule for Tema 1 & Tema 2
- future lab work
- smalltalk about final exam

Randomized Data Structures

- O privire pe scurt, “din avion” asupra Skip lists.

Randomized Data Structures

- O privire pe scurt, “din avion” asupra Skip lists.

Introduse in 1989 de către W. Purgh, sunt structuri dinamice bazate pe factor aleator (randomized)

Randomized Data Structures

- O privire pe scurt, “din avion” asupra Skip lists.

Introduse in 1989 de către W. Purgh, sunt structuri dinamice bazate pe factor aleator (randomized)

Skip lists are a probabilistic data structure that seem likely to supplant balanced trees as the implementation method of choice for many applications. Skip list algorithms have the same asymptotic expected time bounds as balanced trees and are simpler, faster and use less space.

— William Pugh, Concurrent Maintenance of Skip Lists (1989)

Listele “standard”

Listele “standard”

Q: Complexitatea cautarii unui element intr-o lista sortata?

Listele “standard”

Q: Complexitatea cautarii unui element intr-o lista sortata?

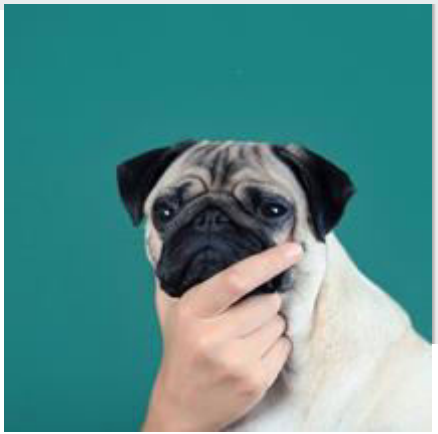
A: O(n)

Listele “standard”

Q: Complexitatea cautarii unui element intr-o lista sortata?

A: O(n)

Q: Suntem multumiti?



Listele “standard”

Q: Complexitatea cautarii unui element intr-o lista sortata?

A: O(n)

Q: Suntem multumiti?

A: NU!

Listele “standard”

Q: Complexitatea cautarii unui element intr-o lista sortata?

A: O(n)

Q: Complexitate țintă?

Listele “standard”

Q: Complexitatea cautarii unui element intr-o lista sortata?

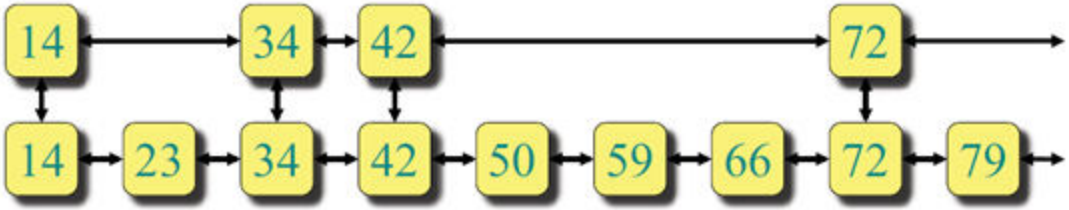
A: O(n)

Q: Complexitate țintă?

A: O(log n)

Skip Lists

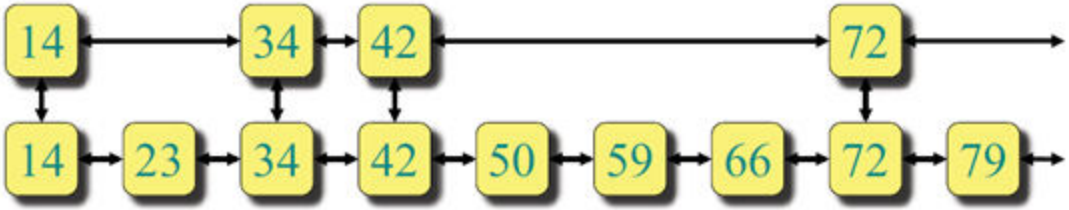
Pentru o cautare mai eficienta, vom retine 2 liste, dupa modelul urmator:



Skip Lists

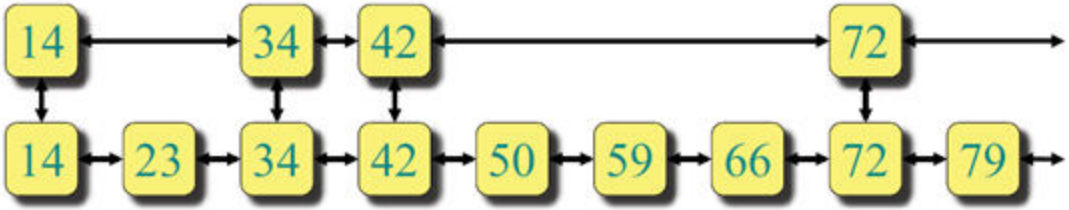
SEARCH(x):

- Walk right in top linked list (L1) until going right would go too far
- Walk down to bottom linked list (L2)
- Walk right in L2 until element found (or not)



Skip Lists

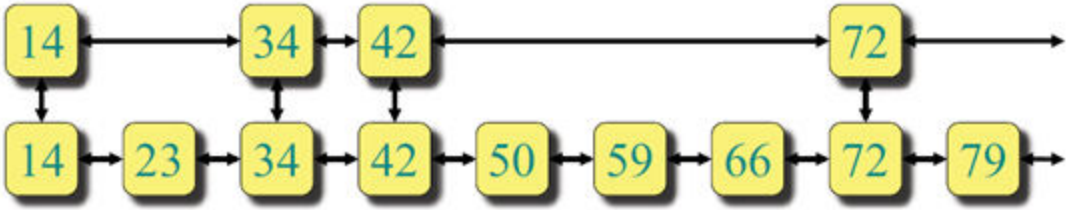
Q: Cum ar trebui distribuite nodurile din L1 (nivelul de mai sus) astfel incat cautarea sa fie cat mai eficienta?



Skip Lists

Q: Cum ar trebui distribuite nodurile din L1 (nivelul de mai sus) astfel incat cautarea sa fie cat mai eficienta?

A: Uniform distribuit!

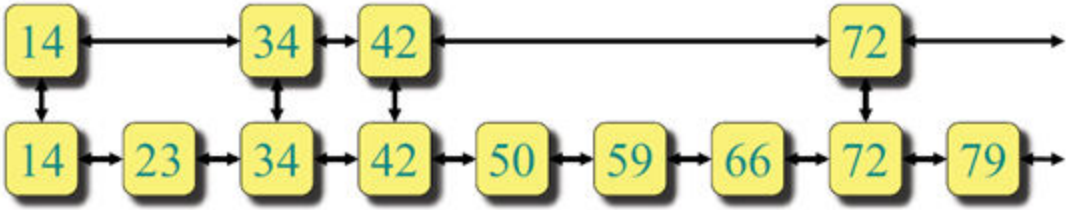


Skip Lists

Q: Cum ar trebui distribuite nodurile din L1 (nivelul de mai sus) astfel incat cautarea sa fie cat mai eficienta?

A: Uniform distribuit!

Q: cate noduri ar trebui sa existe in L1?



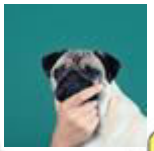
Skip Lists

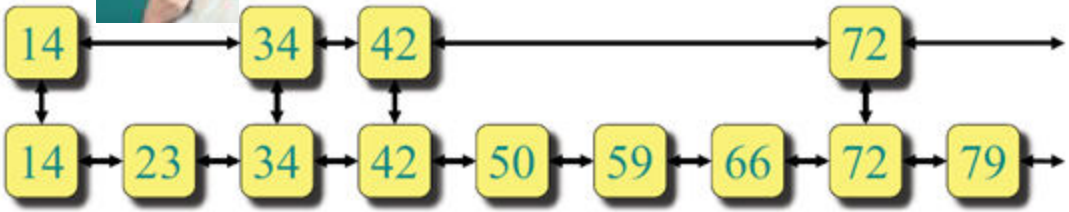
Q: Cum ar trebui distribuite nodurile din L1 (nivelul de mai sus) astfel incat cautarea sa fie cat mai eficienta?

A: Uniform distribuit!

Q: cate noduri ar trebui sa existe in L1?

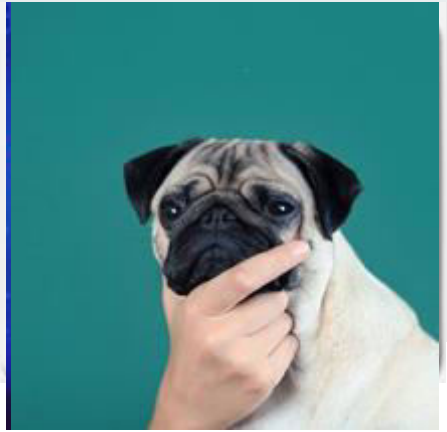
A:

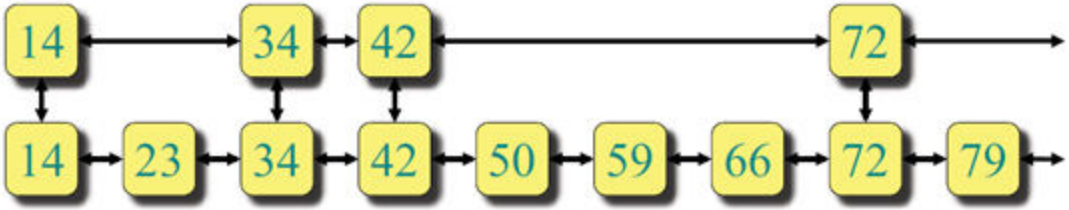




Skip Lists: Numarul de elemente per nivel

Costul unei căutări în listă este aprox $|L1| + \frac{|L2|}{|L1|}$



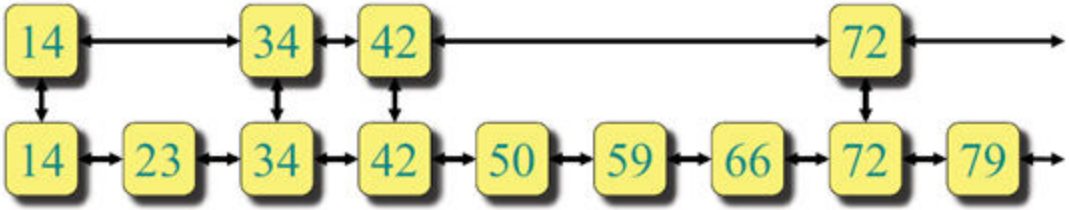


Skip Lists: Numarul de elemente per nivel

Costul unei căutări în listă este aprox $|L1| + \frac{|L2|}{|L1|}$

Relatia de mai sus este minimizata atunci cand $|L1|^2=|L2|=n$; deci

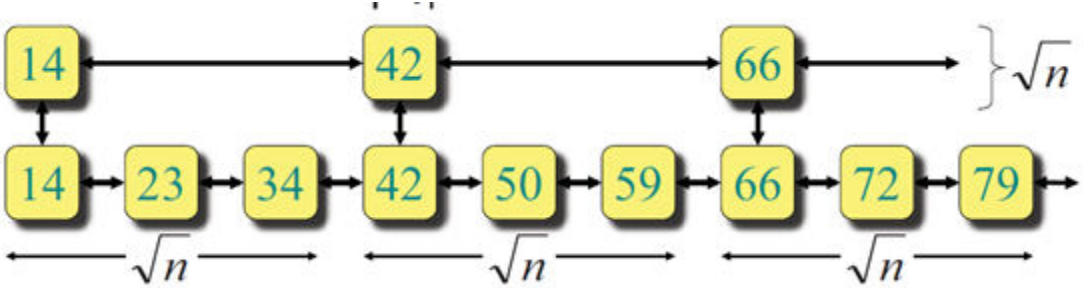
$|L1|=sqrt(n)$

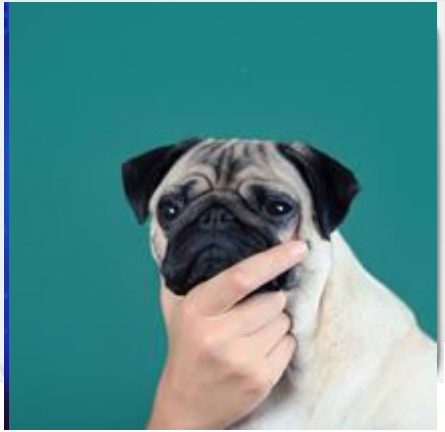


Skip Lists: Numarul de elemente per nivel

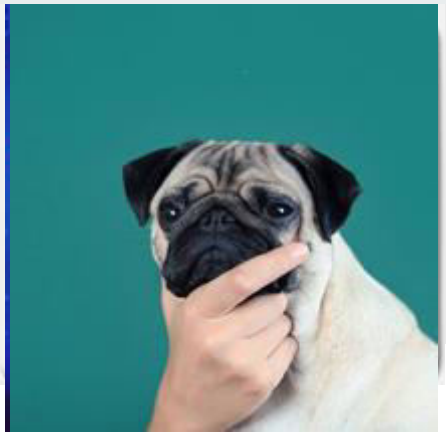
$|L1| = \sqrt{n}$; $|L2| = n$; *Avem costul total de cautare pe 2 nivele*:

$$|L1| + \frac{|L2|}{|L1|} = 2\sqrt{n}$$





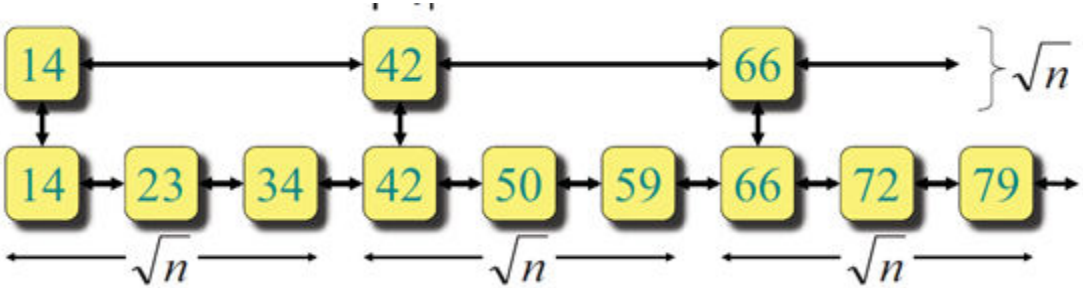
Skip Lists: Numarul de elemente per nivel



$|L1| = \sqrt{n}$; $|L2| = n$; *Avem costul total de cautare pe 2 nivele*:

$$|L1| + \frac{|L2|}{|L1|} = 2\sqrt{n}$$

Dar pentru 3 nivele?
Dar 4 nivele?
Dar *k* nivele?



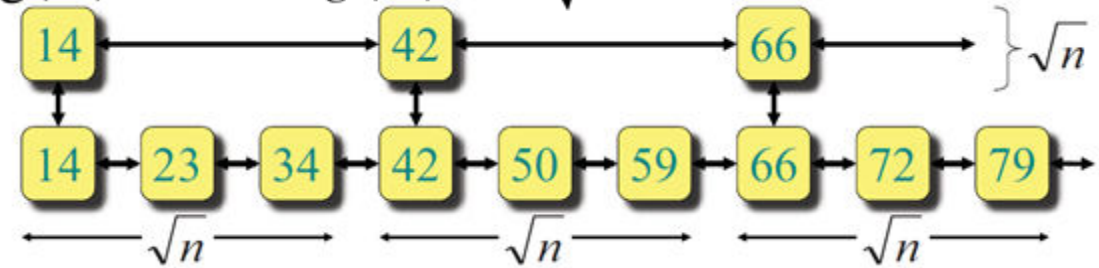
Skip Lists: Numarul de elemente per nivel

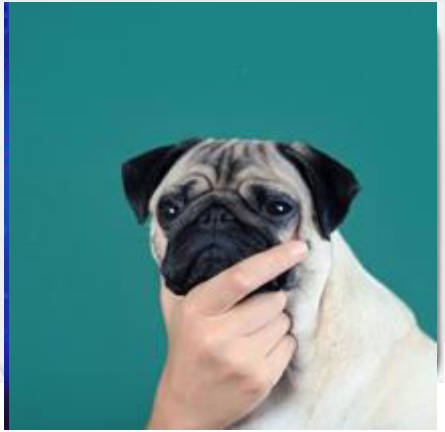
$2\ nivele: 2\sqrt{n}$

$3\ nivele: 3\sqrt[3]{n}$

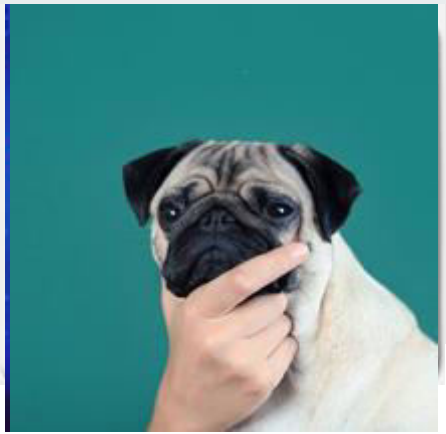
$k\ nivele: k\sqrt[k]{n}$

$\lg(k)\ nivele: \lg(k).\sqrt[\lg(k)]{n}$





Skip Lists: Numarul de elemente per nivel

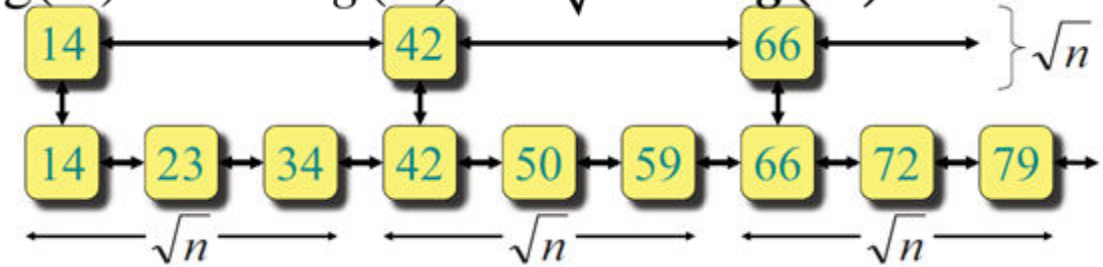


$2\ nivele: 2\sqrt{n}$

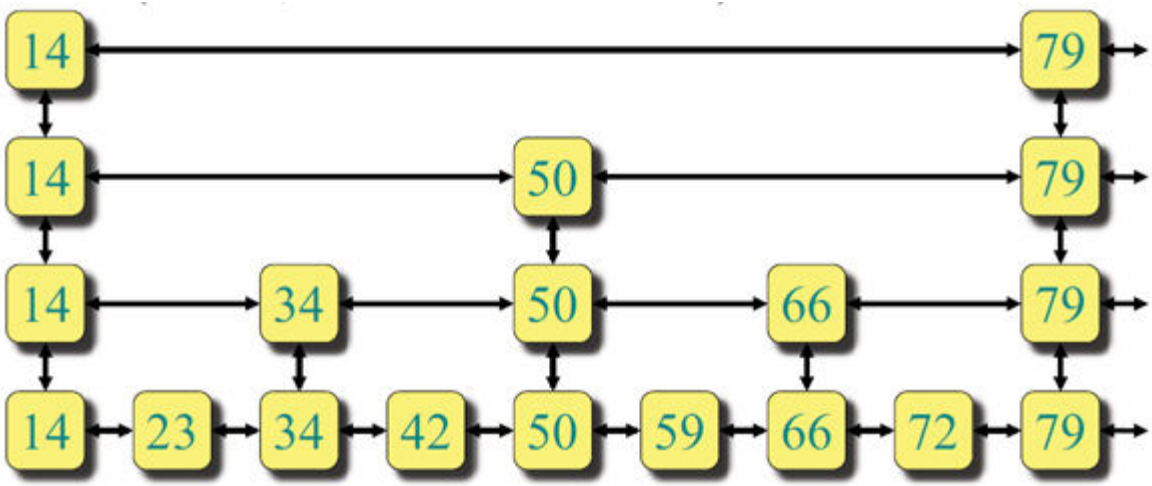
$3\ nivele: 3\sqrt[3]{n}$

$k\ nivele: k\sqrt[k]{n}$

$\lg(k)\ nivele: \lg(k)\sqrt[\lg(k)]{n} = 2\lg(n)$

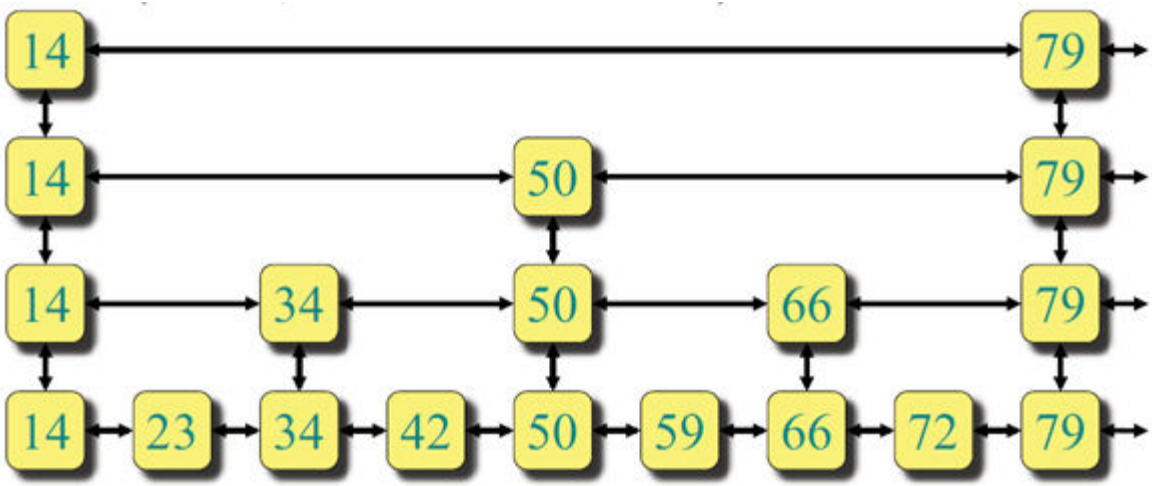


Skip Lists: Numarul de elemente per nivel



Cu ce seamana oare?

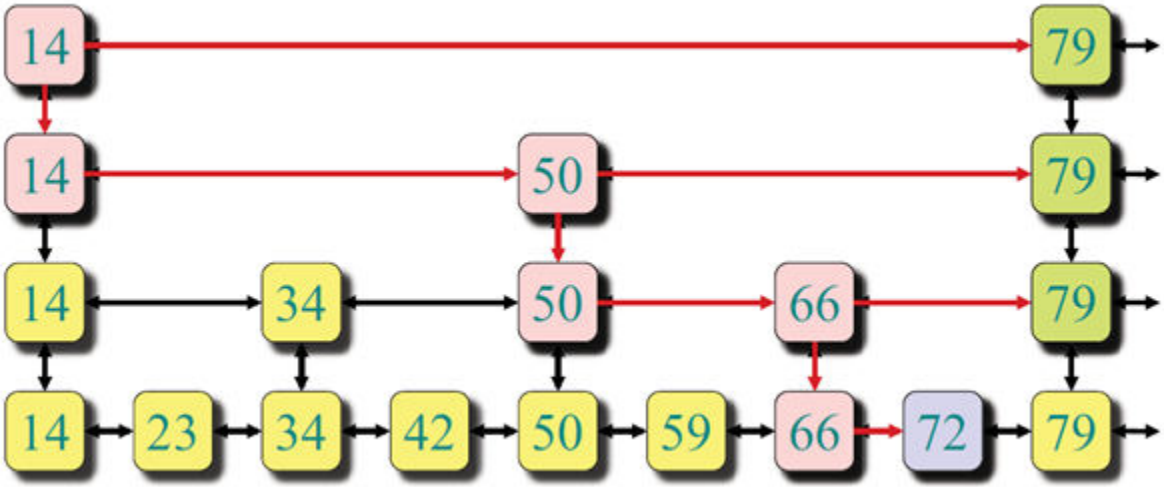
Skip Lists: Numarul de elemente per nivel

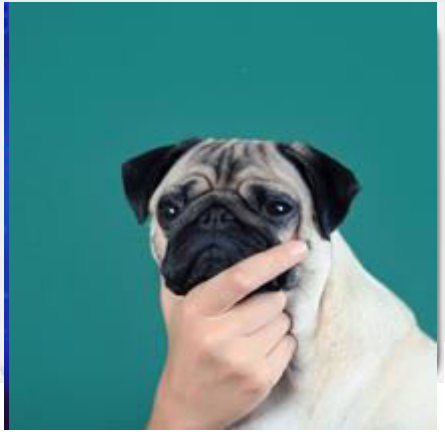


Cu ce seamana oare?

Un arbore!

Skip Lists: Search (X)





Search(72)

Skip Lists: Insert (X)

Pentru a insera un nou element X in lista:

- ii cautam pozitia in nivelul inferior (search(x))
- il inseram pe nivelul inferior

Skip Lists: Insert (X)

Pentru a insera un nou element X in lista:

- ii cautam pozitia in nivelul inferior (search(x))
- il inseram pe nivelul inferior
- il inseram si pe *unele* nivele superioare

OBSERVATIE: Nivelul inferior va contine intoteauna toate elementele

Skip Lists: Insert (X)

Pentru a insera un nou element X in lista:

- ii cautam pozitia in nivelul inferior (search(x))
- il inseram pe nivelul inferior
- il inseram si pe *unele* nivele superioare

OBSERVATIE: Nivelul inferior va contine intoteauna toate elementele

Q: Pe cate alte nivele inserez X?

A: Dau cu banul! Daca pica pajura, inserez pe inca un nivel, altfel ma opresc!

Skip Lists: Insert (X)

Pentru a insera un nou element X in lista:

- ii cautam pozitia in nivelul inferior (search(x))
- il inseram pe nivelul inferior
- il inseram si pe *unele* nivele superioare

OBSERVATIE: Nivelul inferior va contine intoteauna toate elementele

Q: Pe cate alte nivele inserez X?

A: Dau cu banul! Daca pica pajura, inserez pe inca un nivel, altfel ma opresc!

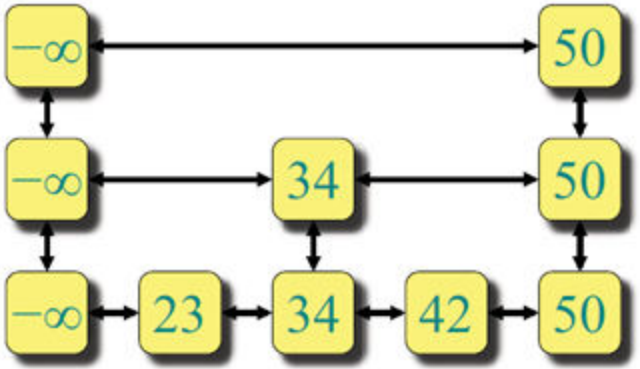
Consecinta: ½ dintre elemente vor fi doar pe nivelul 0. ¼ dintre elemente vor fi doar pe nivelele 0 si 1. ⅛ vor fi doar pe nivelele 0, 1 si 2, etc...

Skip Lists: Exercitiu

EXERCISE: Try building a skip list from scratch by repeated insertion using a real coin

Small change:

- Add special $-\infty$ value to *every* list $\Rightarrow$ can search with the same algorithm



Skip Lists: Delete (x)

Se cauta elementul x in lista (se gaseste pe cel mai de sus nivel).

Se sterge eelemntul de pe toate nivelele!

Skip Lists: How good are they?

Whiteboard S23

Whiteboard S24