CÁC PHƯƠNG PHÁP SẮP XẾP CƠ BẢN

Mục tiêu

- Hiểu rõ tư tưởng các thuật toán sắp xếp cơ bản
- Nắm vững và cài đặt được các thuật toán sắp xếp cơ bản trên ngôn ngữ lập trình C++

Nội dung

- Bài toán sắp xếp
- Ba phương pháp sắp xếp:
  - Sắp xếp lựa chọn
  - Sắp xếp chèn
  - Sắp xếp nổi bọt

Bài toán sắp xếp

- Cho một dãy gồm N đối tượng, cần sắp xếp các đối tượng theo thứ tự nhất định theo một hay một tập thuộc tính của các đối tượng

Cụ thể hóa bài toán:

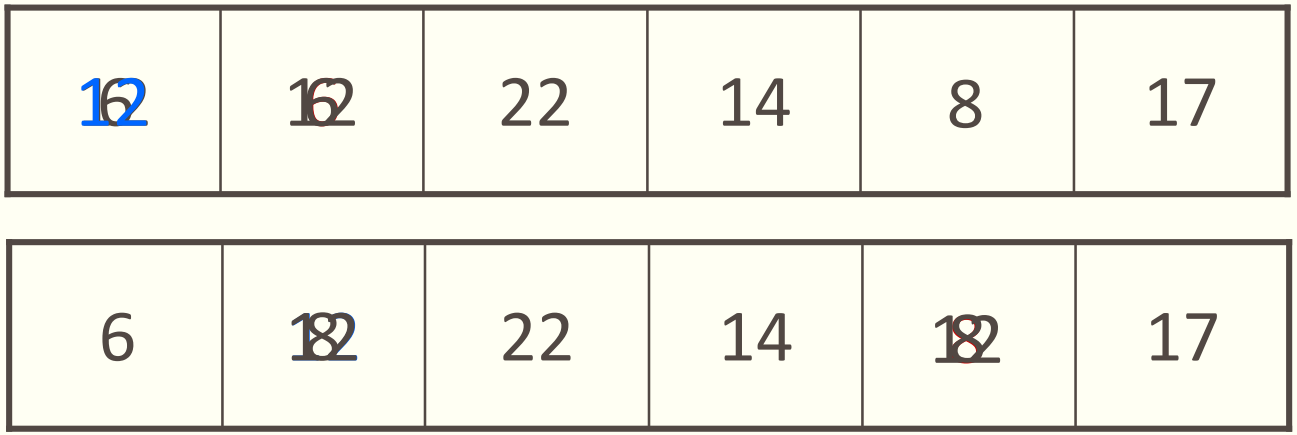
Cho một dãy gồm n số nguyên, sắp xếp dãy theo thứ tự tăng dần

Sắp xếp lựa chọn

- Ý tưởng: Mỗi bước lặp chọn lựa phần tử nhỏ nhất đưa về đầu dãy

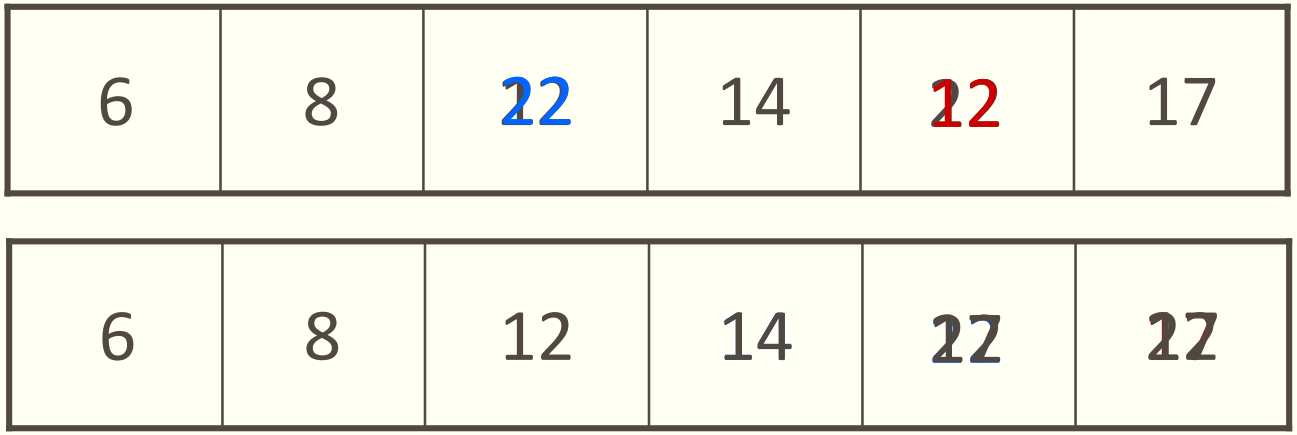
- Cách thực hiện : Lặp n-1 bước:
  - Đánh dấu giá trị "đầu tiên" chưa được sắp
  - Tìm giá trị nhỏ nhất trong các số chưa được sắp
  - hoán đổi giá trị đánh dấu và giá trị nhỏ nhất tìm được

Selection Sort



- Cho N số cần sắp xếp
- Lặp đi lặp lại n-1 lần những bước sau:
  - Đánh dấu giá trị "đầu tiên" chưa được sắp
  - Tìm giá trị nhỏ nhất trong các số chưa được sắp
  - hoán đổi giá trị đánh dấu và giá trị nhỏ nhất tìm được

Ví dụ minh họa



Thuật toán

```
void SelectionSort(int a[],int n){
          for(int i=1;i<=n-1;i++){
         int min=a[i];
         int index=i;
         for (int j=i+1;j<=n;j++)
             if (min>a[j]){min=a[j]; index=j;}
             int tmp;
             tmp=a[i];
             a[i]=a[index];
             a[index]=tmp;
       }
   }
```

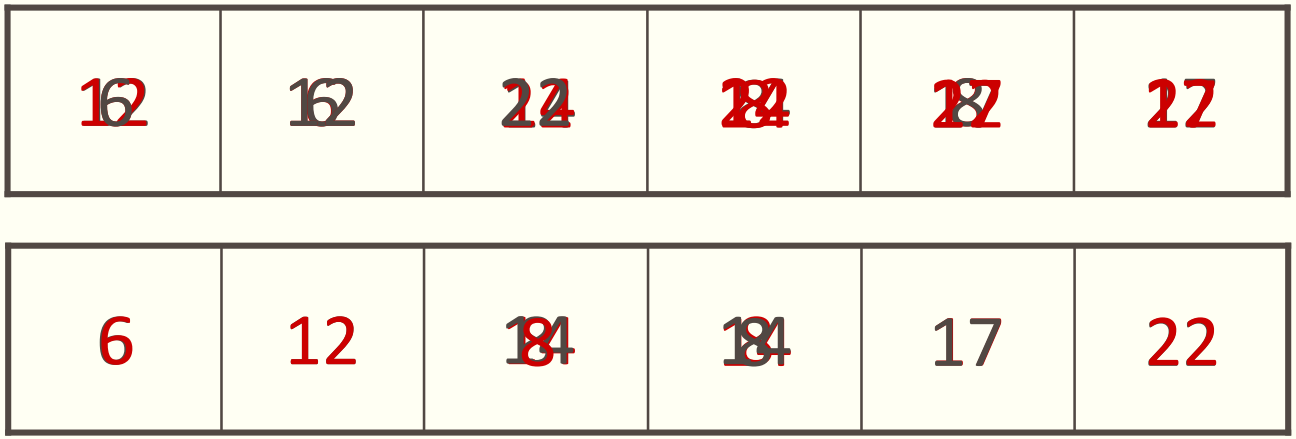
- Độ phức tạp của thuật toán: $O(n^2)$

Sắp xếp nổi bọt

- Ý tưởng: Mỗi bước lặp, hoán vị các phần tử kề nhau khi chúng chưa có thứ tự đúng

- Cách thực hiện : Lặp n-1 bước:
  - Duyệt dãy từ trái qua phải, với mỗi cặp số kề nhau:
    - Nếu số bên trái lớn hơn số bên phải thì hoán đổi giá trị cho nhau.

Ví dụ minh họa



Ví dụ minh họa

6	12 8	12 8	14	17	22

6	8	12	14	17	22

Thuật toán

```
void BubbleSort(int a[],int n){
     for(int i=1;i<=n-1;i++)
       for (int j=1;j<=n-1;j++)
        if (a[j]>a[j+1]){
          int tmp;
          tmp=a[j];
          a[j]=a[j+1];
          a[j+1]=tmp;
     }
}
```

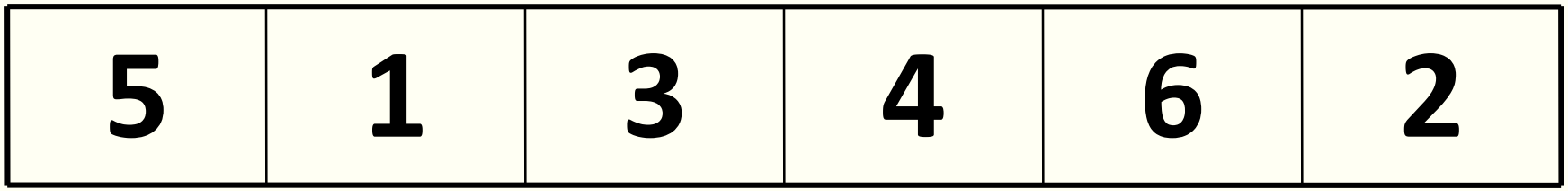
- Độ phức tạp của thuật toán: $O(n^2)$

Sắp xếp chèn

- Ý tưởng: Xem dãy cần sắp xếp gồm 2 dãy:
  - Đích: Các phần tử đã sắp đúng vị trí
  - Nguồn: Các phần từ còn lại của dãy
  - Trong mỗi bước lặp,chèn lần lượt các phần tử ở dãy nguồn vào dãy đích để được một dãy đích được sắp

- Cách thực hiện :
- Khởi tạo: dãy đích gồm phần tử a_1, dãy nguồn gồm các phần tử $a_2,.....,a_n$
- Lặp n-1 bước, chèn lần lượt các phần tử $a_2,.....,a_n$ vào đúng vị trí trong dãy đích.

Ví dụ minh họa

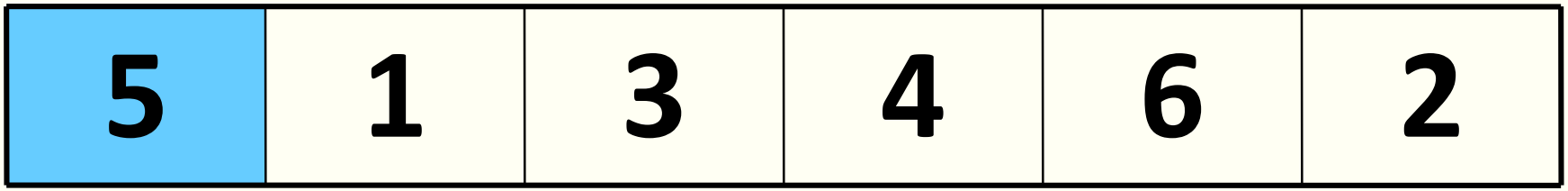


So sánh

Chuyển dữ liệu

Đã sắp xếp

Ví dụ minh họa

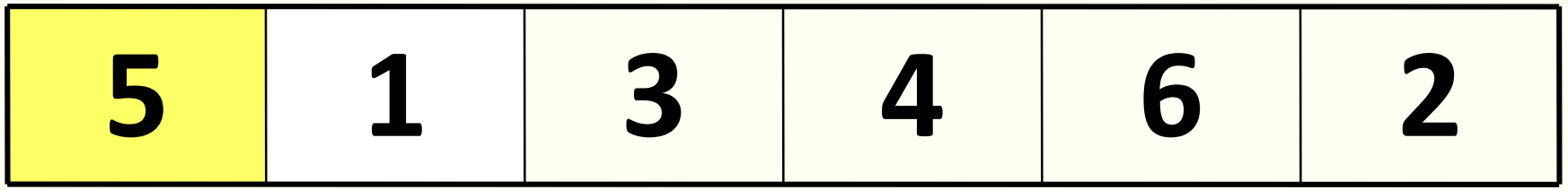


So sánh

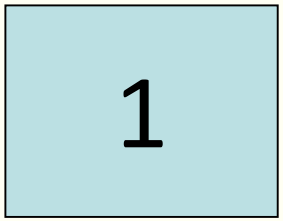
Chuyển dữ liệu

Đã sắp xếp

Ví dụ minh họa



Temp

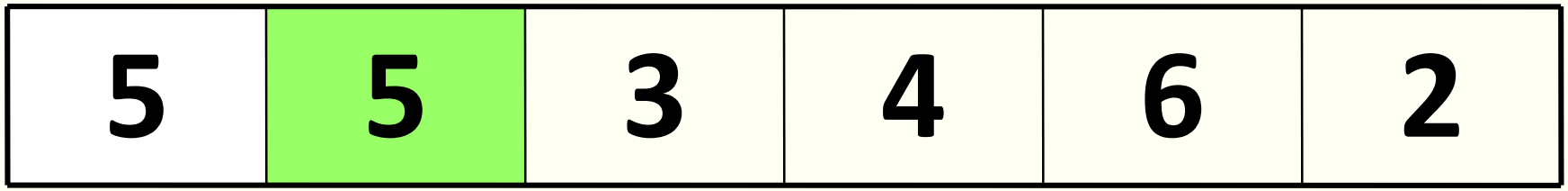


So sánh

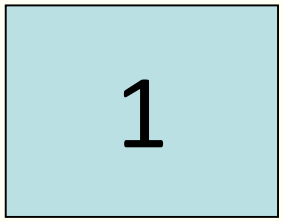
Chuyển dữ liệu

Đã sắp xếp

Ví dụ minh họa



Temp

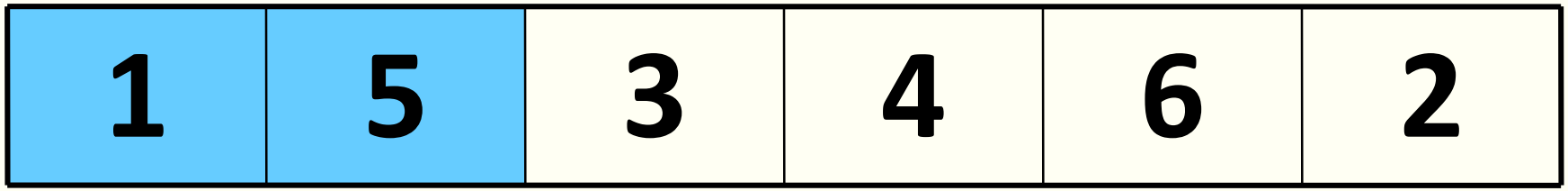


So sánh

Chuyển dữ liệu

Đã sắp xếp

Ví dụ minh họa

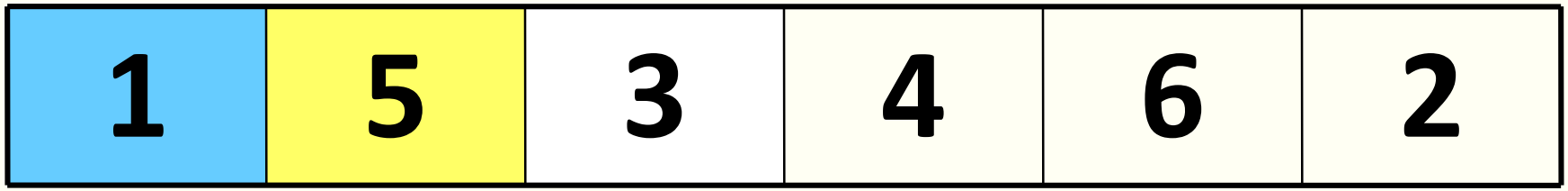


So sánh

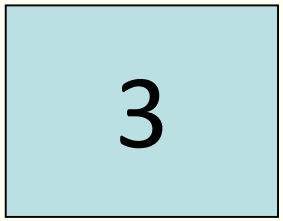
Chuyển dữ liệu

Đã sắp xếp

Ví dụ minh họa



Temp

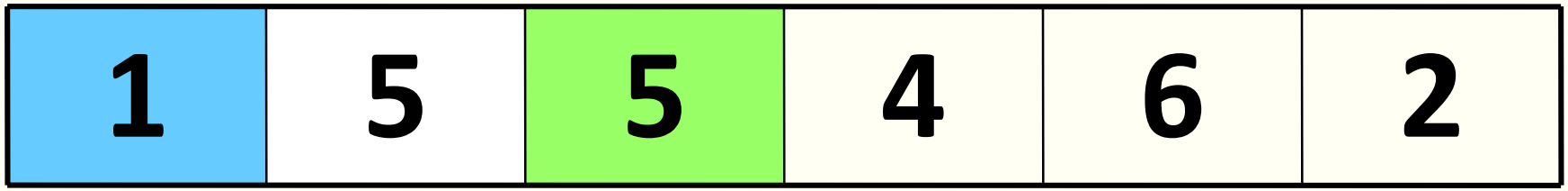


So sánh

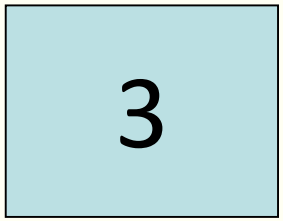
Chuyển dữ liệu

Đã sắp xếp

Ví dụ minh họa



Temp

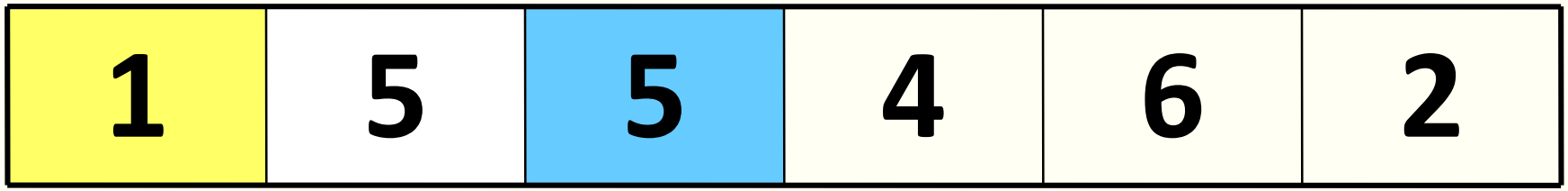


So sánh

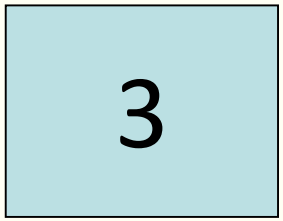
Chuyển dữ liệu

Đã sắp xếp

Ví dụ minh họa



Temp

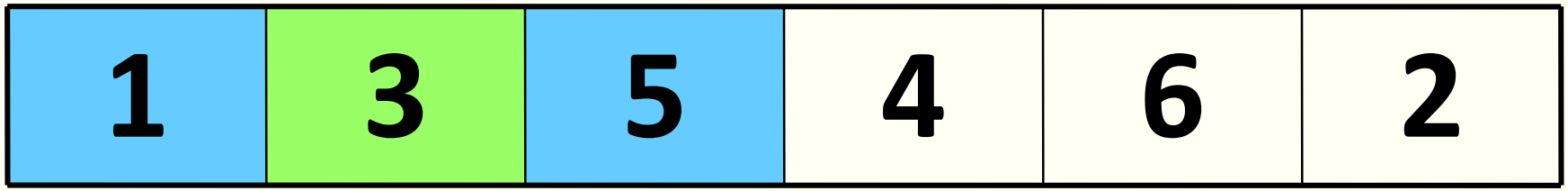


So sánh

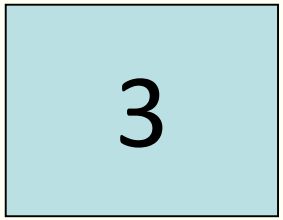
Chuyển dữ liệu

Đã sắp xếp

Ví dụ minh họa



Temp

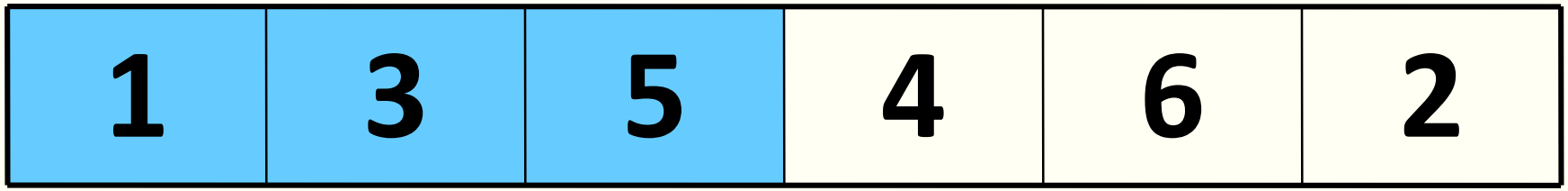


So sánh

Chuyển dữ liệu

Đã sắp xếp

Ví dụ minh họa

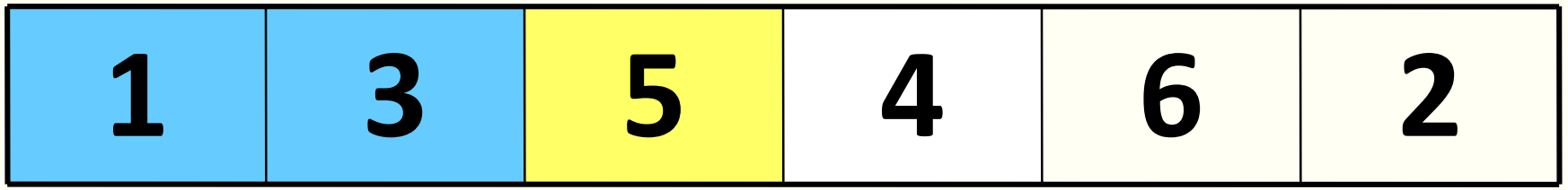


So sánh

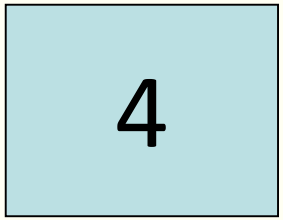
Chuyển dữ liệu

Đã sắp xếp

Ví dụ minh họa



Temp



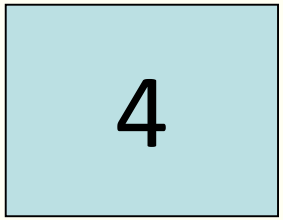
So sánh

Chuyển dữ liệu

Đã sắp xếp

Ví dụ minh họa

Temp

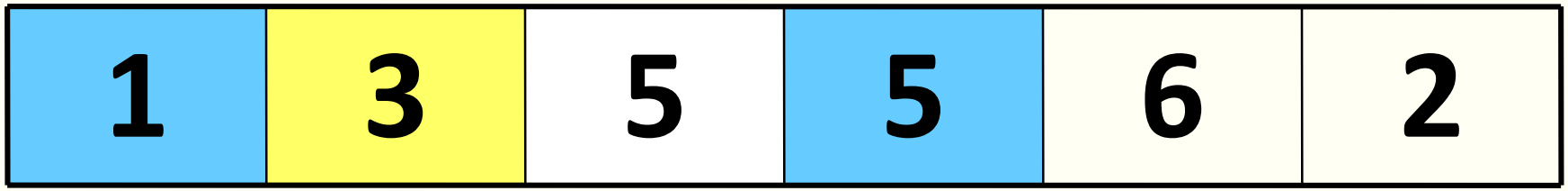


So sánh

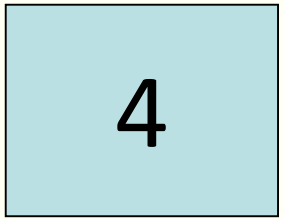
Chuyển dữ liệu

Đã sắp xếp

Ví dụ minh họa



Temp

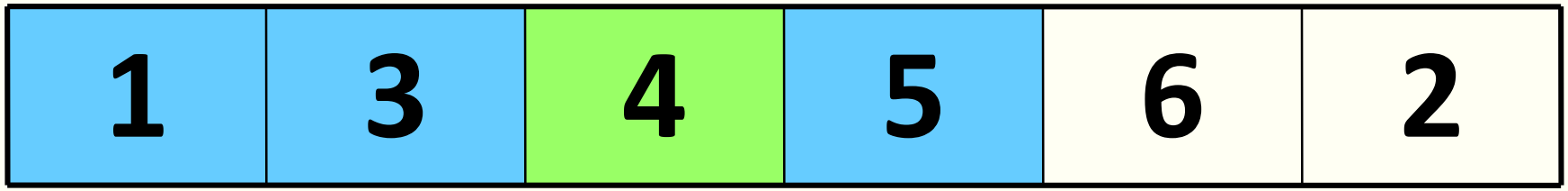


So sánh

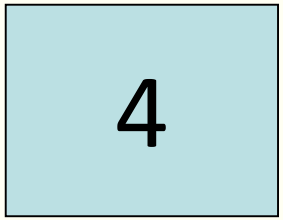
Chuyển dữ liệu

Đã sắp xếp

Ví dụ minh họa



Temp

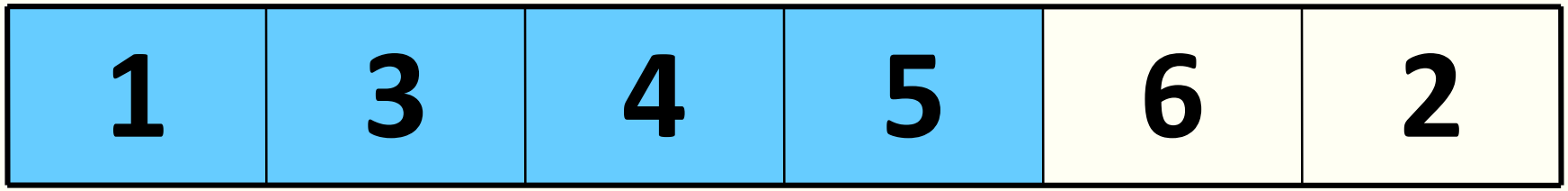


So sánh

Chuyển dữ liệu

Đã sắp xếp

Ví dụ minh họa

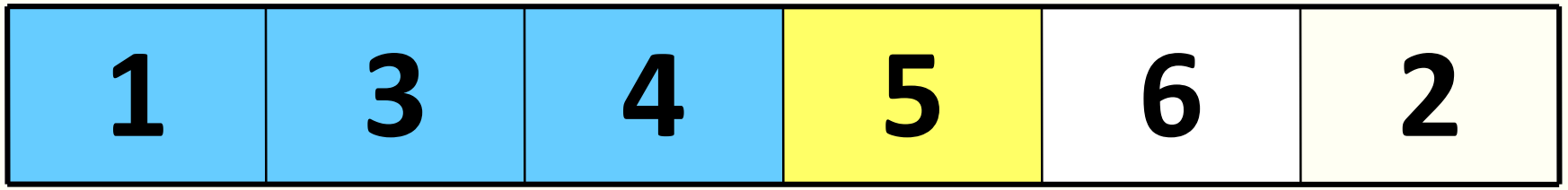


So sánh

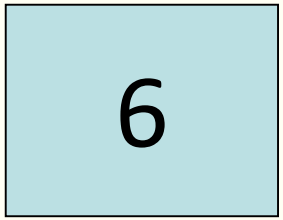
Chuyển dữ liệu

Đã sắp xếp

Ví dụ minh họa



Temp

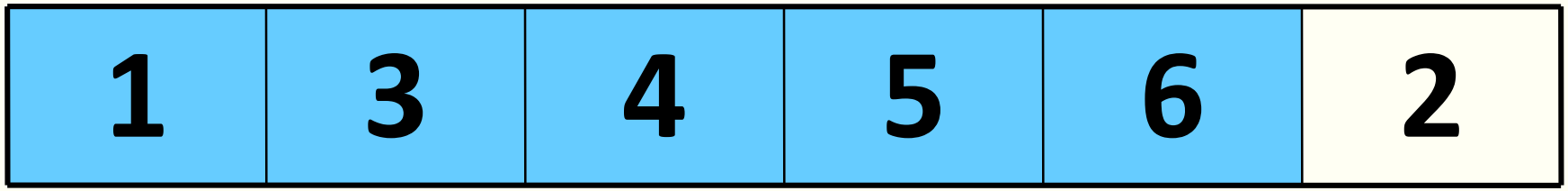


So sánh

Chuyển dữ liệu

Đã sắp xếp

Ví dụ minh họa

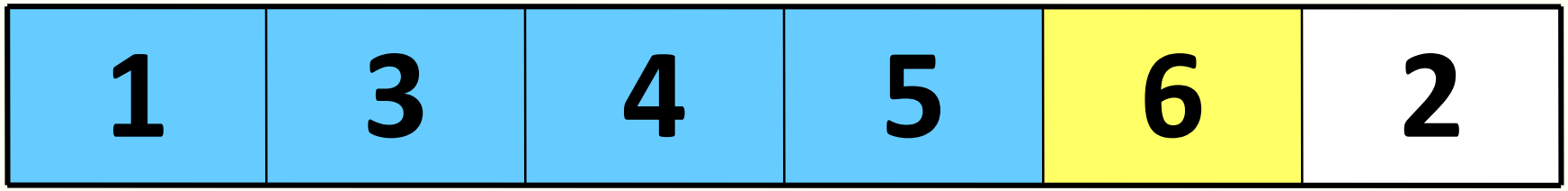


So sánh

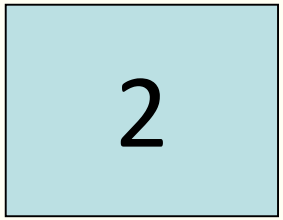
Chuyển dữ liệu

Đã sắp xếp

Ví dụ minh họa



Temp

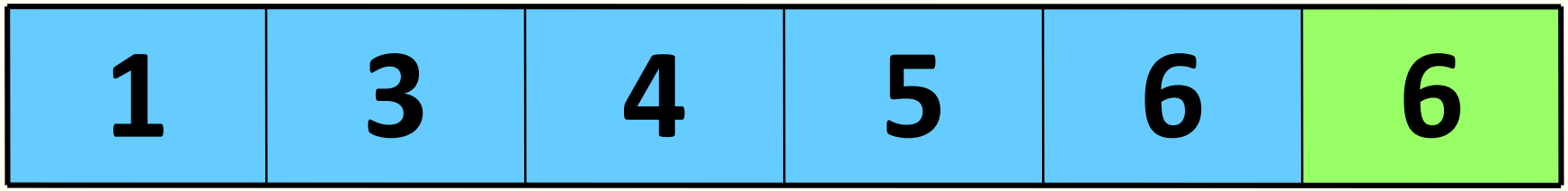


So sánh

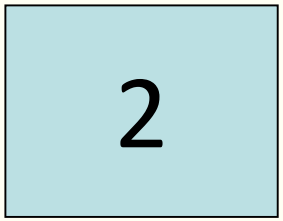
Chuyển dữ liệu

Đã sắp xếp

Ví dụ minh họa



Temp

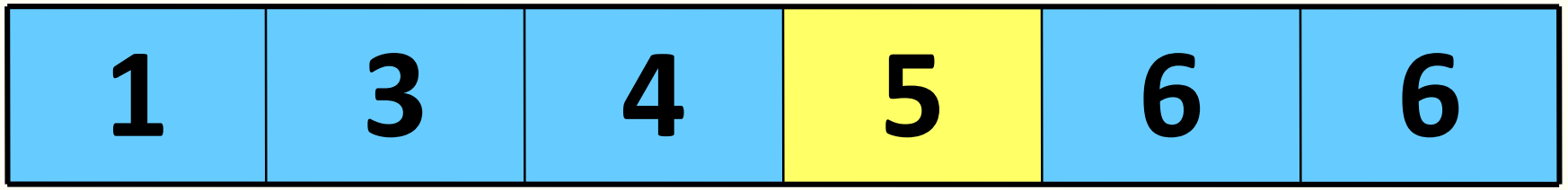


So sánh

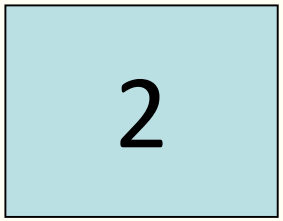
Chuyển dữ liệu

Đã sắp xếp

Ví dụ minh họa



Temp

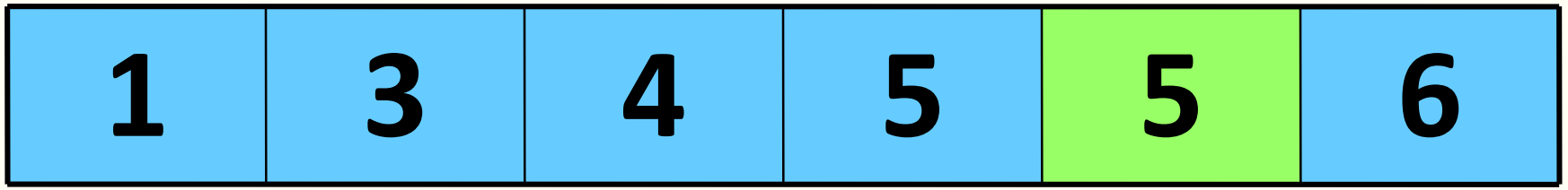


So sánh

Chuyển dữ liệu

Đã sắp xếp

Ví dụ minh họa



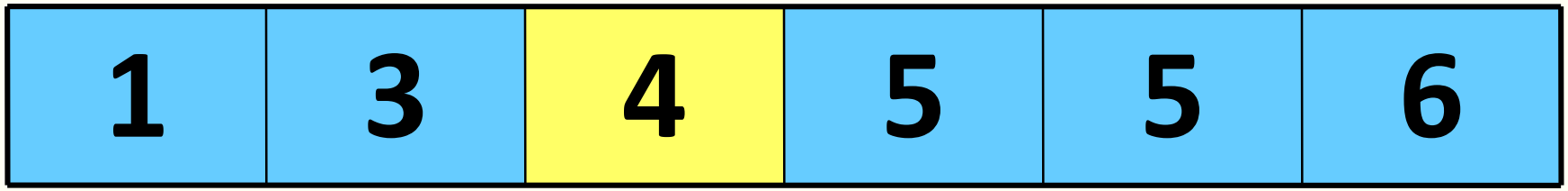
Temp 2

So sánh

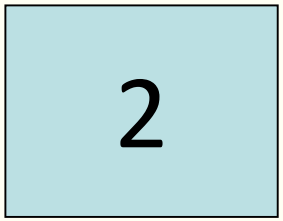
Chuyển dữ liệu

Đã sắp xếp

Ví dụ minh họa



Temp

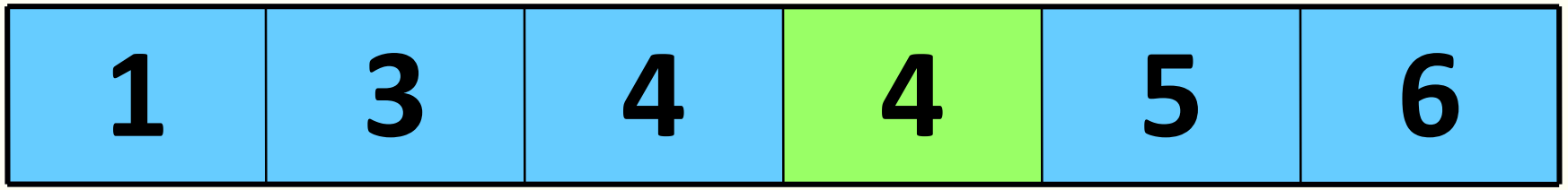


So sánh

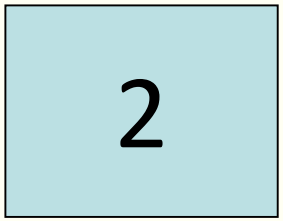
Chuyển dữ liệu

Đã sắp xếp

Ví dụ minh họa



Temp

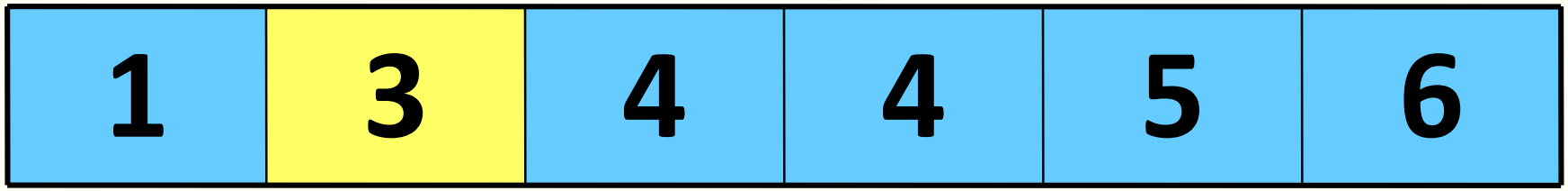


So sánh

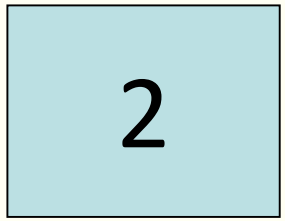
Chuyển dữ liệu

Đã sắp xếp

Ví dụ minh họa



Temp

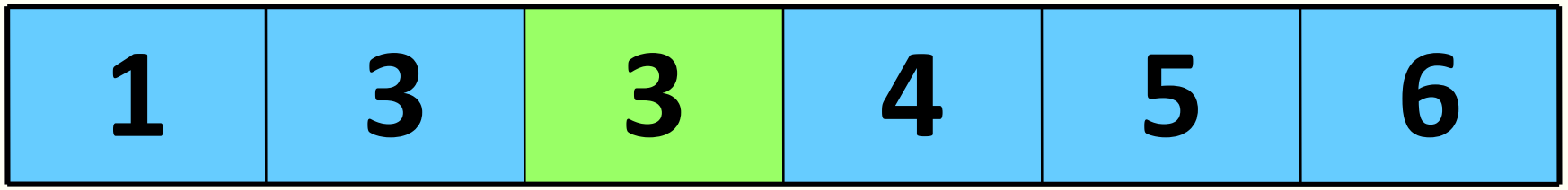


So sánh

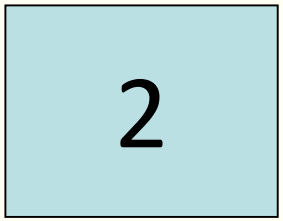
Chuyển dữ liệu

Đã sắp xếp

Ví dụ minh họa



Temp

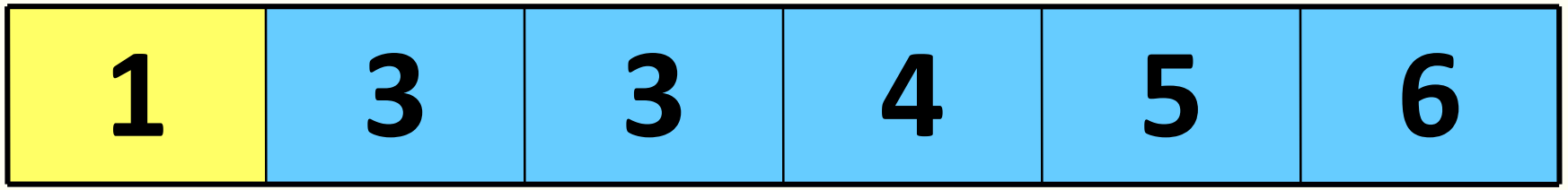


So sánh

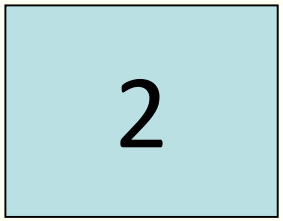
Chuyển dữ liệu

Đã sắp xếp

Ví dụ minh họa



Temp

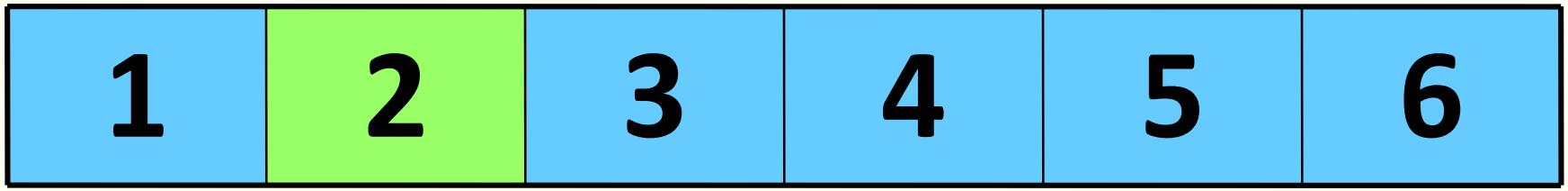


So sánh

Chuyển dữ liệu

Đã sắp xếp

Ví dụ minh họa



KẾT THÚC!

So sánh

Chuyển dữ liệu

Đã sắp xếp

Thuật toán

```
void InsertionSort(int a[],int n){
     for(int i=2;i<=n;i++){
          int j=i-1;
          bool ok=false;
          int temp=a[i];

          while ((j>=1)&&!ok){
                if (a[j]>temp) {a[j+1]=a[j];j--;}
                else ok=true;
          }
          a[j+1]=temp;

     }
}
```

- Độ phức tạp của thuật toán: $O(n^2)$

Tổng kết

- Các phương pháp sắp xếp đơn giản có độ phức tạp $O(n^2)$.
- Kích thước dữ liệu nhỏ, dễ cài đặt

Tiếp theo…

- Sắp xếp nhanh