Algoritmi di ordinamento

Bubble sort

- Immagina delle carte con dei numeri in fila.
- Confronti due carte alla volta.
- Se sono nell'ordine sbagliato, le scambi.
- Ripeti fino a quando tutti i numeri sono ordinati.

In [3]:
```python
def bubble_sort(lista):
    n = len(lista)

    for i in range(n - 1):  # Passaggi attraverso la lista
        for j in range(n - 1 - i):  # Confronti tra coppie di elementi
            if lista[j] > lista[j + 1]:  # Se sono nell'ordine sbagliato, scambiali
                lista[j], lista[j + 1] = lista[j + 1], lista[j]
```

In [16]:
```python
# Esempio di utilizzo
numeri = [5, 2, 9, 1, 5, 6]
print(numeri)
bubble_sort(numeri)
print(numeri)
```

```
[5, 2, 9, 1, 5, 6]
[1, 2, 5, 5, 6, 9]
```

Selection sort

- Trova il numero più piccolo nella lista e mettilo in prima posizione.
- Trova il secondo più piccolo e mettilo in seconda posizione.
- Ripeti fino a ordinare tutta la lista.

In [19]:
```python
def selection_sort(arr):
    n = len(arr)

    for i in range(n):
        min_index = i  # Assume che l'elemento corrente sia il più piccolo

        for j in range(i + 1, n):  # Cerca il valore più piccolo nella parte rimanente
            if arr[j] < arr[min_index]:
                min_index = j

        # Scambia l'elemento più piccolo trovato con quello nella posizione corrente
        arr[i], arr[min_index] = arr[min_index], arr[i]
```

In [25]:
```python
# Esempio di utilizzo
numeri = [5, 2, 9, 1, 5, 6]
print(numeri)
selection_sort(numeri)
print(numeri)
```

```
[5, 2, 9, 1, 5, 6]
[1, 2, 5, 5, 6, 9]
```

Differenze in termini di complessità (AVANZATO)

Algoritmo	Complessità Caso Peggiore	Complessità Caso Medio	Complessità Caso Migliore	Spazio Extra	Stabilità
Bubble Sort	**$O(n^2)$**	**$O(n^2)$**	**$O(n)$** (se già ordinato)	O(1)	Sì
Selection Sort	**$O(n^2)$**	**$O(n^2)$**	**$O(n^2)$**	O(1)	No

Bubble Sort

- Confronta **elementi adiacenti** e li scambia se non sono nell'ordine giusto.
- **MOLTO lento** su liste grandi perché fa sempre **$O(n^2)$** confronti.
- **Veloce solo se la lista è quasi ordinata** (**$O(n)$** in quel caso).

Esempio: Se hai `n = 1000`, Bubble Sort fa circa **1.000.000** operazioni!

Selection Sort

- **Trova il minimo** e lo mette nella posizione giusta, ripetendo il processo.
- **Sempre $O(n^2)$**, anche se la lista è già ordinata.
- **Più efficiente di Bubble Sort** perché fa **meno scambi**, ma resta lento.

Chi è più veloce?

Selection Sort è più veloce in generale perché fa meno scambi.
Bubble Sort è accettabile **solo** se la lista è quasi ordinata.

Per liste più grandi di **100 elementi**, è meglio usare **Quick Sort o Merge Sort**!