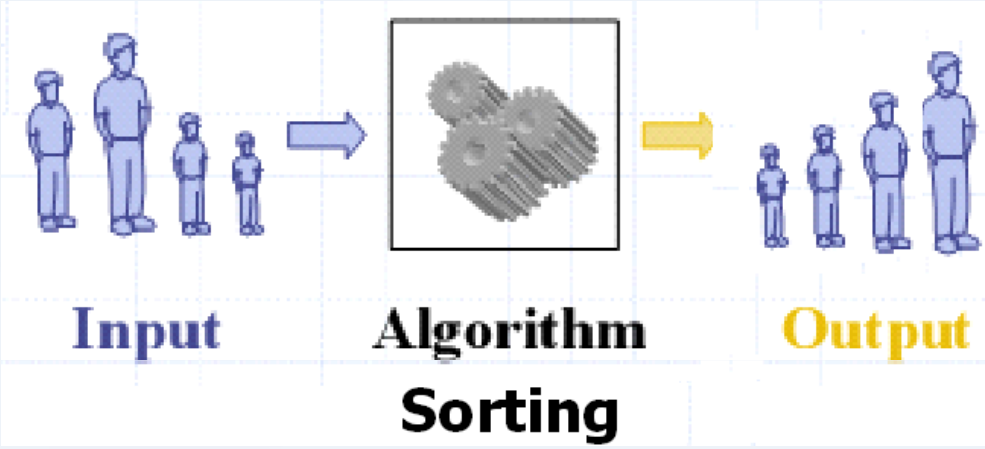
Bài 11: Sắp xếp (Sorting)



Sorting

Bài toán

- **Input:**
  - Dãy các phần tử (và một thứ tự)
  - (Dãy các phần tử thường được lưu bằng mảng.)
- **Output:**
  - Dãy các phần tử được sắp theo thứ tự tăng hoặc giảm dần theo một hoặc một vài thuộc tính của nó (các thuộc tính này gọi là thuộc **tính khóa**).
  - Thuộc tính khóa được sắp xếp theo một hàm logic, ví dụ (<=) hoặc các toán tử so sánh khác.

Các thuật toán sắp xếp nội với thời gian chạy $O(n^2)$

- Nổi bọt – Bubble sort
- Chèn – Insertion sort
- Chọn – Selection sort

Sắp xếp nổi bọt – Bubble sort

Ý tưởng:

Thực hiện chuyển dần các phân tử có giá trị khóa nhỏ về đầu dãy, các phần tử có khóa lớn về cuối dãy.

Sắp xếp nổi bọt – Bubble sort

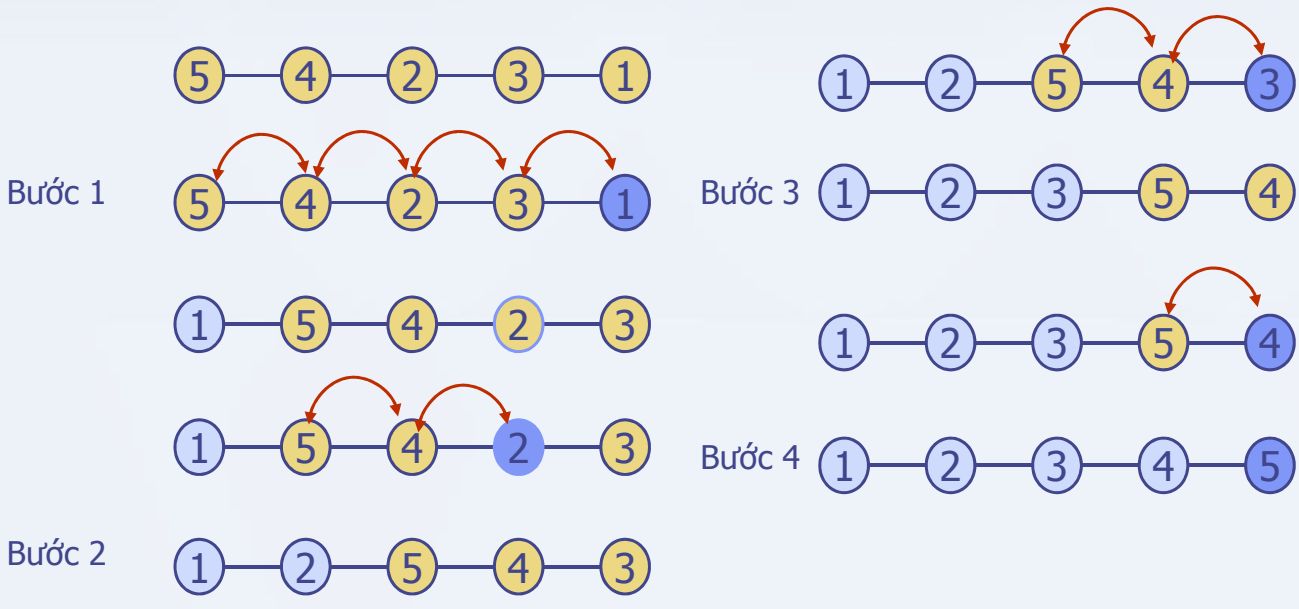
Giải thuật:

❖ Đi từ cuối về đầu mảng, trong quá trình đi nếu phần tử ở dưới (sau) **<** phần tử đứng ngay trên (trước) nó thì phần tử nhẹ sẽ bị **"trồi"** lên phía trên phần tử nặng (đổi chỗ). Kết quả là phần tử nhỏ nhất (nhẹ nhất) sẽ được đưa lên (trồi lên) trên bề mặt (đầu mảng) rất nhanh.

❖ Sau mỗi lần đi chúng ta đưa được một phần tử trồi lên đúng chỗ. Do vậy, sau N–1 lần đi thì tất cả các phần tử trong mảng A sẽ có thứ tự tăng.

Sắp xếp nổi bọt – Bubble sort

Ví dụ sắp xếp dãy sau theo thứ tự tăng dần:



Thuật toán

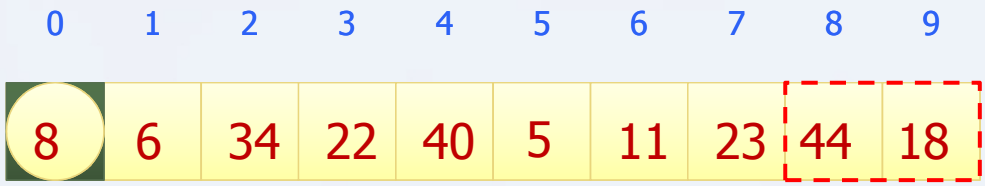
Algorithm ***BubbleSort(Array A, n)***
Input: Mảng A có n phần tử
Output: Mảng A được sắp theo thứ tự tăng dần của khóa

```
for i ← 0 to n-2 do
    for j ← n-1 downto i+1 do
        if A[j].Key < A[j-1].Key then
            swap(A[j-1], A[j]);
```

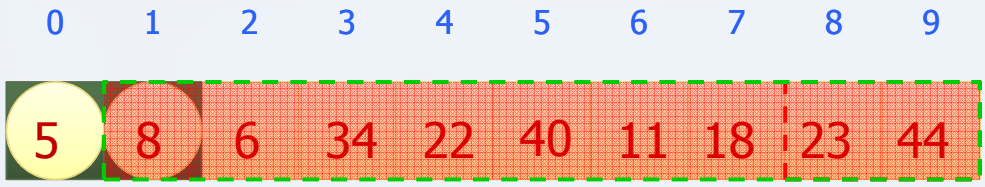
- Trong đó swap là thủ tục hoán vị hai phần tử

```
void Swap(object &a, object &b){
        object tg;
        tg = a;  a = b;   b = tg;
}
```


Minh họa thuật toán Bubble sort

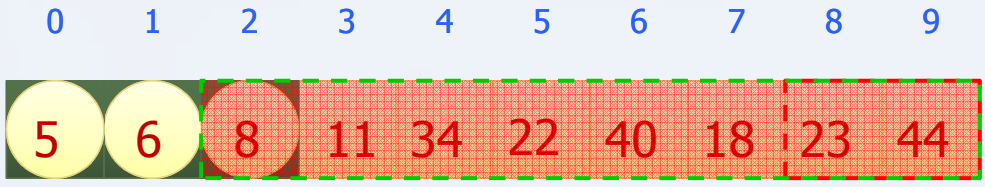


Minh họa thuật toán Bubble sort



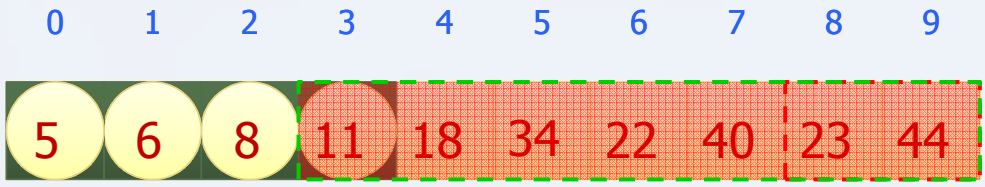
Temp

Minh họa thuật toán Bubble sort



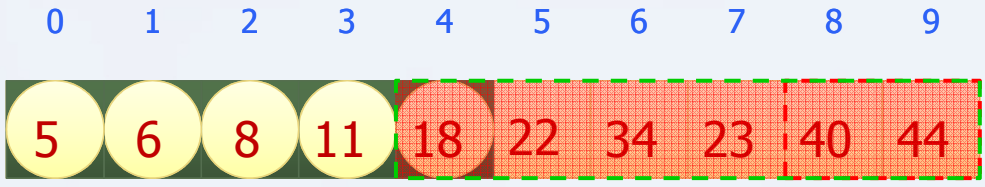
Temp

Minh họa thuật toán Bubble sort



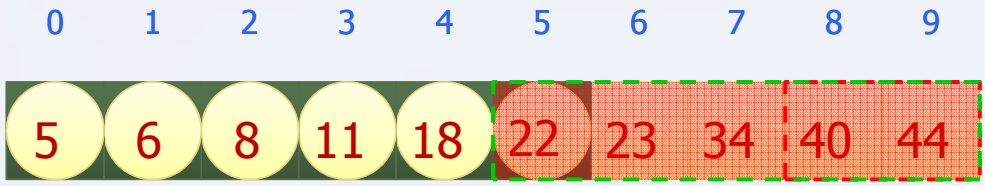
Temp

Minh họa thuật toán Bubble sort



Temp

Minh họa thuật toán Bubble sort



Temp

Minh họa thuật toán Bubble sort

Temp

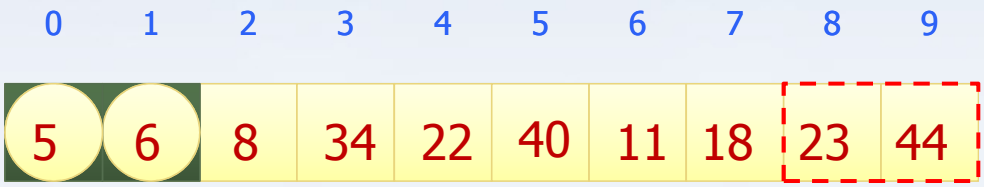
Minh họa thuật toán Bubble sort

Temp

Minh họa thuật toán Bubble sort

Temp

Cài đặt thuật toán Bubble sort




```
void BubbleSort ( int A[], int n){
    int i, j, t;
    for (i = 0; i < n-1;  i++)
        for (j = n-1; j > i; j--){
          if (A[j] < A[j-1]){
              t=A[j];
              A[j]=A[j-1];
              A[j-1]=t;
          }
        }
}
```

Chứng minh thời gian chạy của thuật toán trong trường hợp xấu nhất là $O(n^2)$

?

Thời gian chạy

Algorithm ***BubbleSort(Array A, n)***
Input: Mảng A có n phần tử
Output: Mảng A được sắp theo thứ tự tăng dần của khóa

```
for i ← 1 to n-1 do                          n+2
    for j ← n downto i+1 do                  n-i+2
        if A[j].Key < A[j-1].Key then        4
            swap(A[j-1], A[j]);              6
```

- Thời gian chạy:

$T(n) = (n+2) + (n-1)*2 + 10*[(n-1) + (n-2) + ..2+1]$

- Thời gian chạy của thuật toán là $O(n^2)$

Ví dụ:

Mô tả quá trình sắp xếp của dãy số

12 43 11 34 23 43

Sắp xếp chọn - Selection sort

- **Ý tưởng:**

Chọn phần tử có khóa nhỏ nhất trong các phần tử còn lại chuyển nó về đầu và loại bỏ nó khỏi dãy.

Sắp xếp chọn - Selection sort

- **Giải thuật:**

❖ Ta chọn phần tử có giá trị nhỏ nhất trong N phần tử chưa có thứ tự này để đưa lên đầu nhóm.

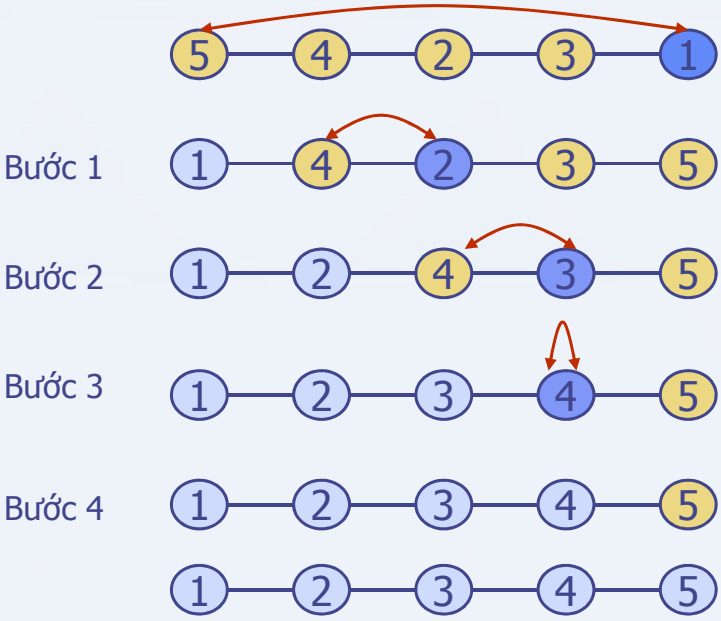
❖ Sau lần thứ nhất ta còn lại N-1 phần tử đứng ở phía sau dãy A chưa có thứ tự.

❖ Chúng ta tiếp tục chọn phần tử có giá trị nhỏ nhất trong N-1 phần tử này để đưa lên đầu nhóm

❖ Làm tiếp tục cho đến cuối dãy

Sắp xếp chọn - Selection sort

- Ví dụ sắp xếp dãy sau theo thứ tự tăng dần:

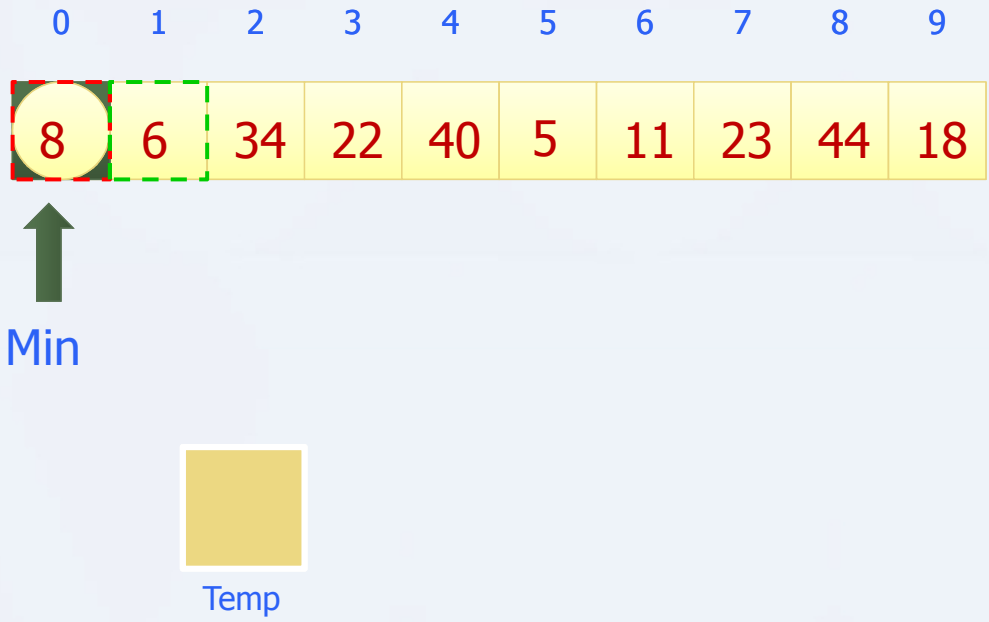


Thuật toán

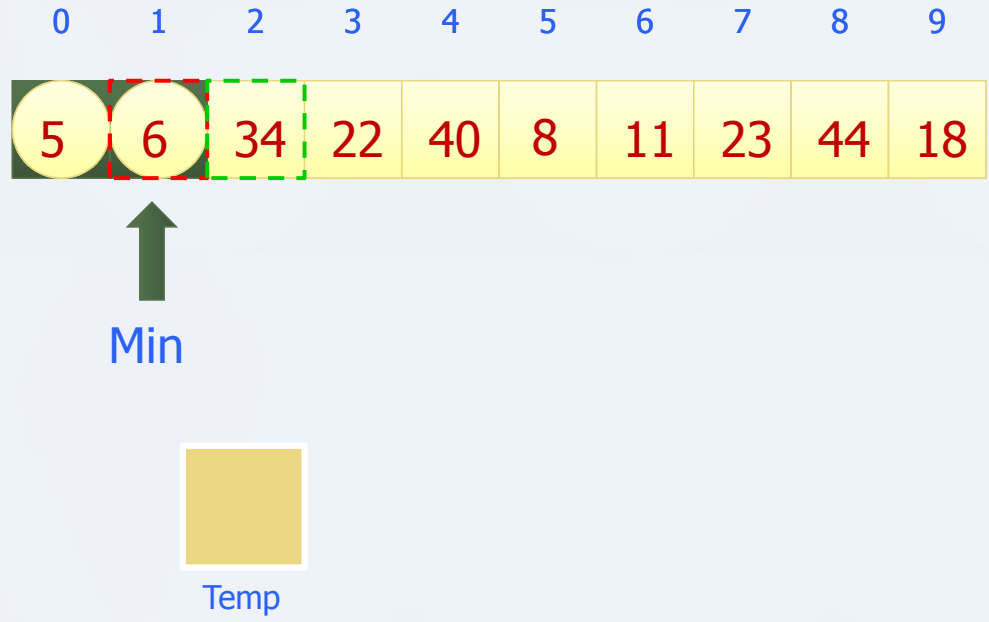
Algorithm ***SelectionSort(Array A, n)***
Input: Mảng A có n phần tử
Output: Mảng A được sắp theo thứ tự tăng dần của khóa

```
for i ← 0 to n-2  do
        posmin ← i ;
        for j ← i+1 to n-1 do
            if A[posmin].Key > A[j].Key then
                posmin ← j ;
        if posmin ≠ i then
                swap (A[i], A[posmin]);
```

Minh họa thuật toán Selection sort

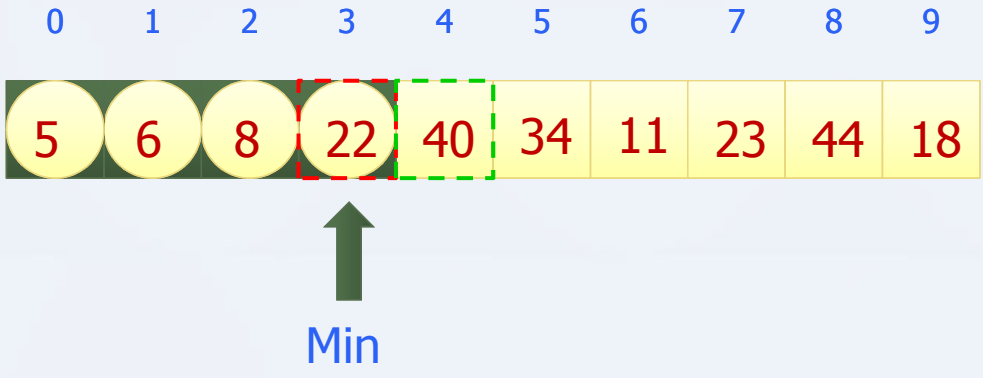


Minh họa thuật toán Selection sort

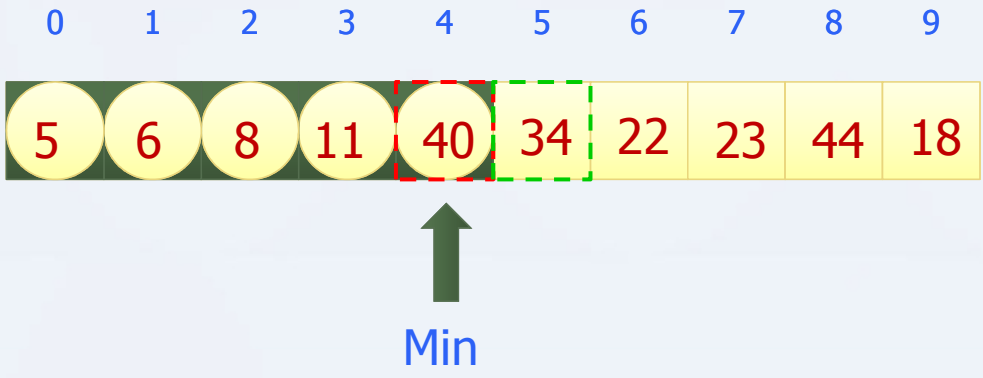


Minh họa thuật toán Selection sort

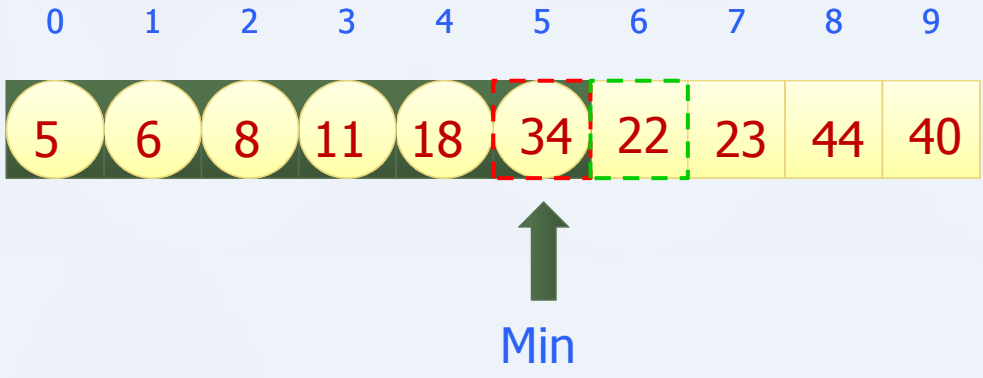
Minh họa thuật toán Selection sort



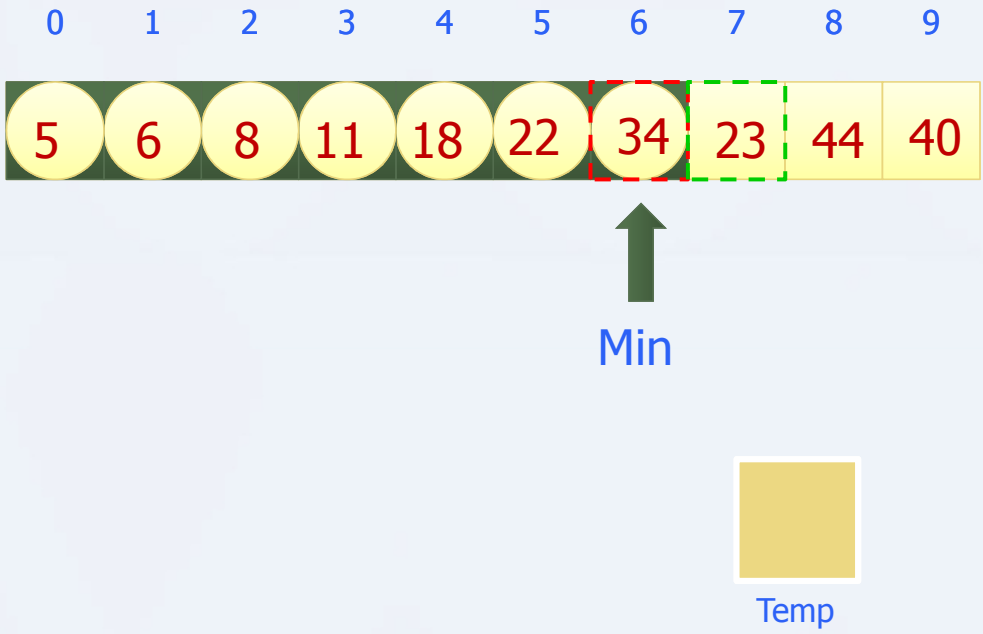
Minh họa thuật toán Selection sort



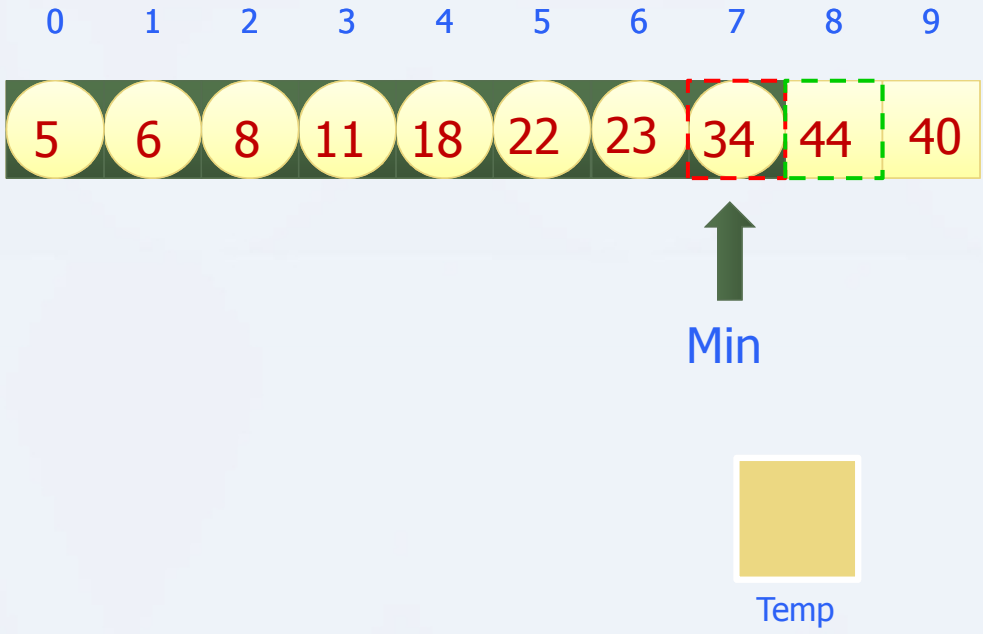
Minh họa thuật toán Selection sort



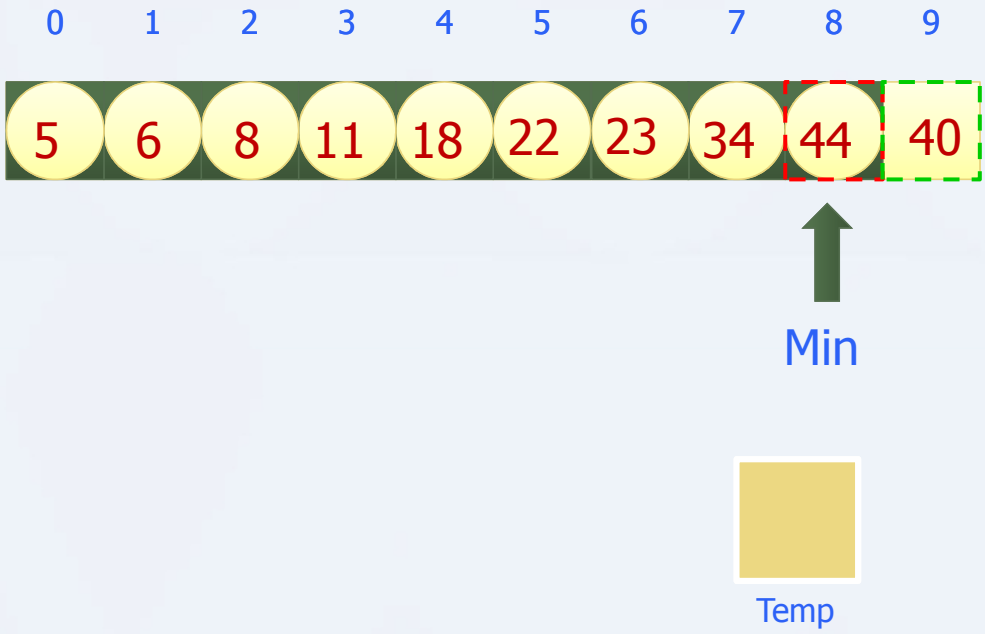
Minh họa thuật toán Selection sort



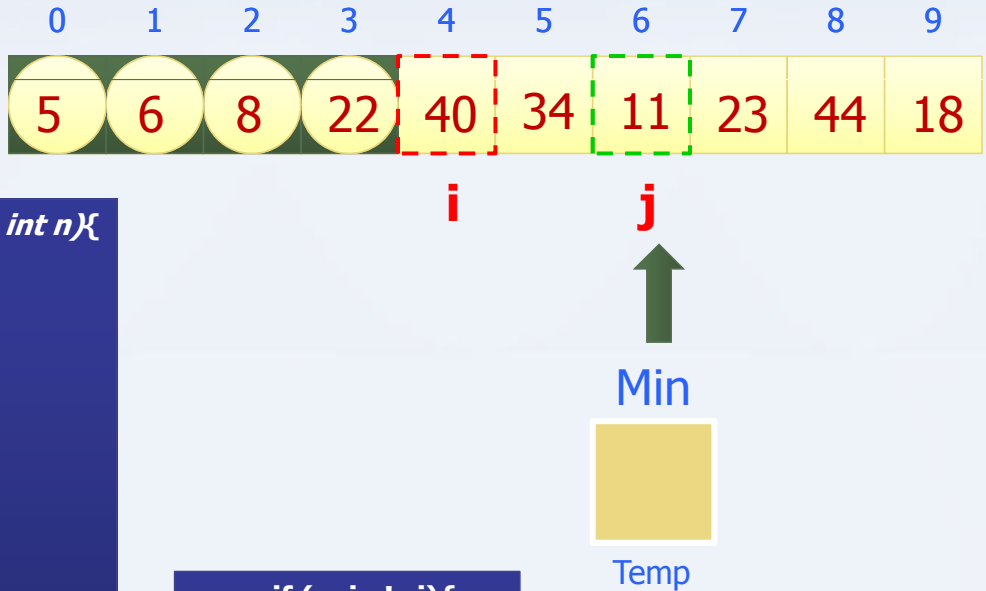
Minh họa thuật toán Selection sort



Minh họa thuật toán Selection sort



Cài đặt thuật toán Selection sort




```
void SelectionSort ( int  A[], int n){
    int i, j, min, t;

    for(i=0;i<n-1;i++){
        min = i;
        for(j=i+1;j<n;j++) {

            if (A[j] <A[min])
                min =j;
        }

        t=A[i];
        A[i]=A[min];
        A[min]=t;

    }
}
```

```
if ( min !=i){
    t=A[i];
    A[i]=A[min];
    A[min]=t;
}
```

Chứng minh thời gian chạy của thuật toán trong trường hợp xấu nhất là $O(n^2)$

?

Thời gian chạy

```
for i ← 1 to n-1  do                                  n+2
        posmin ← i ;                                  n-1
        for j ← i+1 to n do                           (n-i +2)*(n-1)
            if A[posmin].Key > A[j].Key then          3
                posmin ← j ;                          1
        if posmin ≠ i then                            n-1
                swap (A[i], A[posmin]);               6(n-1)
```

Thời gian chạy của thuật toán

$T(n) = (n+2) + 4* [(n-1)+(n-2)+..+1] + 10*(n-1)$

Thời gian chạy của thuật toán là $O(n^2)$

Ví dụ:

Mô tả quá trình sắp xếp của dãy số

12 43 11 34 23 435

Sắp xếp chèn – Insertion sort

Ý tưởng:

❖ Bắt chước cách sắp xếp quân bài khi chơi bài. Muốn sắp một bộ bài theo trật tự người chơi bài rút lần lượt từ quân thứ 2, so với các quân đứng trước nó để chèn vào vị trí thích hợp.

Sắp xếp chèn – Insertion sort

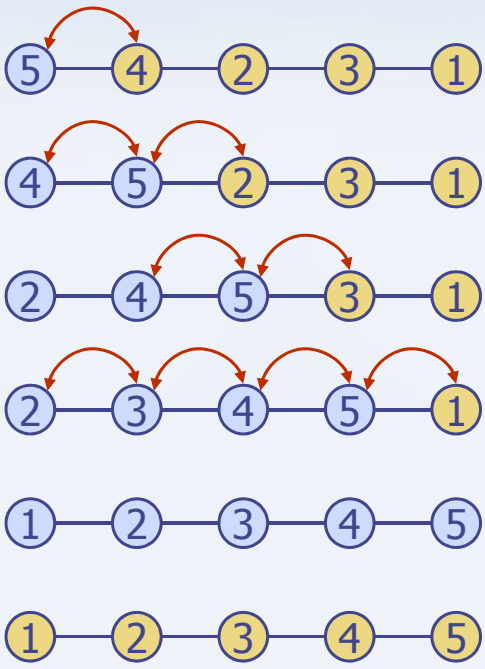
Giải thuật:

- Đi từ đầu dãy đến cuối dãy, lần lượt lấy các phần tử của dãy chèn vào vị trí thích hợp trong một dãy mới đã được sắp.

- Lấy phần tử thứ A[j] chèn vào dãy gồm các phần tử từ A[1]..A[j-1] sao cho ta được dãy A[1]..A[j] được sắp. Trong đó dãy A[1]..A[j-1] là dãy đã được sắp.

Sắp xếp chèn – Insertion sort

Ví dụ sắp xếp dãy sau theo thứ tự tăng dần:

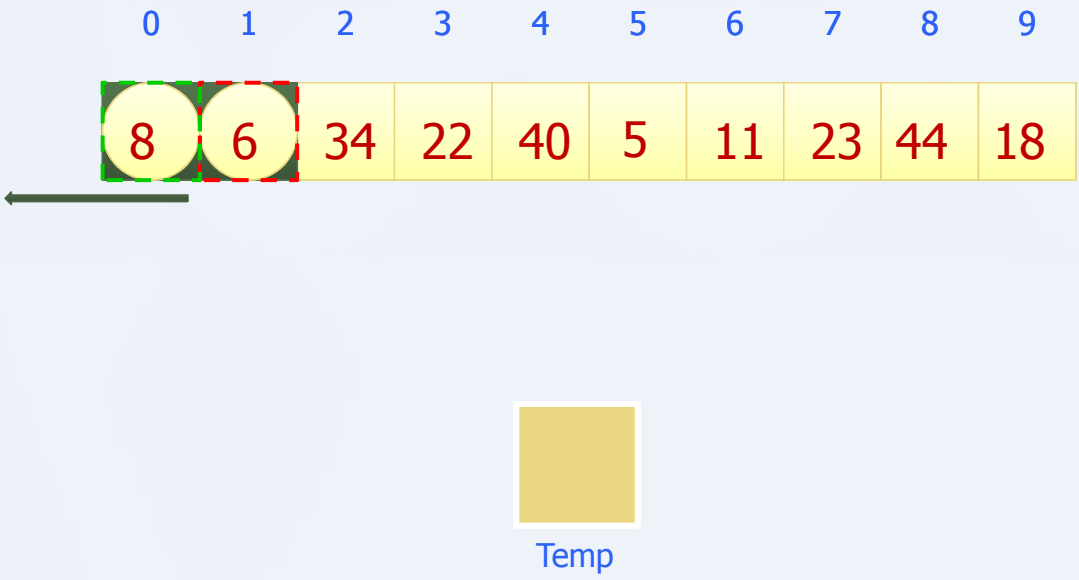


Thuật toán

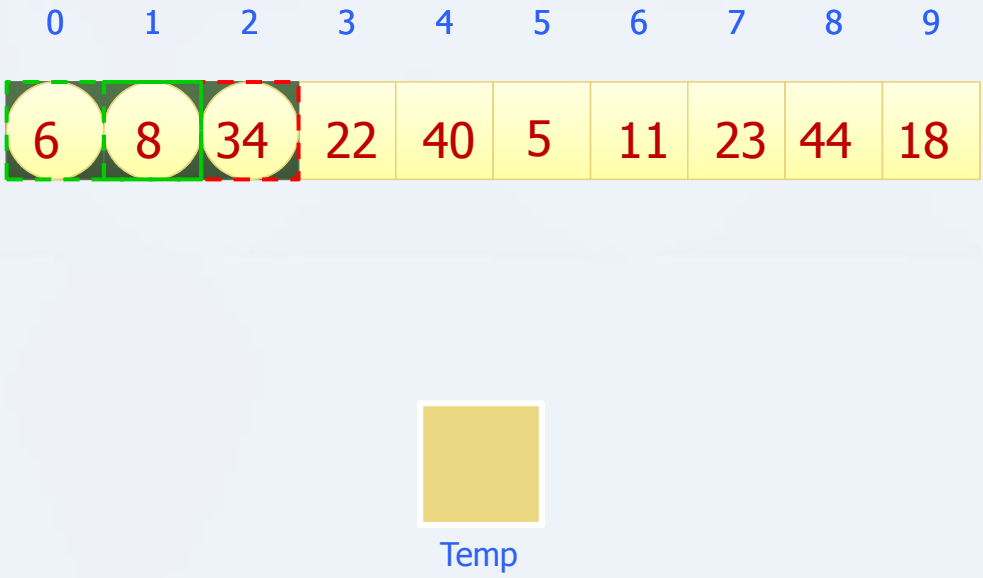
Algorithm ***InsertionSort(Array A, n)***
Input: Mảng A có n phần tử
Output: Mảng A được sắp theo thứ tự tăng dần của khóa

```
for i ← 1 to n-1 do
    j ← i-1;
    x ← A[i];
    while (A[j].Key>x.Key) and (j>=0) do
        A[j+1] ← A[j];
        j ← j-1;
    A[j+1] ← x;
```

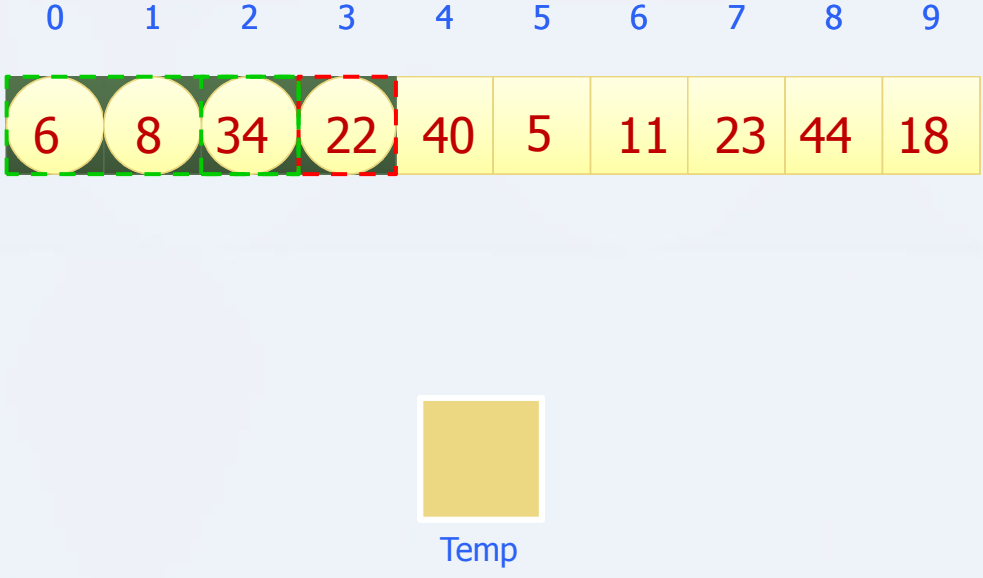

Minh họa thuật toán Insertion sort



Minh họa thuật toán Insertion sort

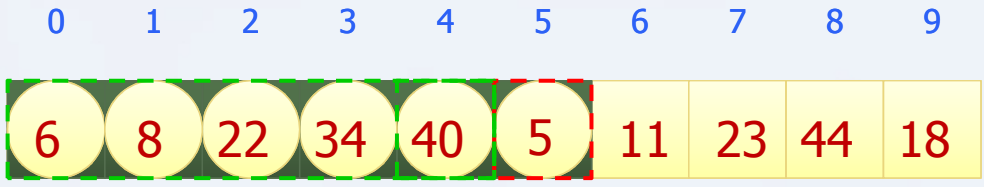


Minh họa thuật toán Insertion sort

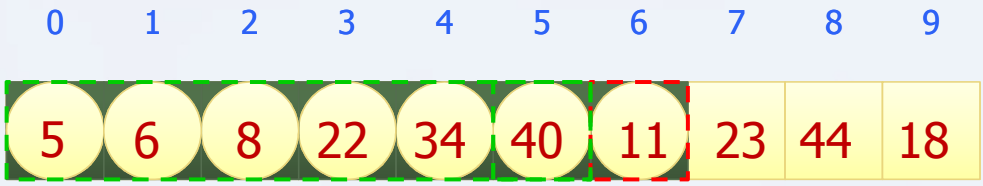


Minh họa thuật toán Insertion sort

Minh họa thuật toán Insertion sort



Minh họa thuật toán Insertion sort

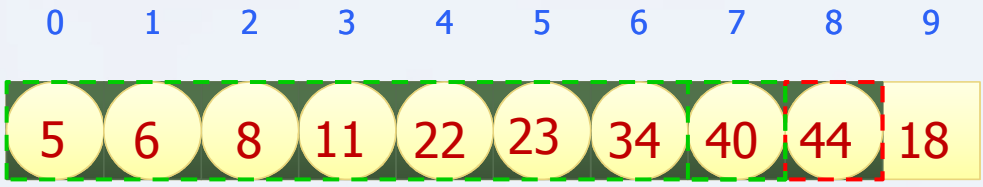


Temp

Minh họa thuật toán Insertion sort

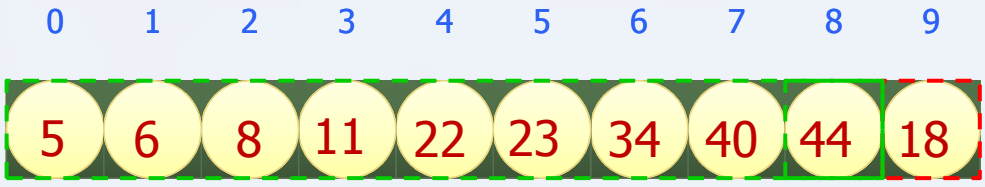
Temp

Minh họa thuật toán Insertion sort



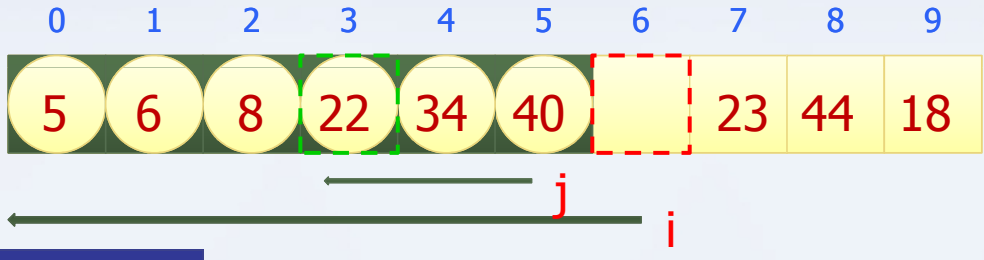
Temp

Minh họa thuật toán Insertion sort



Temp

Cài đặt thuật toán Insertion sort




```
void InsertionSort( int A[], int n){
      int i, j, x;
      for(i=1;i<n;i++){
          x=A[i];
          j=i-1;
          while (x<A[j]&& j>=0){
             A[j+1]=A[j];
             j--;
          }
          A[j+1]=x;
       }
}
```

Chứng minh thời gian chạy của thuật toán trong trường hợp xấu nhất là $O(n^2)$

?

Ví dụ:

Mô tả quá trình sắp xếp của dãy số

12 43 11 34 23 43 12 435