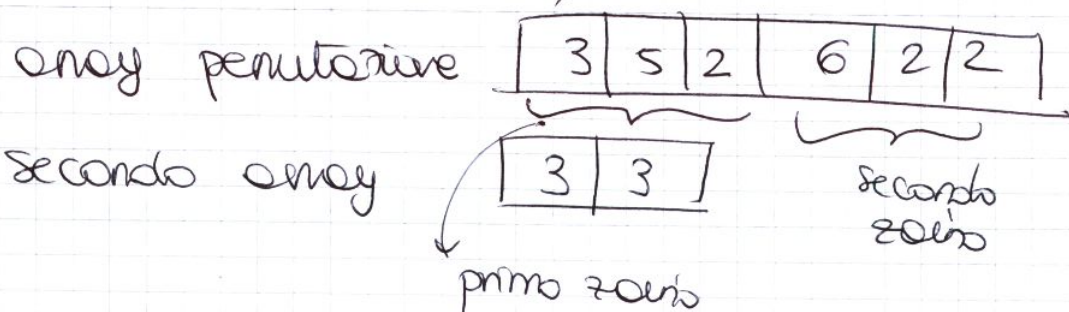


## Schema di Soluzione

- ① Si può usare una modifica di COUNTING-SORT togliendo agli elementi il valore 551, in modo che i valori ~~551~~  $x$  siano compresi tra 0 e 135. La complessità è lineare.
- ② Heap-Delete( $H, i$ ) elimina l'elemento  $i$  sostituendolo con l'ultimo elemento dell'heap, decremantando la size, e ripristinando la proprietà dell'heap chiedendo ~~MAX-HEAPIFY~~ MAX-HEAPIFY oppure controllando l'elemento rispetto ai suoi antenati.
- ③ La soluzione si può rappresentare con due array. Il primo array, di dimensione  $n$ , contiene una permutazione dell'insieme dei pesi degli oggetti. Il secondo array, di dimensione  $R \leq k$ , contiene il numero di oggetti da inserire in ciascuno zaino rispetto alla permutazione.

ESEMPIO

$$Z = 10, R = 2, k = 2.$$



Il problema appartiene alla classe NP perché dato un certificato composto da una permutazione dei pesi, l'algoritmo di verifica li somma ordinatamente fino a raggiungere il valore  $Z$ , incrementa il numero di zaini, e riprende a sommare.  $\#$

Alla fine si controlla se il numero di zaini  $\leq k$ .

~~Verifica~~ Verifica ( $P, k$ )

somma = 0

R = 0

for ( $i = 1, i \leq n; i++$ ) {

if ( $summa + P[i] > Z$ ) {

R++;

summa = P[i];

}

else summa = summa + P[i];

}

if ( $R \leq k$ ) return TRUE;

else return FALSE;

④

L'albero ~~si chiama~~ è composto dalla radice, un sottoalbero completamente bilanciato di altezza  $h-1$ , e un ~~albero~~ sottoalbero di Fibonacci di altezza  $h-2$ .