Schema di Soluzione

(1) Si può usare una modifica di COUNTING-SORT togliendo agli elementi il valore 551, in modo che i valori 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, decrementando la size, e ripristinando le proprietà dell'heap chiamando 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 $z = 10$, $R = 2$, $k = 2$.

array permutazione

3	5	2	6	2	2

secondo array

3	3

primo zaino — secondo zaino

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 è $\leq k$.

~~Verifica~~ Verifica (P, k)

```
somma = 0
R = 0
for (i = 1, i ≤ n; i++) {
    if (somma + P[i] > Z) {
        R++;
        somma = P[i];
    }
    else somma = somma + P[i];
}
if (R ≤ k) return TRUE;
else return FALSE;
```

(4) L'albero ~~si costruisce~~ è composto dalla radice, un sottoalbero completamente bilanciato di altezza $h-1$, e un ~~albero~~ sottoalbero di Fibonacci di altezza $h-1$