

Sia dato il seguente algoritmo di ordinamento in ordine non decrescente dei valori contenuti nell'array A.

```
BUBBLESORT(A)
1  n ← length[A]
2  while n > 1
3    do i ← 1
4      while i < n
5        do if A[i] > A[i+1]
6          then scambia A[i] ↔ A[i+1]
7          i ← i + 1
8    n ← n - 1
```

- a) Analizzarne la complessità temporale.
- b) Proporne una versione ricorsiva.
- c) Analizzare la complessità temporale dell'algoritmo ricorsivo proposto.
- d) Analizzare la complessità spaziale dell'algoritmo ricorsivo proposto.
- e) Modificare la versione iterativa dell'algoritmo in modo che ogni iterazione scorra l'array A dalla fine verso l'inizio.
- f) Proporre una versione ricorsiva dell'algoritmo di cui al punto e).
- g) Analizzare la complessità temporale dell'algoritmo ricorsivo proposto al punto precedente.
- h) Analizzare la complessità spaziale dell'algoritmo ricorsivo proposto al punto f).
- i) Modificare la versione iterativa dell'algoritmo in modo che alterni iterazioni in cui scorre l'array A dall'inizio verso la fine a iterazioni in cui scorre l'array A dalla fine verso l'inizio. (L'algoritmo risultante prende il nome di SHAKERSORT.)
- j) Analizzare la complessità temporale dell'algoritmo proposto al punto precedente.
- k) Proporre una versione ricorsiva di SHAKERSORT.
- l) Analizzare la complessità temporale della versione ricorsiva di SHAKERSORT proposta.
- m) Analizzare la complessità spaziale della versione ricorsiva di SHAKERSORT proposta.
- n) Migliorare l'algoritmo assegnato e quelli proposti ai punti precedenti così da effettuare una nuova iterazione o una nuova chiamata ricorsiva solo se il (sotto)array da ordinare non è già ordinato.