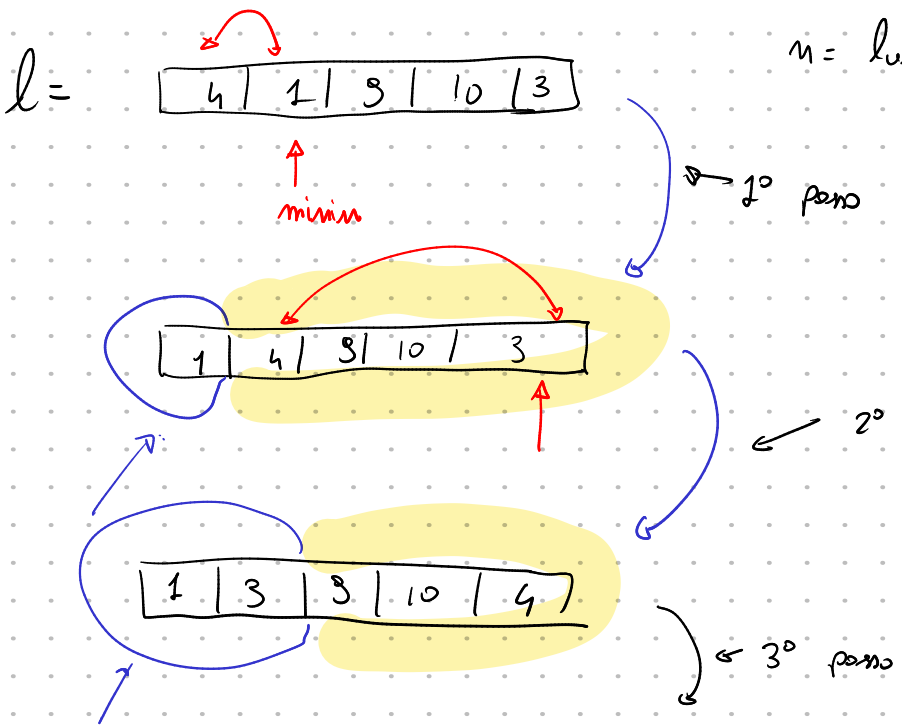
```python
def minimum_position(l, start):
    """
    Restituisce la posizione dell'elemento
    Si assume che start sia una posizione v
    """
    minpos = start
    for i in range(start + 1, len(l)):
        if l[i] < l[minpos]:
            minpos = i
    return minpos

def selection_sort(l):
    """Ordina gli elementi di l."""
    for i in range(len(l) - 1):
        m = minimum_position(l, i)
        l[i], l[m] = l[m], l[i]
```

Domanda:

Come varia il tempo di esecuzione di SELECTIO_SORT al variare della lunghezza di l.

Domanda effettiva alla quale possiamo rispondere

Quante scansioni della lista l esegue la funzione SELECTION_SORT al variare della lunghezza di l?



n = lunghezza di l

$n-1$ passi:

1° passo: (n) letture di l per trovare il minimo
2 letture ulteriori per effettuare lo scambio

2° passo: $n-1$ letture per trovare il minimo della parte rimanente
2 letture per scambiare

3° passo: $n-2$ letture per trovare il minimo della parte rimanente
2 letture per lo scambio

ultimo passo: (2) letture per trovare il minimo (perché la parte grande si è ridotta a 2 soli elementi)
2 letture per scambiare

Totale =

$$\underbrace{(n+2)}_{1^\circ \text{ passo}} + \underbrace{(n-1+2)}_{2^\circ \text{ passo}} + \underbrace{(n-2+2)}_{3^\circ \ldots \text{ passo}} + \ldots + \underbrace{(2+2)}_{(n-1)^\circ \text{ passo}}$$

$$= \left(n + (n-1) + (n-2) + (n-3) + \dots 2 \right) + \overbrace{\left(2 + 2 + 2 + 2 \dots + 2 \right)}$$

tempo per la ricerca dei minimi ↑ — tempo per gli scambi ↑

Somma estremi

$$= \frac{(n+2)(n-1)}{2} + 2(n-1)$$

sono $n-1$ addendi tutti uguali a 2

$$= \frac{n^2 + 2n - n - 2}{2} + 2n - 2$$

$$= \frac{n^2}{2} + \frac{n}{2} - 1 + 2n - 2$$

$$= \frac{n^2}{2} + \frac{5}{2}n - 3$$

lo possiamo ignorare

$n = 1000$ $\quad \frac{n^2}{2} = 500.000$ $\quad \frac{3}{2} = 1.500$

Il numero di accessi alla lista l è $O(n^2)$

Se $n = 100$ $\quad n^2 = 10000$

$n = 2 \cdot 100$ $\quad n^2 = (2 \cdot 100)^2 = 2^2 \cdot 100^2 = 4 \cdot 10.000$

4 volte il risultato di prima

---

ricerca lineare: numero accessi alla lista n $\quad O(n)$

nel caso pessimo

ricerca binaria: numero accessi alla lista $\leq \log_2 n$ $\quad O(\log n)$

$n = 1000$ $\quad \log_2 1000 \simeq 10$

il vecchio valore

$n = 2 \cdot 1000$ $\quad \log_2 (2 \cdot 1000) = \log_2 2 + \log_2 100$

$= 1 + \log_2 1000$

Ci sono algoritmi di ordinamento più efficienti del selection sort?

- merge sort
- heap sort
- quick sort

$\Big\}$ $O(n \log n)$