Esercizio. Dato un array di numeri $V[1..n]$, calcolare la lunghezza massima per una sottosequenza non decrescente, ovvero il massimo l tale che esistono $x_1, \ldots, x_l$ con $x_i < x_{i+1}$ e $V[x_i] \leq V[x_{i+1}]$, per ogni $i < l$.

Soluzione. Si procede risolvendo i seguenti problemi:

$$P(l) \;=\; \text{valore minimo al termine di una sottosequenza di lunghezza } l\text{, se esiste, } \infty \text{ altrimenti}$$

con $1 \leq l \leq n$ (in quanto le sottosequenze non vuote in un array di lunghezza n hanno una lunghezza che va da 1 a n). Si utilizza un array $P[1..n]$ per memorizzare le soluzioni a tutti i sottoproblemi $P(l)$. Inizialmente si pone, approssimando per eccesso, $P[l] = \infty$ per ogni l. Poi si analizzano iterativamente i valori da $V[1]$ a $V[n]$ e per ogni nuovo valore $V[k]$ si aggiorna il relativo problema che tale valore permette di migliorare. In particolare, tale valore potrà migliorare un sottoproblema andandosi ad inserire successivamente alla sottosequenza più lunga finora trovata che termina con un valore inferiore o uguale ad esso. Se invece tutte le sequenze terminano con un valore più grande, il nuovo valore $V[k]$ permetterà di ridurre la soluzione finora trovata al problema $P[1]$. Più precisamente si cerca il valore $i = max\big(\{0\} \cup \{l \mid P[l] \leq V[K]\}\big)$ e si pone $P[i+1] = V[K]$.

Algorithm 1: MAXLUNGSOTTOSEQUENZA(NUMBER $V[1..n]$) $\rightarrow$ NATURAL

```
NUMBER P[1..n]
NATURAL i, maxL = 0
for l = 1 to n do
    P[l] = ∞
for k = 1 to n do
    i = max({0} ∪ {l | P[l] ≤ V[K]})
    P[i + 1] = V[K]
    if (i + 1 > maxL) then
        maxL = i + 1
return maxL
```

Si noti che $maxL$, inizializzato a 0, tiene traccia della massima lunghezza di sottosequenza finora trovata. Si noti inoltre che il vettore P sarà sempre ordinato in modo non decrescente, in quanto inizialmente contiene tutti valori uguali a ∞, e ad ogni modifica si inserisce il nuovo valore $V[K]$ mantenendo l'ordinamento (in particolare, si colloca $V[k]$ subito dopo i valori minori o uguali ad esso). Quindi, l'operazione di ricerca dell'indice $max\big(\{0\} \cup \{l \mid P[l] \leq V[K]\}\big)$ può essere effettuata con una ricerca binaria, di costo $O(\log n)$. Il costo computazionale di tale algoritmo risulta quindi essere $O(n \log n)$ in quanto questa ricerca binaria viene effettuata n volte.