(I) Dato un array non ordinato di interi $A[1,n]$ e un intero $1 \le k \le n$ realizzare una procedura

select (A, k)

che restituisca il k-mo elemento più piccolo (elem. di posiz. k nell'array ordinato

In tempo

(a) $O(n \log n)$

(b) $O(n + k \log n)$

(c) $O(n \log k)$

(a) ordino il vettore!

```
select (A, k)
    n = A.length
    MergeSort (A, 1, n)        ~ Θ(n log n)
    return A[k]
```

(b) Costruisco un min-heap e estraggo k volte l'elemento più piccolo

```
Select (A, k)
    Build MinHeap (A)          ~ Θ(n)
    for i = 1 to k             ← O(k log n)
        x = ExtractMin (A)
    return x
```

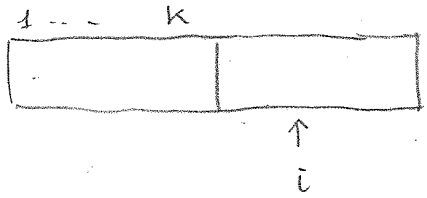
invariante di ciclo
$A[1, n-i+1]$ contiene gli $n-i$ elementi più grandi di A

$\Theta(n) + O(k \log n) = O(n + k \log n)$

(c) Costruisco un max heap con i primi K elementi e scorrendo i rimanenti, confrontandoli con il massimo dei K ed eventualmente inserendoli al posto del massimo ottengo i K elementi più piccoli. Il massimo è quello cercato

Select (A, K)

$\Theta(K)$ { Build Max Heap (A, K) ← solo i primi K elementi



m log K { for i = K+1 to m
    if A[i] < A[1] ← Heap Max (A)
        A[i] ↔ A[1]
        Max Heapify (A, 1, K)
return A[1] ← alla fine è il massimo dei K elementi più piccoli

invariante
A[1, K] Max heap
A[K+1, i-1] ≥ A[1, K]

$O(m \log K)$

(d) Esiste una soluzione lineare, variante di QuickSort con una 5-partizione → vedi libro.