Agenda

- Introduction
- Arrays
- Passing Array as Function Arguments
- Multi-Dimensional Arrays
- Sorting

Sorting

- Sorting: process of arranging items in some sequence or list

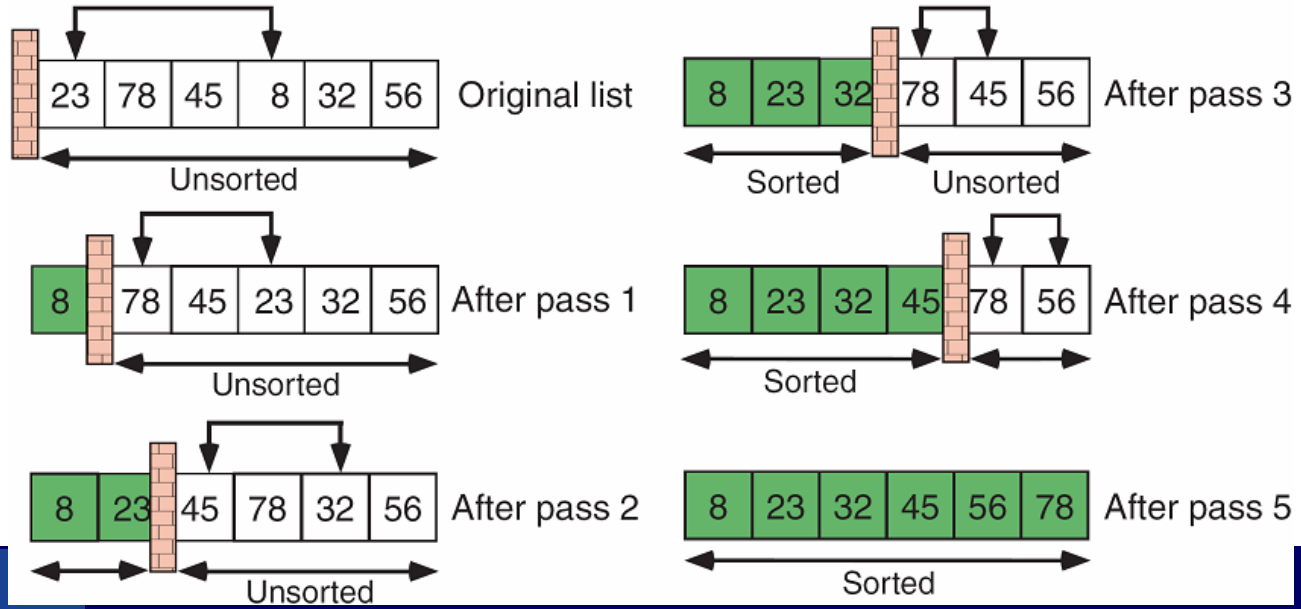
  Sequence: (4, 3, 8, 1, 9, 2)

  - Sorted in ascending order: (1, 2, 3, 4, 8, 9)
  - Sorted in descending order: (9, 8, 4, 3, 2, 1)

- Sorting algorithms
  - Selection sort
  - Bubble sort
  - Insertion sort
  - Quicksort/mergesort/heapsort/…
  - ETC.

Selection Sort

- Idea (sorting in ascending order)
  - List is divided into two sublists, **sorted** and **unsorted**
    - Initially, all elements are in **unsorted**
  - At each pass, select the smallest from **unsorted** sublist and put it at the end of **sorted** sublist
    - **sorted** gains one, but **unsorted** loses one.
  - Repeat n times



Selection Sort Algorithm

- Selection sort

```
void SelectionSort(int array[], int size)
{
   int i;   // start of unsorted
   int j;   // index in unsorted
   for(i = 0; i < size; i++){
      for(j = i + 1; j < size; j++){
         if(array[j] < array[i]){
            int temp = array[i];
            array[i] = array[j];
            array[j] = temp;
         } // if
      } // for j
   } // for i
}
```

- Main

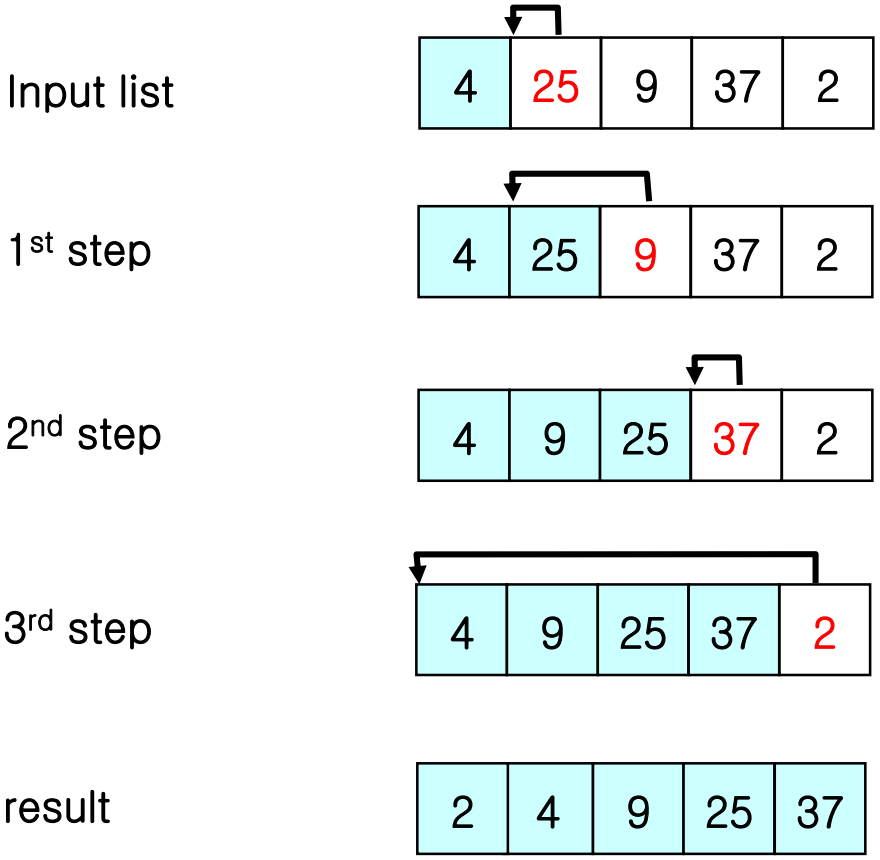
```
int main()
{
   int a[5] = { 3, 2, 4, 1, 0 };
   int i = 0;
   SelectionSort(a, 5);

   for(i = 0; i < 5; i++)
      printf("a[%d] = %d₩n", i, a[i]);
}
```

run

Insertion Sort

- Pick one record from unsorted list and insert it into sorted list

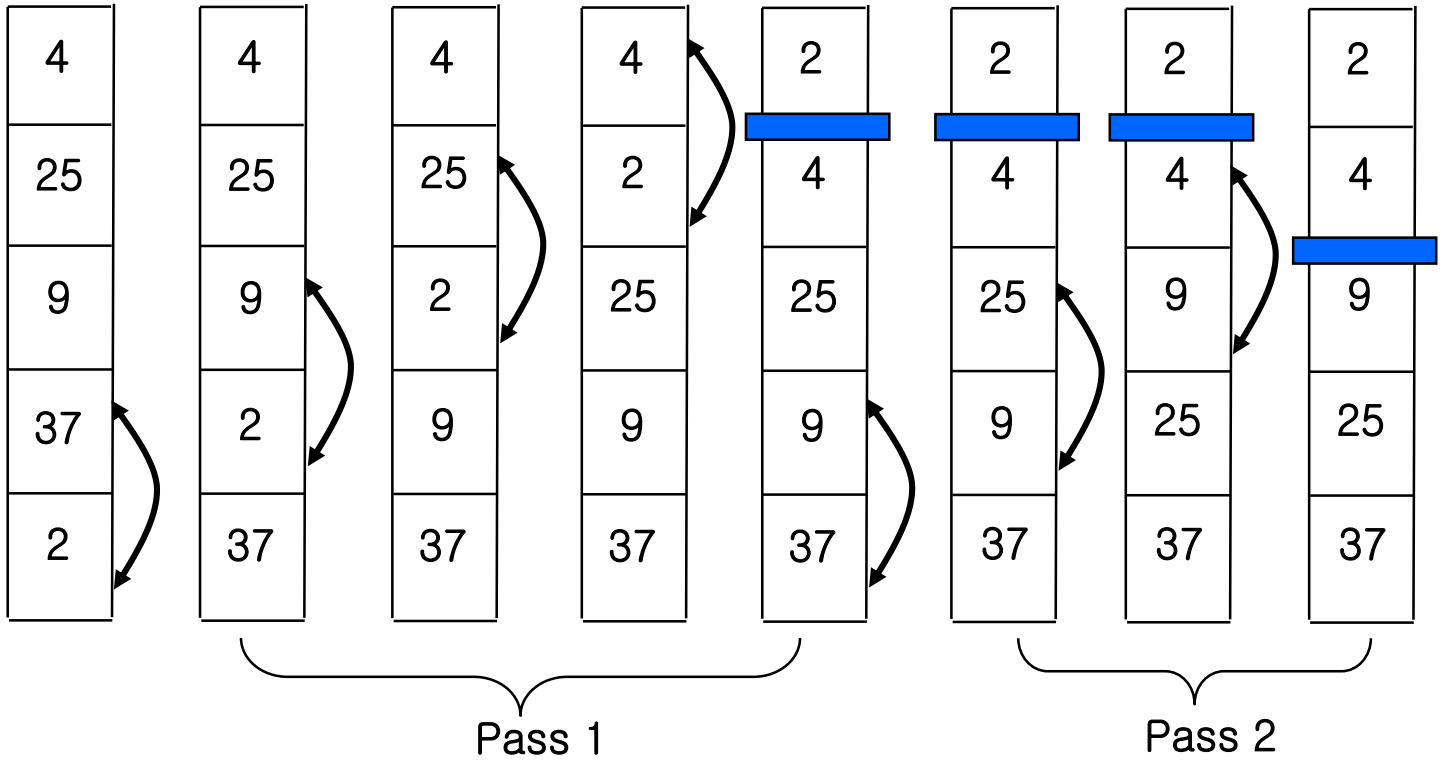


Insertion Sort

- Algorithm

```
void insertion_sort(element list[], int n)
{
    int i, j;
    element next;
    for(i = 1; i < n; i++){
        next = list[i];
        for(j = i - 1; j >= 0 && next.key < list[j].key; j--)
            list[j+1] = list[j];
        list[j+1] = next;
    }
}
```

Bubble Sort


4
25
9
37
2
4
25
9
2
37
4
25
2
9
37
4
2
25
9
37
2
4
25
9
37
2
4
25
9
37
2
4
9
25
37
2
4
9
25
37
Pass 1
Pass 2