

Algoritmi e Strutture Dati

Lezione 17

31 ottobre 2025

Ordinare senza confrontare

Problema: Ordinare n interi in $[0..b-1]$

A	1	0	4	2	0	2	$b=5$
---	---	---	---	---	---	---	-------

Y	2	1	4	0	1	integerSort
	0	1	2	3	4	

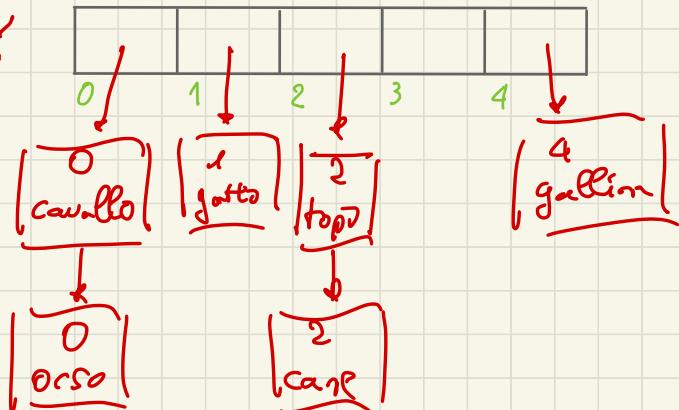
A	10	0	41	22	02	24
---	----	---	----	----	----	----

Problema: Ordinare n record con chiavi intere in $[0..b-1]$

A

1	0	4	2	0	2
gatto	cavollo	gallina	topo	orso	cane

Y



bucketSort

code



stabilità

A

0	0	1	2	2	4
cavollo	orso	gatto	topo	cane	gallina

Problema: Ordinare n record con chiavi intere in $[0..b-1]$

ALGORITMO bucketSort (Array A[0..n-1], intero b)

Sia $Y[0..b-1]$ un array

FOR $i \leftarrow 0$ TO $b-1$ DO

 | $Y[i] \leftarrow$ coda vuota

FOR $i \leftarrow 0$ TO $n-1$ DO

 | $x \leftarrow A[i].chiave$

 | $Y[x].enqueue(A[i])$

$j \leftarrow 0$

FOR $i \leftarrow 0$ TO $b-1$ DO

 | WHILE NOT ($Y[i].isEmpty()$) DO (Viempimento array ordinati)

 | $A[j] \leftarrow Y[i].dequeue()$

 | $j \leftarrow j + 1$

prestazione

bucket

$\Theta(b)$

riempimento

bucket

$\Theta(n)$

b iterazioni

A	1	0	4	2	0	0	2
gatto	cavillo	gallina	topo	orsa	cane		cane

Fermo

Y	0	1	2	3	4
	null				
0	cavillo	gatto	topo		gallina
1				cane	
2					
3					
4					

A	0	0	1	2	2	4
cavillo	orsa	gatto	topo	cane	gallina	

Viempimento array ordinati
 $\Theta(b+n)$
 in pass. $\Theta(b+n)$
 complessità mass.

BucketSort

$$\text{Tempo } \Theta(b) + \Theta(n) + \Theta(b+n) = \Theta(b+n)$$

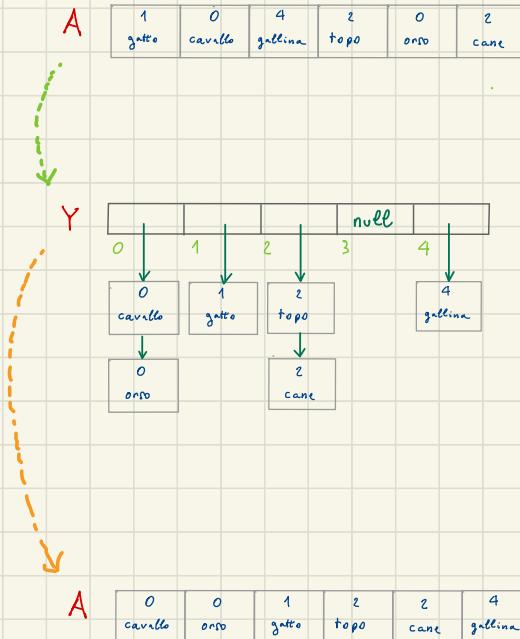
- Se $b = O(n)$ allora $\Theta(b+n) = \Theta(n)$
Tempo Lineare!

- Se $b \approx n^2$ $\Theta(b+n) = \Theta(n^2)$
meglio HeapSort!

NON IN LOCO

Stabile

utilizzabile su liste

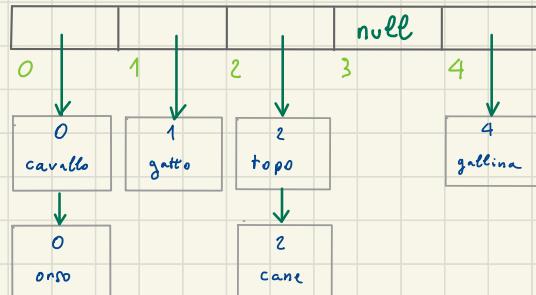


BucketSort può essere utilizzato anche per ordinare liste manipolando direttamente i postiatori

A



Y



colloca ciascun
nodo di A nella
codice corrispondente
alla chiave

A



concatena
le liste (code)
una dopo
l'altra

Problema: ordinare un insieme di persone rispetto alla data del compleanno

Alberto 22 gennaio

Anna 9 dicembre

Claudia 27 aprile

Franco 31 maggio

Lorenzo 15 gennaio

Lucia 14 dicembre

Paolo 12 gennaio

Sara 9 giugno

Problema: ordinare un insieme di persone rispetto alla data del compleanno

bucket sort
(chiave: giorno)



Alberto	22	gennaio	Anna	9	dicembre
Anna	9	dicembre	Sara	9	giugno
Claudia	27	aprile	Paolo	12	gennaio
Franco	31	maggio	Lucia	14	dicembre
Lorenzo	15	gennaio	Lorenzo	15	gennaio
Lucia	14	dicembre	Alberto	22	gennaio
Paolo	12	gennaio	Claudia	27	aprile
Sara	9	giugno	Franco	31	maggio

Problema: ordinare un insieme di persone rispetto alla data del compleanno

bucket sort (chiave: giorno)			bucket sort (chiave: mese)		
Alberto	22	gennaio	Anna	9	dicembre
Anna	9	dicembre	Sara	9	giugno
Claudia	27	aprile	Paolo	12	gennaio
Franco	31	maggio	Lucia	14	dicembre
Lorenzo	15	gennaio	Lorenzo	15	gennaio
Lucia	14	dicembre	Alberto	22	gennaio
Paolo	12	gennaio	Claudia	27	aprile
Sara	9	giugno	Franco	31	maggio

Problema: ordinare un insieme di dati con chiavi intere

1234	5131	224	37	37
37	41	5131	41	41
224	1234	1234	48	48
5131	224	37	5131	224
41	37	4237	224	1234
48	4232	41	1234	1237
4237	48	48	4237	5131

ALGORITMO radixSort (Array A[0..n-1])

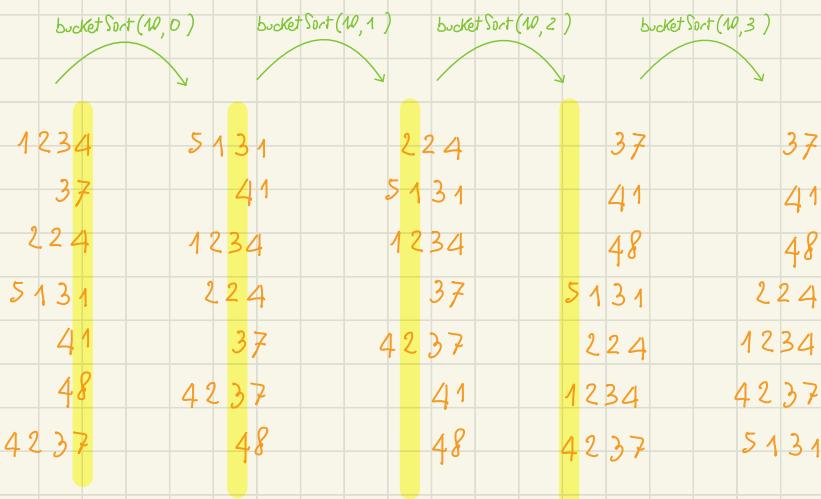
$t \leftarrow 0$

WHILE (\exists chiave K in A t.c. $\lfloor K/b^t \rfloor \neq 0$) DO

 | bucketSort (A, b, t)

 | | $t \leftarrow t+1$

↑
A contiene una chiave
composta da almeno
 $t+1$ cifre



RadixSort

PROCEDURA bucketSort (Array A[0..n-1], intero b, intero t)

 | Si è Y[0..b-1] un array

 | FOR i ← 0 TO b-1 DO

 | | Y[i] ← coda vuota

 | FOR i ← 0 TO n-1 DO

 | | x ← cifra di posto t nella
 | | rappresentazione in base b
 | | di A[i].chiave

 | | Y[x].enqueue (A[i])

 | j ← 0

 | FOR i ← 0 TO b-1 DO

 | | WHILE NOT Y[i].isEmpty () DO

 | | | A[j] ← Y[i].dequeue ()

 | | | j ← j+1

 | predisposizione
 | bucket

 | riempimento
 | bucket

 | riempimento
 | array
 | ordinato

RadixSort

- Se devo ordinare n chiavi tra 0 e $\underbrace{999}_{\text{3 digits}} \underbrace{999}_{\text{3 digits}} \underbrace{999}_{\text{3 digits}}$:

$$b = 10^3 \quad / 1000 \text{ bucket}$$

→ 3 passi. Si: bucketSort

Tempo $\Theta(n)$

RadixSort

Meglio usare b potenza di 2 (divisioni più veloci!)

- Se devo ordinare n chiavi su 32 bit, cioè tra 0 e $2^{32}-1$:

00111101100110110010010000001111
↓ | ↓ | ↓ | ↓ |
8 256 buckets , a posteriori

$$b = 2^8$$

256 buckets

a posteriori

$$b = 2^{16}$$

65536 buckets

2 passaggi

Tempo $\Theta(n)$

Come ottenere la cifra di posizione t di x in base b ?

es base 10, $t=2$

$$x = 1234$$

$$x = 12345$$

$$(x/10^2) \bmod 10$$

$$(x/b^t) \bmod b$$

Come ottenere la cifra di posizione t di x in base b ?

$$(x / b^t) \bmod b$$

es base 16, $b=2$

$$x = 99999$$

$$(9999 / 16^2) \bmod 16 = 7$$

in binario:

x 0010011100001111 >> 8

0010 0111 x/162

© 0000 1111 16-1

Dury

0000 0111

$$(x/16^2) \text{ mod } 16$$

Come ottenere la cifra di posizione t di x in base b ?

$$(x / b^t) \bmod b$$

es base 16, $t=2$

$$x = 9999$$

$$(9999 / 16^2) \bmod 16 = 7$$

in binario:

$$x \underbrace{0010}_{\text{a}} \underbrace{0111}_{\text{dx}} \underbrace{0000}_{\text{di}} \underbrace{1111}_{\text{8 position}}$$

>> 8

$$\nearrow 16^2 = (2^4)^2 = 2^8$$

$$= \underbrace{0010}_{\text{a}} \underbrace{0111}_{\text{dx}} x / 16^2$$

shift
a dx di
8 position:

$$\underbrace{0010}_{\text{a}} \underbrace{0111}_{\text{dx}} \& \text{ AND}$$

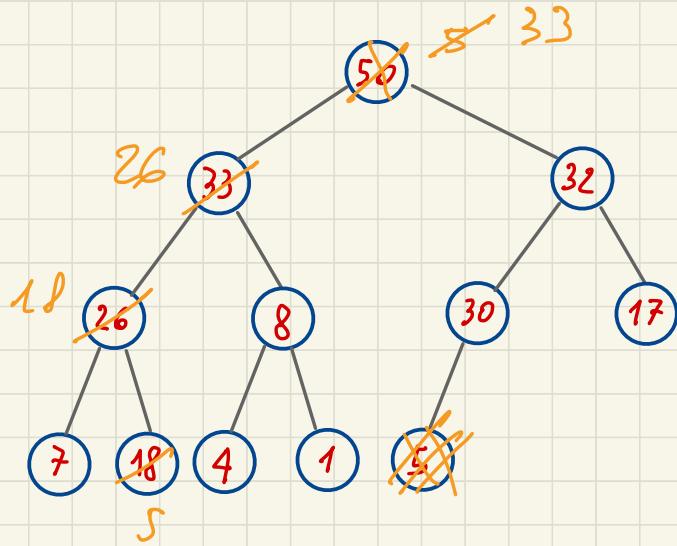
$$00001111 = 2^{16} - 1$$

$$\overline{00000111}$$

$$(x / 16^2) \bmod 16$$

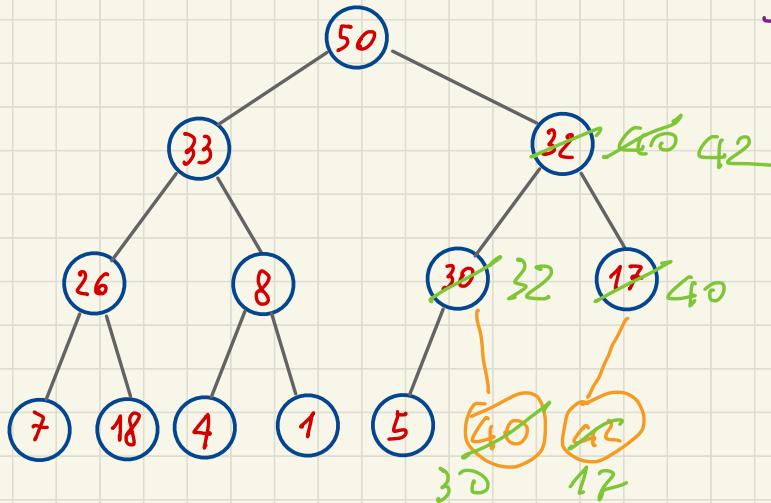
Operazioni su heap

Operazioni su heap



- Trova elemento di chiave massima
passi $\Theta(1)$
- Cancella elemento di chiave massima
passi: $\Theta(\log n)$

Operazioni su heap



- Inserisci un nuovo elemento

aggiungi 40

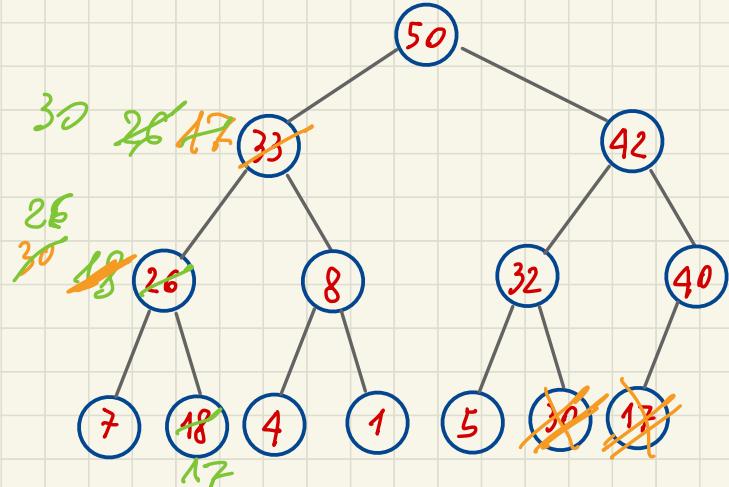
aggiungi 42

aggiungi un nuovo foglio
risistema "dal basso"

$\Theta(\log n)$ passi



Operazioni su heap



Cancella 33

Cancella 18

$\Theta(\lg n)$ pass

(conoscendo dove si trova il nodo da cancellare)

- Cancella un elemento di chiave x

- sostituisce x con proltimo
figlio (che viene eliminato)

- Sia f la chiave dell'ultimo figlio

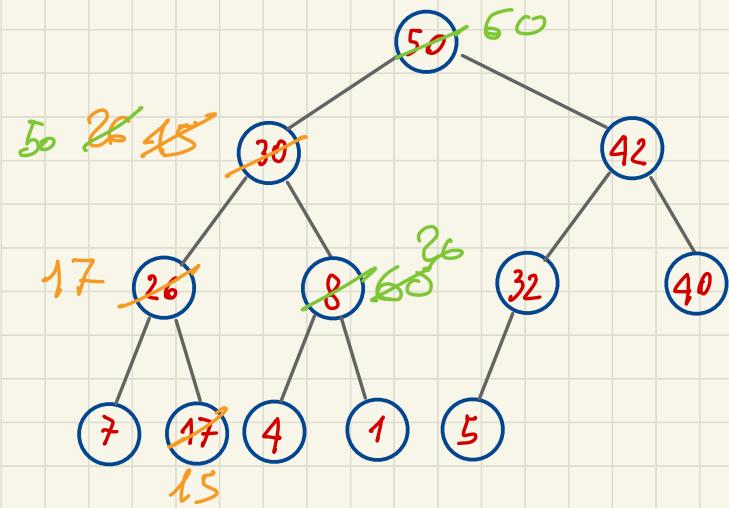
Se $f < x$

applica risistema del
nostro insieme

Se $f > x$

applica risistema del barre
& perfino dal resto non modificati

Operazioni su heap



- Modifica la chiave di un elemento

30 → 15

chiave **decrece**
risistema

8 → 60

chiave **cresce**
risistema dal basso

$O(\log n)$ passi

(supponendo di conoscere
la posizione del nodo da modificare)