```
quickSort(ITEM[] A, int primo, int ultimo)
    // su almeno due elementi
    if primo < ultimo then
        int j ← perno(A, primo, ultimo)  // logica dell'algoritmo

        // richiamo l'algoritmo su entrambi i sottovettori
        quickSort(A, primo, j − 1)
        quickSort(A, j + 1, ultimo)

// sposta gli elementi più piccoli a sinistra del perno, i più grandi a destra
int perno(ITEM[] A, int primo, int ultimo)
    ITEM x ← A[primo]  // il perno è il primo elemento
    int j ← primo  // il cursore parte dal primo elemento

    // spostamenti "in-place"
    da i ← primo fino a ultimo fai
        if A[i] < x then // l'elemento è più piccolo del perno
            j + +  // sposta il cursore j
            swap(A[i], A[j])  // scambia gli elementi:  i ↔ j

    /* a questo punto tutti gli elementi posizionati prima della posizione j sono più piccoli
       del perno, rimane solo da riposizionare il perno nella sua posizione finale          */

    // riposiziono il perno
    A[primo] ← A[j] A[j] ← x

    // restituisco la posizione del perno
    return j
```