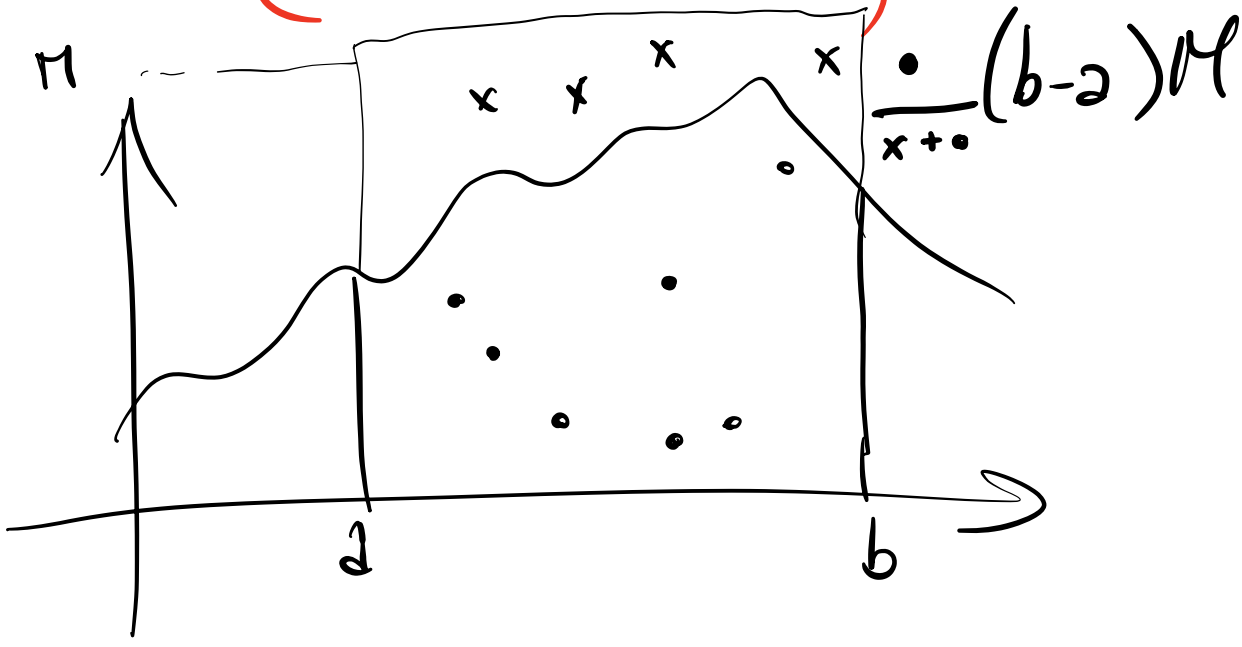
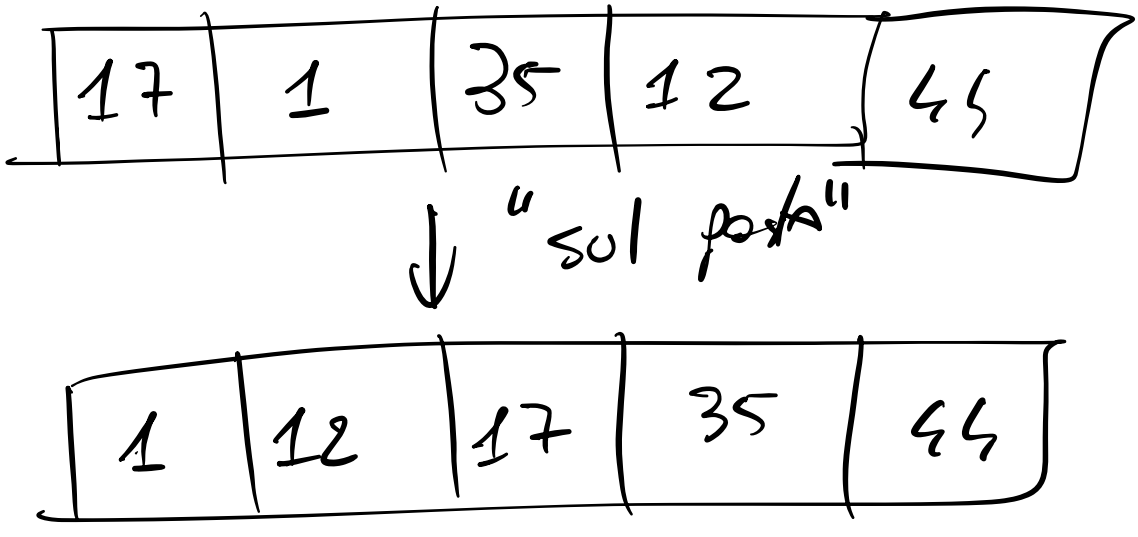
ANALISI NUMERICA

INTEGRAZIONE MONTECARLO (HIT-OR-MISS)



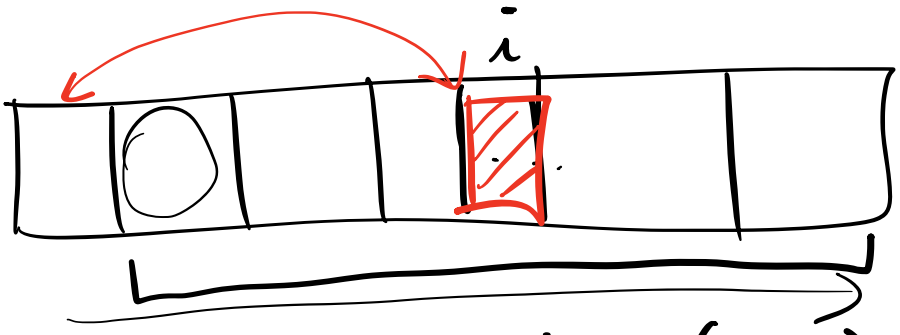
ORDINAMENTO DI SLICE TCP



$O(n^2)$

$O(n \lg n)$

SELECTION SORT



$$n + (n-1) + (n-2) + \dots + 1$$

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

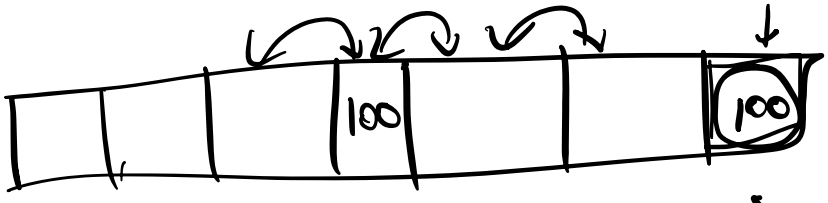
```
func selectionSort (x int[]) {
    n := len(x)
    for i:=0; i<n . i++ {
        La slice x[i:] è disordinata
        mentre x[:i] è ordinata
        min = x[i]
        indexMin = i
        for j:=i+1; j<n; j++ {
            if x[j] < min {
                min = x[j]
                indexMin = j
            }
        }
        x[i], x[indexMin] = x[indexMin], x[i]
    }
}
```

x[i:], x[indexMin] = x[indexMin], x[i]

}

}

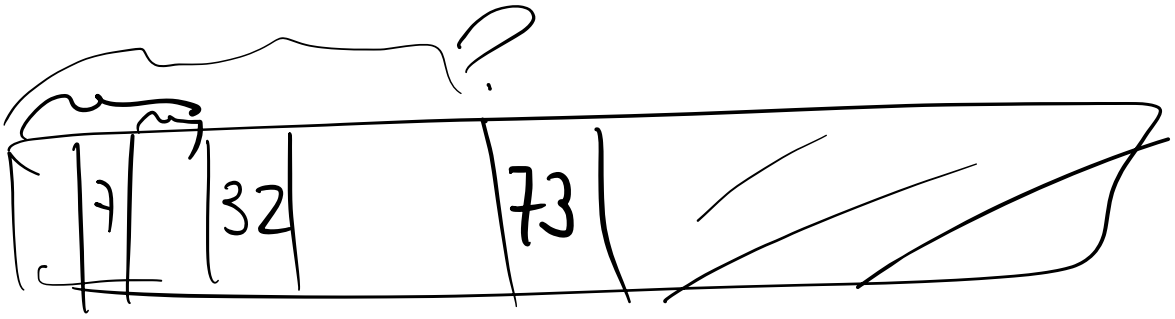
BUBBLE SORT




```
func bubbleSort(x int[]) {
    n := len(x)
    for i := n; i > 0; i-- {
        swap := False
        for j := 0; j < i; j++ {
            if x[j] > x[j+1] {
                x[j], x[j+1] = x[j+1], x[j]
                swap = True
            }
        }
        if !swap { return }
    }
}
```

x[i:] è ordinato e contiene gli elementi più grandi

RICERCA DICOTOMICA



15

$n \quad \frac{n}{2} \quad \frac{n}{4} \ldots$

N° confronti è il minimo k t.c.

$$\frac{n}{2^k} = 1$$

$$n = 2^k$$

$$\log_2 n = \log_2 2^k$$

$$\boxed{k = \log_2 n}$$

```
func binarySearch(x []int, needle int)
                                bool {
    left, right := 0, len(x)
    for left < right {
        mid := (left+right)/2
        if x[mid] == needle {
            return true
        }
        if x[mid] < needle {
            left = mid+1
        } else {
            right = mid
        }
    }
    return false
}
```