Algoritmo 1.1 – Algoritmo di ordinamento D&I

```
mergeSort(ITEM[] A, int primo, int ultimo)
    se primo < ultimo allora // devono esistere almeno due elementi
        int mezzo ← ⌊(primo+ultimo)/2⌋
        mergeSort(A, primo, mezzo)
        mergeSort(A, mezzo+1, ultimo)
        merge(A, primo, ultimo, mezzo)

merge(ITEM A, int primo, int ultimo, int mezzo)
    int i, j, k, h

    // Inizializzo i puntatori
    i ← primo j ← mezzo k ← primo
    // k:  indica la prossima posizione di scrittura

    // fintanto che entravi
    finché i ⩽ mezzo and j ⩽ ultimo fai
        se A[i] ⩽ A[j] allora
            // l'elemento è gia ordinato
            B[k] ← A[i]
            i++
        altrimenti
            B[k] ← A[j]
            j++

        // in entrambi i casi ho inserito un valore
        k++
    // se uno dei due vettori finisce ricopio la parte ordinata alla fine del vettore
       d'appoggio
    j ← ultimo
    da h ← mezzo fino a i fai
        A[j] ← A[h]
        j −−

    // ricopio il vettore d'appoggio del vettore originale
    da j ← primo fino a k − 1 fai
        A[j] ← B[j]
```

Equazione di ricorrenza:

$$T = \begin{cases} \Theta(1) & n = 1 \\ T(\lfloor n/2 \rfloor) + T(\lfloor n/2 \rfloor) + \Theta(n) & n > 1 \end{cases} = \begin{cases} c & n = 1 \\ 2T(n/2) + dn & n > 1 \end{cases}$$

Analisi per livelli:

$$\mathcal{O}\left(\sum_{i=0}^{k} 2^i \frac{n}{2^i}\right) = \mathcal{O}\left(\sum_{i=0}^{k} n\right) = \mathcal{O}(k \cdot n) = \mathcal{O}(n \log n)$$

Teorema dell'esperto:

$$\alpha = \log_2 2 = 1 \qquad T = \mathcal{O}(n^\alpha \log n)$$
$$\beta = 1 \qquad = \mathcal{O}(n \log n)$$
$$\alpha = \beta$$