Sorting Algorithms

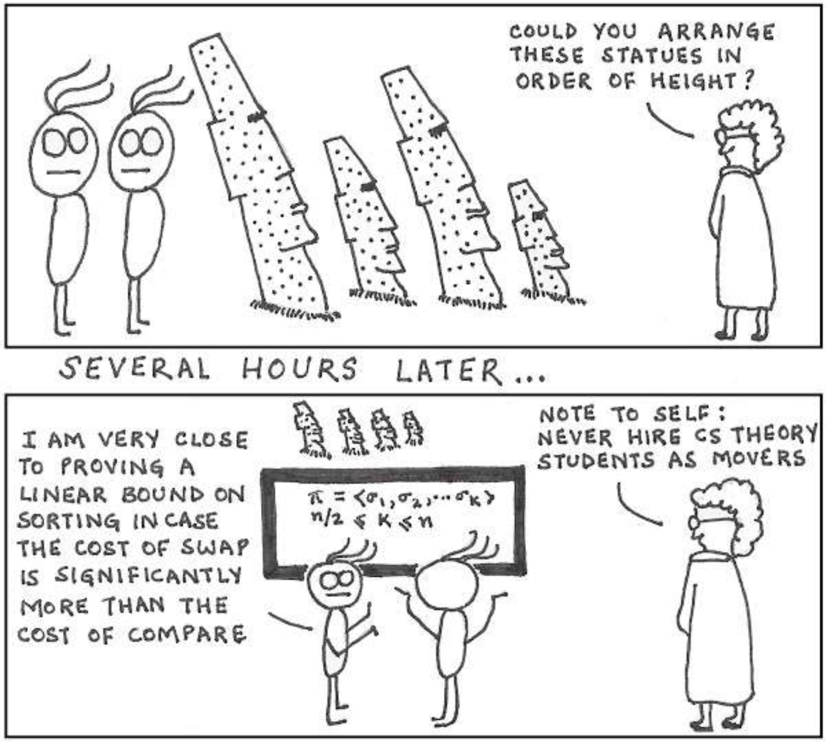
Design & Analysis

Sorting Problem

- Sorting Problem
  - Input: A sequence of n numbers(a1,a2,...an)
  - Output: A permutation (reordering)(a1',a2',...an') of the input sequence such that a1'<=a2'<=...an
- Efficiency and Passes
  - Efficiency denotes how much time the algorithm takes to sort the elements
    - Efficiency of sorting algorithm is measured in terms of time complexity –big-Oh notations
    - $O(n^2)$, $O(n\log n)$
  - Passes -The phases in which elements are moving to acquire their proper position

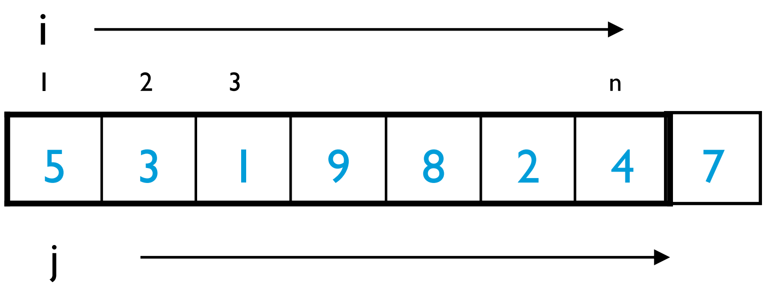
Analysis of:

- Sorting Algorithms
  - Bubble Sort
  - Selection Sort
  - Insertion Sort



Bubble Sort

- Idea:
  - Repeatedly pass through the array
  - Swaps adjacent elements that are out of order



- Easier to implement, but slower than Insertion sort

Example

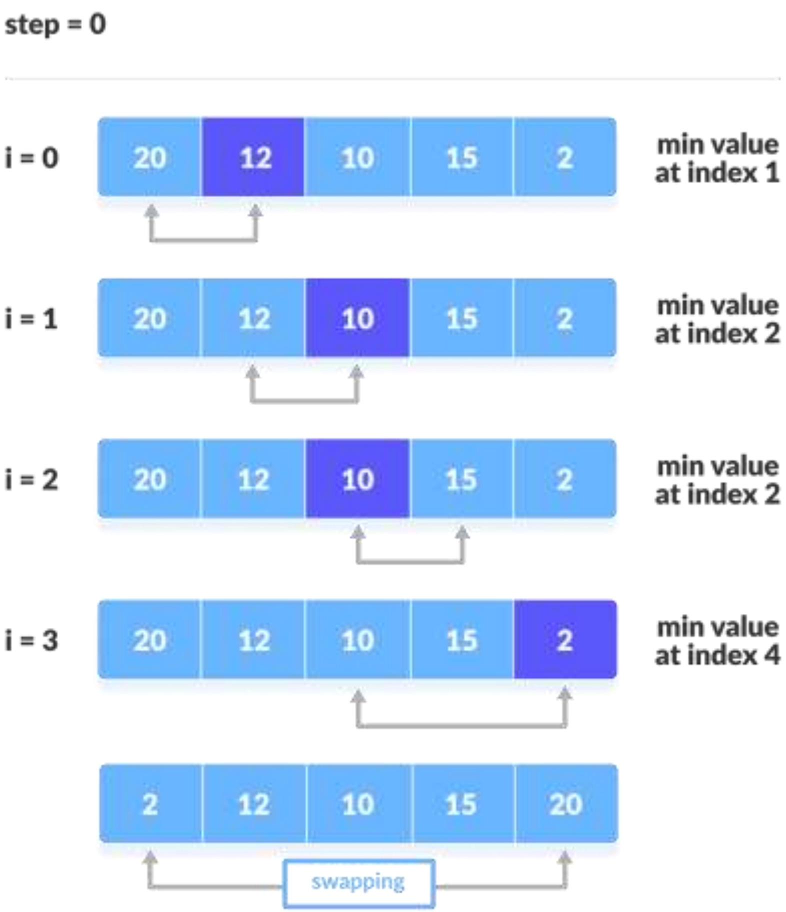
Bubble Sort

```
for(i=0;i<n-1;i++)
{
 for(j=0;j<n-i-1;j++)
 {
  if (a[j]>a[j+1])
      {
       temp=a[j];
       a[j]=a[j+1];
       a[j+1]=temp;
      }
  }
 }
```

Selection Sort

- Idea:
  - Find the smallest element in the array
  - Exchange it with the element in the first position
  - Find the second smallest element and exchange it with the element in the second position
  - Continue until the array is sorted
- Disadvantage:
  - Running time depends only slightly on the amount of order in the file

Example



step = 0
i = 0
20
12
10
15
2
min value at index 1
i = 1
20
12
10
15
2
min value at index 2
i = 2
20
12
10
15
2
min value at index 2
i = 3
20
12
10
15
2
min value at index 4
2
12
10
15
20
swapping

Illustration

5 3 4 1 2

Selection Sort

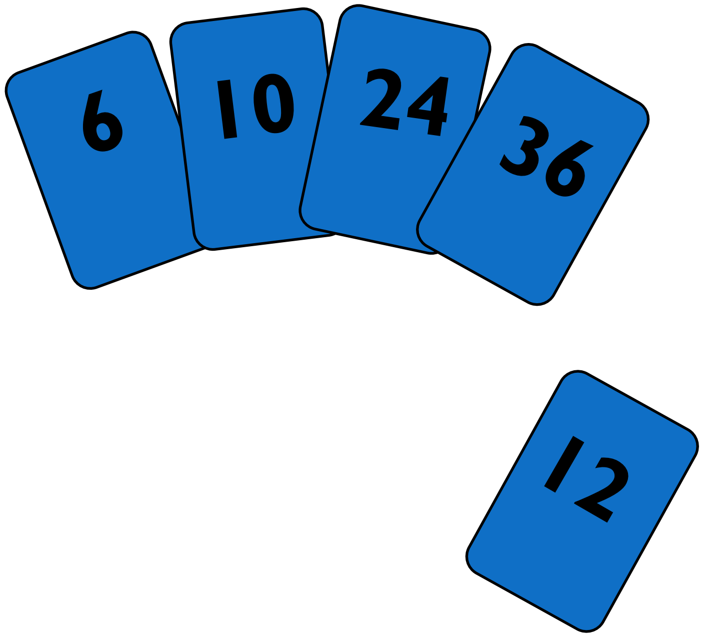
Selection Sort

```
for(i=0;i<n;i++)
{
        min=a[i];
        pos=i;
        for(j=i+1;j<n;j++)
        {
                if(a[j]<min)
                {
                        min=a[j];
                        pos=j;
                }
        }
        if (pos!=i)
        {
                temp=a[i];
                a[i]=a[pos];
                a[pos]=temp;
        }
}
```

Insertion Sort

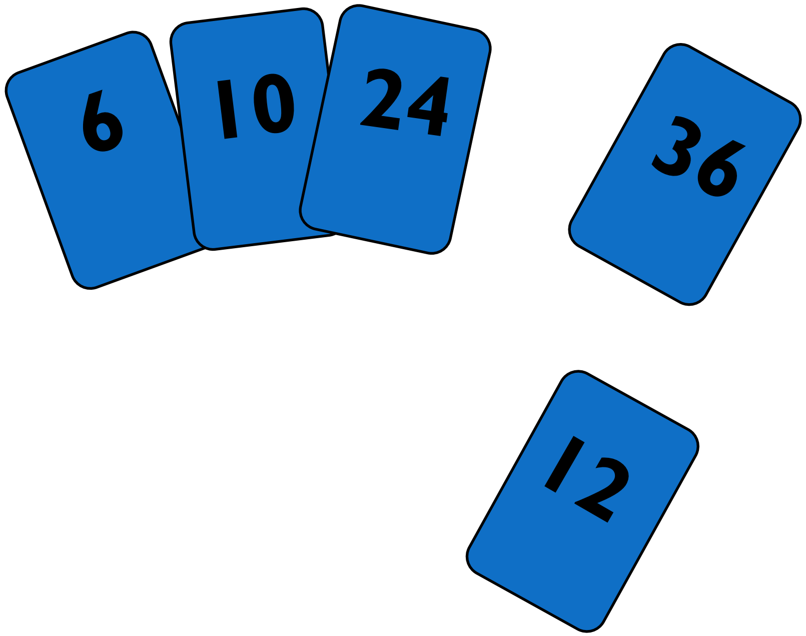
- Idea: like sorting a hand of playing cards
  - Start with an empty left hand and the cards facing down on the table.
  - Remove one card at a time from the table, and insert it into the correct position in the left hand
    - compare it with each of the cards already in the hand, from right to left
  - The cards held in the left hand are sorted
    - these cards were originally the top cards of the pile on the table

Insertion Sort

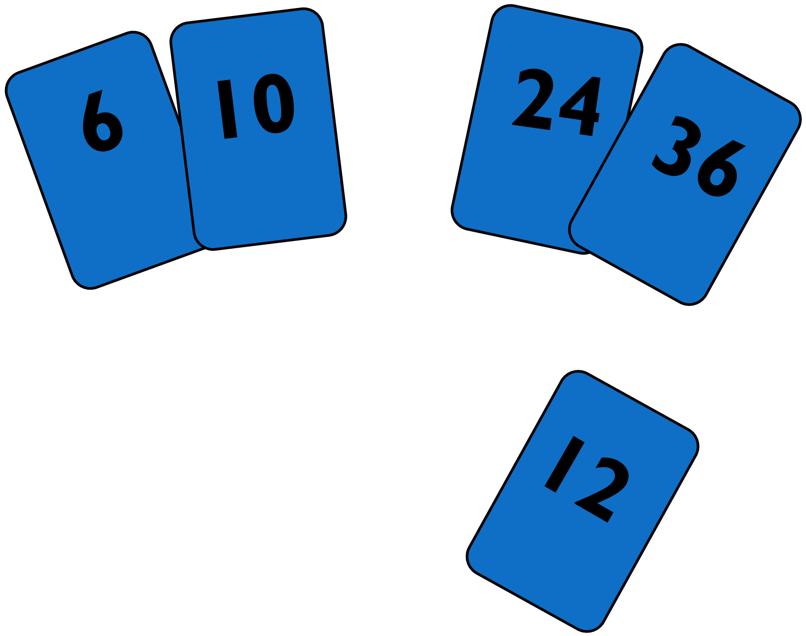


To insert 12, we need to make room for it by moving first 36 and then 24.

Insertion Sort



Insertion Sort



Insertion Sort

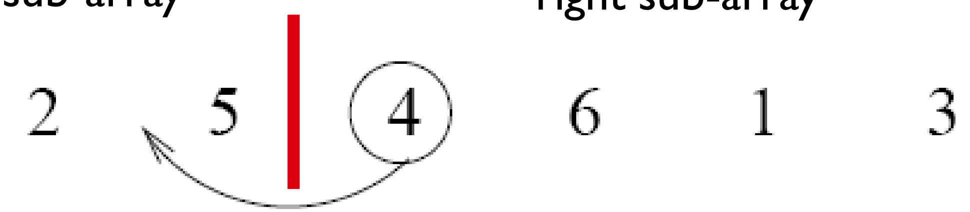
input array

5 2 4 6 1 3

at each iteration, the array is divided in two sub-arrays:

left sub-array

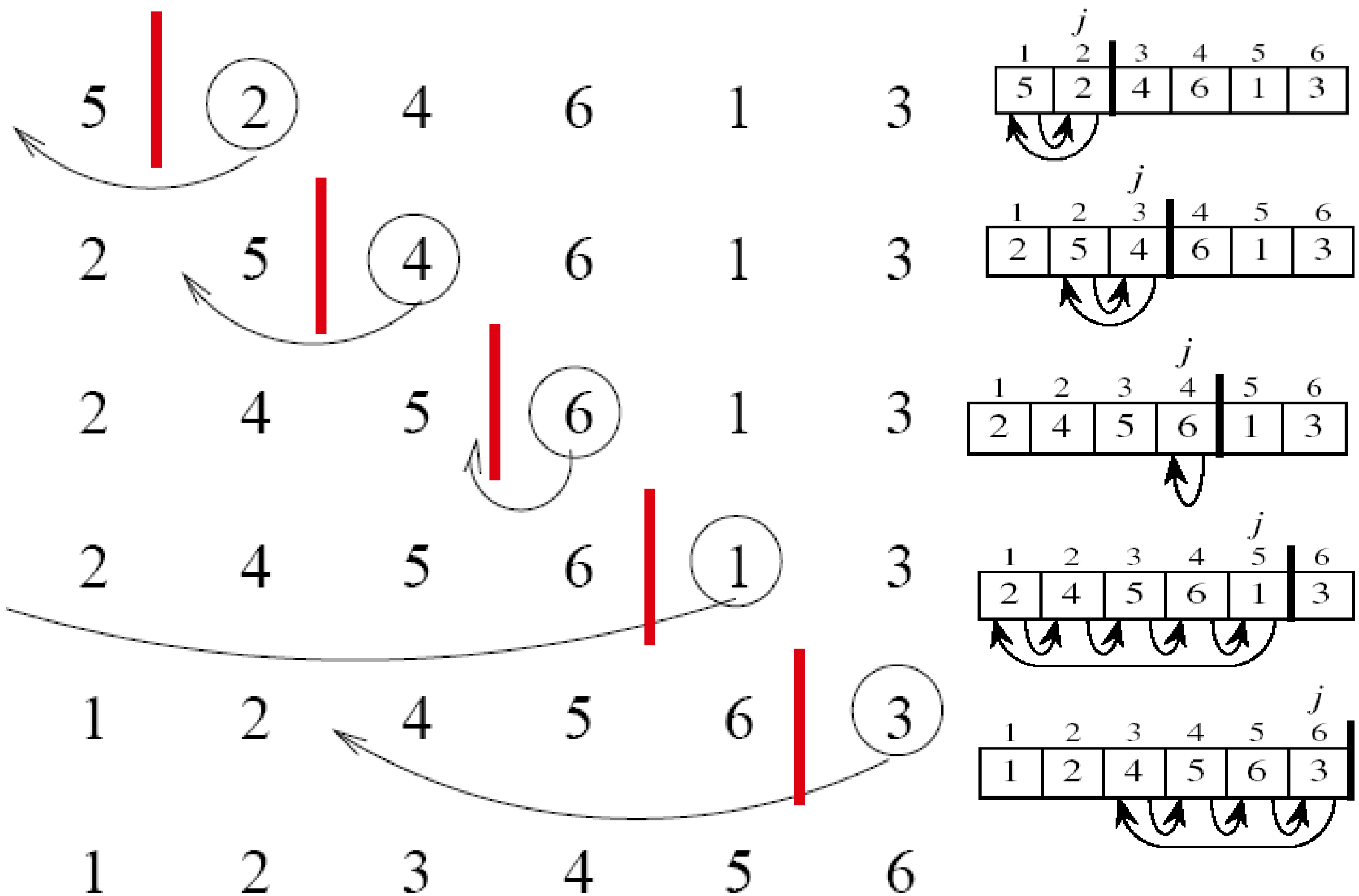
right sub-array



sorted

unsorted

Insertion Sort



INSERTION-SORT

Alg.: INSERTION-SORT(*A*)

1	2	3	4	5	6	7	8
a_1	a_2	a_3	a_4	a_5	a_6	a_7	a_8

key

for $j \leftarrow 2$ **to** n

  do $key \leftarrow A[j]$

   ▷ Insert $A[j]$ into the sorted sequence $A[1 \,..\, j-1]$

   $i \leftarrow j - 1$

   while $i > 0$ and $A[i] > key$

    do $A[i + 1] \leftarrow A[i]$

     $i \leftarrow i - 1$

   $A[i + 1] \leftarrow key$

- Insertion sort – sorts the elements in place

Analysis of Insertion Sort

```
INSERTION-SORT(A)
  for j ← 2 to n
      do key ← A[ j ]
      ▷Insert A[ j ] into the sorted sequence A[1 . . j -1]
         i ← j - 1
         while i > 0 and A[i] > key
             do A[i + 1] ← A[i]
                i ← i – 1
         A[i + 1] ← key
```

Best Case Analysis

- The array is already sorted

“while i > 0 and A[i] > key”

  - $A[i] \le key$ upon the first time the **while** loop test is run (when $i = j - 1$)

Worst Case Analysis

- The array is in reverse sorted order **“while** i > 0 and A[i] > key”
  - Always *A*[i] > key in **while** loop test
  - Have to compare key with all elements to the left of the j-th position ⇒ compare with j-1 elements

Insertion Sort - Summary

- Advantages
  - Good running time for "almost sorted" arrays $\Theta(n)$
- Disadvantages
  - $\Theta(n^2)$ running time in worst and average case


Thank you!