

## Bài 8.4: Thuật toán sắp xếp chèn

---

- ✓ Các đặc điểm
- ✓ Thuật toán sắp xếp chèn
- ✓ Ví dụ minh họa
- ✓ Bài tập thực hành

# Các đặc điểm

- Thuật toán sắp xếp chèn có độ phức tạp  $O(n^2)$ .
- Thuật toán thực hiện việc sắp xếp bằng cách lần lượt đưa từng phần tử từ phần chưa được sắp xếp về vị trí phù hợp ở đầu mảng.
- Thuật toán chia mảng cần sắp xếp thành 2 phần:
  - Phần đầu mảng chứa các phần tử đã được sắp xếp.
  - Phần sau của mảng chứa các phần tử chưa được sắp xếp.
- Tại mỗi lần lặp, tất cả các phần tử lớn hơn phần tử mục tiêu sẽ được dịch phải 1 đơn vị để tìm chỗ chèn phần tử mục tiêu vào.
- Sau đó phần tử mục tiêu sẽ được đưa vào vị trí phù hợp ở phần mảng đã được sắp xếp.

# Thuật toán sắp xếp chèn

➤ Mã giả của thuật toán insertion sort:

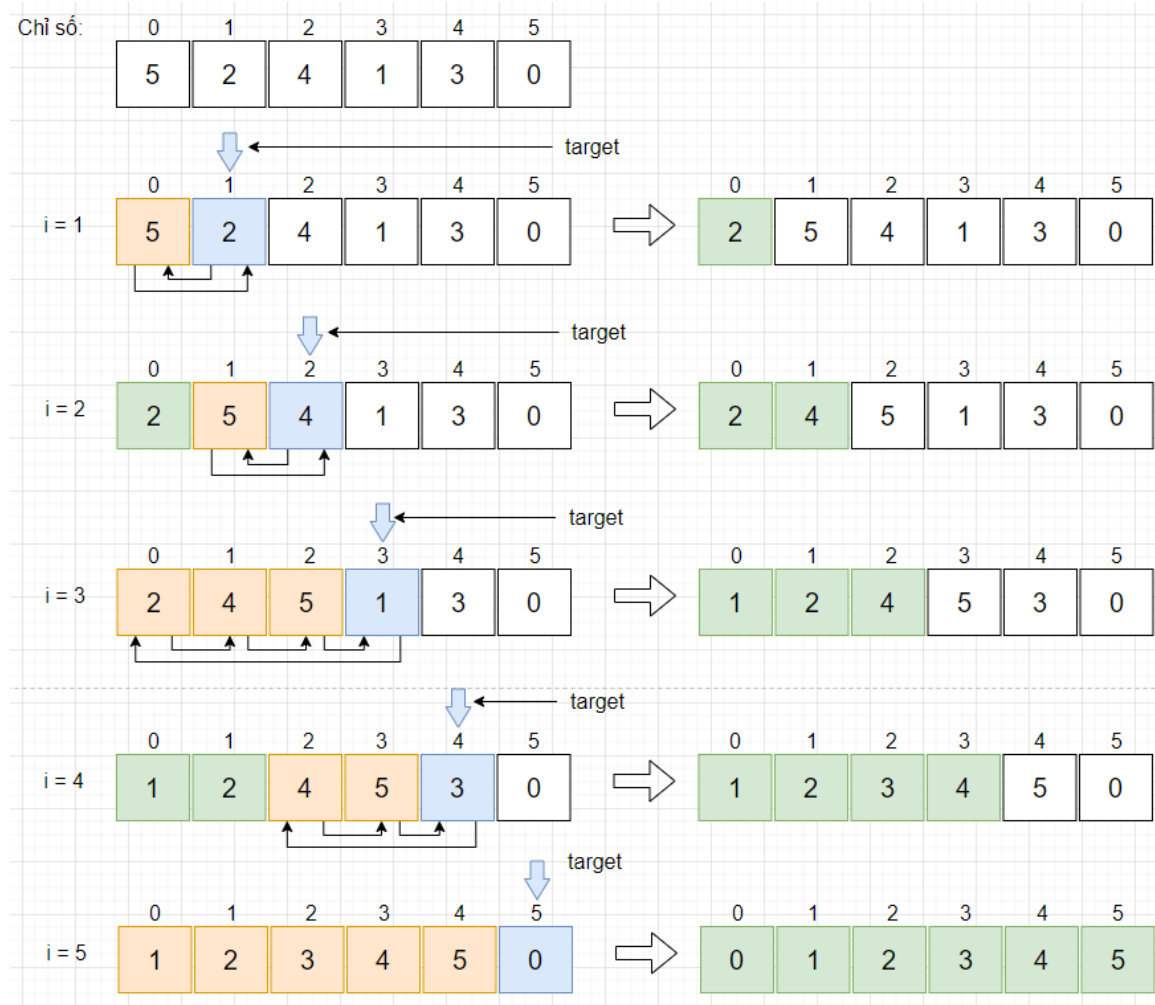
```
void insertionSort(arr[]) {    // arr: mảng đầu vào
    n = arr.length;           // số phần tử trong mảng
    for (i from 1 to n - 1) {  // cho i từ 1 đến n-1
        target = arr[i];       // giá trị cần đưa về vị trí phù hợp
        j = i - 1;             // khởi tạo biến chạy j
        while (j >= 0 && arr[j] > target) {
            arr[j + 1] = arr[j]; // dịch phải phần tử tại vị trí j
            j--;                 // giảm j
        }
        arr[j + 1] = target;    // gán phần tử mục tiêu vào vị trí
    }
}
```

# Thuật toán sắp xếp chèn

➤ Mã thật của thuật toán selection sort:

```
// thuật toán sắp xếp chọn  
public static <T extends Comparable<T>> void insertionSort(T[] arr) {  
    int n = arr.length;  
    for (int i = 1; i < n; i++) {  
        T target = arr[i];  
        int j = i - 1;  
        while (j >= 0 && arr[j].compareTo(target) > 0) {  
            arr[j + 1] = arr[j];  
            j--;  
        }  
        arr[j + 1] = target;  
    }  
}
```

# Minh họa thuật toán



# Nội dung tiếp theo

**Thuật toán sắp xếp shell sort**