Esercizio 1 Un bibliotecario deve disporre n libri su opportuni scaffali. L'i-esimo libro, per $1 \le i \le n$, ha altezza h_i e spessore s_i. Tutti gli scaffali, il cui numero è potenzialmente infinito, hanno una stessa lunghezza L. I libri devono essere disposti secondo un ordine prefissato, dato da un numero di inventario già preassegnato, in modo che la somma degli spessori dei libri su uno stesso scaffale non superi L.

Gli scaffali possono essere posti a varie altezze. La distanza tra due scaffali consecutivi è pari all'altezza del libro più alto posto nello scaffale inferiore. Gli scaffali, inoltre, non devono necessariamente essere riempiti, ma il bibliotecario ha la possibilità di lasciare spazi vuoti al loro interno. Come può il bibliotecario partizionare i libri negli scaffali in modo da minimizzare l'altezza totale degli scaffali utilizzati?
Ad esempio, siano $n = 4$, $L = 4$, ed altezze e spessori come nella seguente tabella:

i	1	2	3	4
h_i	10	20	30	5
s_i	1	2	1	1

- Non è possibile assegnare tutti i libri ad uno stesso scaffale, poiché $\sum_{i=1}^{4} s_i = 5 > L = 4$.
- La partizione che assegna ogni libro ad un diverso scaffale comporta un'altezza totale degli scaffali pari a $10 + 20 + 30 + 5 = 65$.
- La partizione che assegna il libro $\{1\}$ al primo scaffale ed i libri $\{2, 3, 4\}$ al secondo scaffale comporta un'altezza totale degli scaffali pari a $10 + max\{20, 30, 5\} = 40$.
- La partizione che minimizza l'altezza totale degli scaffali assegna i libri $\{1, 2, 3\}$ al primo scaffale ed il libro $\{4\}$ al secondo scaffale e comporta un'altezza pari a $max\{10, 20, 30\} + 5 = 35$.

Per risolvere il problema si propone il seguente algoritmo greedy, dove $SOL[j]$ indica l'insieme dei libri assegnati allo scaffale j:

INPUT: n, L, altezza h_i e spessore s_i di ogni libro
    $j \leftarrow 0$
    $k \leftarrow 0$
    $larghezzaLibri \leftarrow 0$
    FOR $i = 1$ TO n DO
        IF $larghezzaLibri + s_i > L$ THEN
            BEGIN
            $SOL[j] \leftarrow \{k + 1, ..., i - 1\}$
            $j \leftarrow j + 1$
            $k \leftarrow i - 1$
            $larghezzaLibri \leftarrow 0$
            END
        ELSE
            $larghezzaLibri \leftarrow larghezzaLibri + s_i$
    ENDFOR
    $SOL[j] \leftarrow \{k + 1, ..., n\}$
    RETURN SOL

1. *(max 10 punti)* Provare che l'algoritmo proposto non è corretto.
2. *(max 10 punti)* Proporre un algoritmo basato sulla tecnica della programmazione dinamica che in tempo $O(n^2)$ calcoli l'altezza ottima degli scaffali.

3. *(max 10 punti)* Modificare l'algoritmo proposto al punto 2 in modo da avere in output una partizione dei libri in scaffali che minimizzi l'altezza totale degli scaffali utilizzati. La modifica deve avere costo additivo $O(n)$.

Soluzione Esercizio 1

1. Si consideri l'istanza in cui $n = 4$, $L = 3$, ed altezze e spessori sono specificati dalla seguente tabella:

i	1	2	3	4
h_i	3	10	9	5
s_i	1	1	1	1

   L'algoritmo assegna i libri $\{1, 2, 3\}$ al primo scaffale ed il libro $\{4\}$ al secondo scaffale, per un'altezza totale degli scaffali pari a $max\{3, 10, 9\} + 5 = 15$. La partizione che minimizza l'altezza totale degli scaffali assegna invece il libro $\{1\}$ al primo scaffale ed i libri $\{2, 3, 4\}$ al secondo scaffale e comporta un'altezza pari a $3 + max\{10, 9, 5\} = 13$.

2. Studiamo il sottoproblema $\mathcal{P}(i)$ che richiede di calcolare l'altezza minima degli scaffali necessari per disporre i primi i libri. Manteniamo le soluzioni di tali sottoproblemi in un array H di dimensione $n + 1$. Ovviamente, $H[0] = 0$ e $H[n]$ rappresenta la soluzione al nostro problema. Per calcolare $H[i]$, per ogni j tale che $1 \leq j \leq i$, possiamo immaginare di disporre i libri da j ad i in uno stesso scaffale (se possibile) ed i libri da 1 a $j - 1$ in maniera ottima in scaffali la cui altezza totale è data da $H[j - 1]$. Quindi:

$$H[i] = \min_{1 \leq j \leq i \,:\, s_j + \ldots + s_i \leq L} \{H[j - 1] + \max\{h_j, ..., h_i\}\}$$

   Usando questa ricorrenza in una implementazione accorta, possiamo ottenere un algoritmo che calcola $H[n]$ in tempo $O(n^2)$ come segue:

   INPUT: n, L, altezza h_i e spessore s_i di ogni libro
   - $H[0] \leftarrow 0$
   - FOR $i = 1$ TO n DO
     - $larghezza \leftarrow s_i$
     - $altezza \leftarrow h_i$
     - $min \leftarrow H[i - 1] + altezza$
     - $j \leftarrow i - 1$
     - WHILE $j > 0$ AND $larghezza + s_j \leq L$ DO
       - $larghezza \leftarrow larghezza + s_j$
       - $altezza \leftarrow \max\{altezza, h_j\}$
       - IF $min > H[j - 1] + altezza$ THEN $min \leftarrow H[j - 1] + altezza$
       - $j \leftarrow j - 1$
     - ENDWHILE
     - $H[i] \leftarrow min$
   - ENDFOR
   - RETURN $H[n]$

3. Dato l'array H, una partizione ottima dei libri in scaffali può essere ricostruita in tempo $O(n)$ procedendo dall'ultimo verso il primo elemento dell'array come segue:

   INPUT: n, L, altezza h_i e spessore s_i di ogni libro, array H
   OUTPUT: l'array SOL (di dimensione al più n) tale che $SOL[k]$ indica l'insieme dei libri assegnati allo scaffale k

```
j ← n
t ← n
k ← −1
WHILE j > 0 DO
    altezza ← h_j
    WHILE H[t] ≠ altezza + H[j − 1] DO
        j ← j − 1
        altezza ← max{altezza, h_j}
    ENDWHILE
    k ← k + 1
    A[k] ← {j, ..., t}
    j ← j − 1
ENDWHILE
FOR i ← 0 TO k DO SOL[k − i] ← A[i]
RETURN SOL
```

Sebbene ci siano due cicli WHILE nidificati, l'assegnamento $j \leftarrow j - 1$ fa si che ogni elemento dell'array H sia esaminato una sola volta e che il tempo di esecuzione sia pertanto lineare.