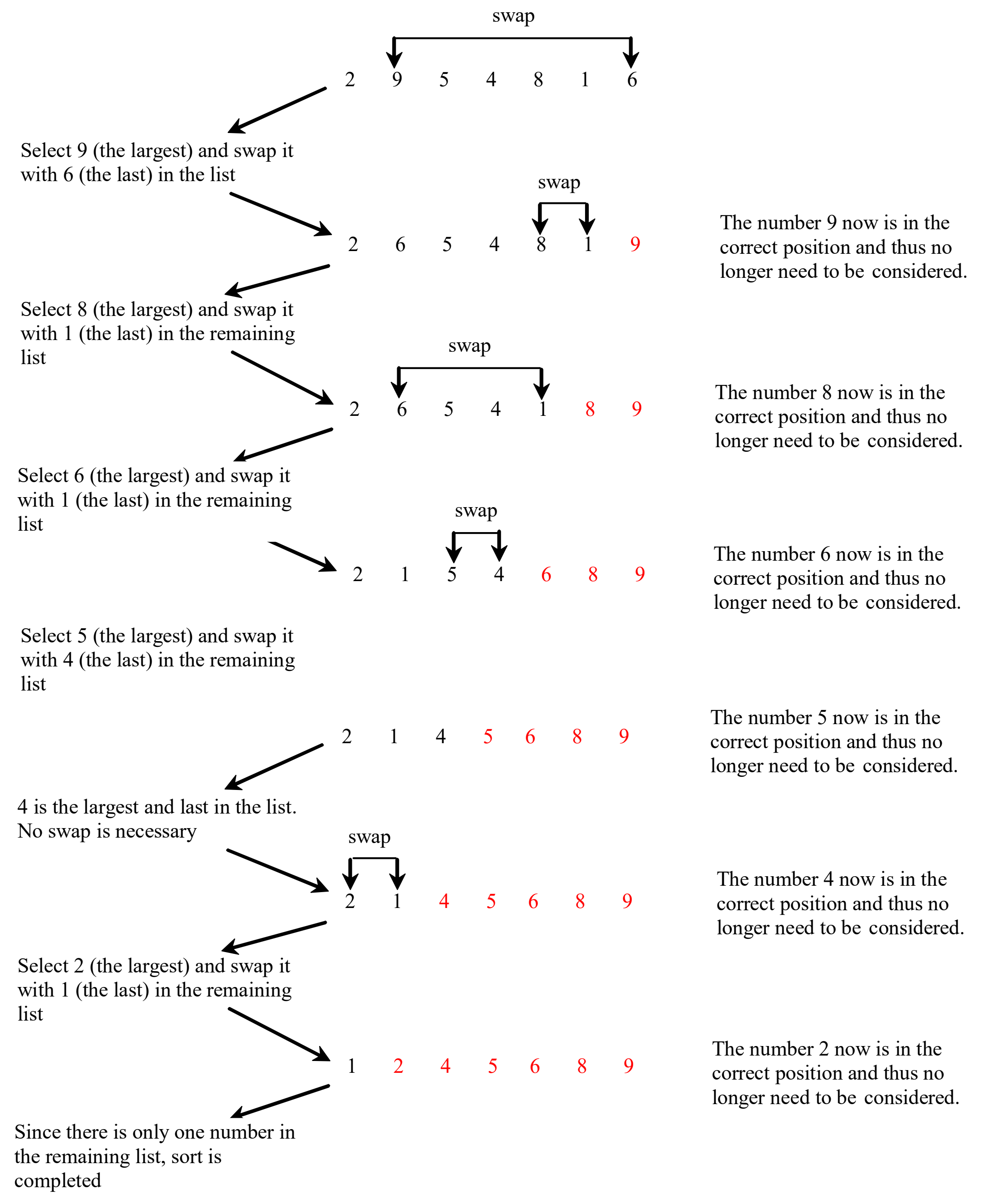
Sorting algorithms 1
- Insertion, Selection, Bubble

선택정렬

{2, 9, 5, 4, 8, 1, 6}


swap
2 9 5 4 8 1 6
Select 9 (the largest) and swap it with 6 (the last) in the list
swap
2 6 5 4 8 1 9
The number 9 now is in the correct position and thus no longer need to be considered.
Select 8 (the largest) and swap it with 1 (the last) in the remaining list
swap
2 6 5 4 1 8 9
The number 8 now is in the correct position and thus no longer need to be considered.
Select 6 (the largest) and swap it with 1 (the last) in the remaining list
swap
2 1 5 4 6 8 9
The number 6 now is in the correct position and thus no longer need to be considered.
Select 5 (the largest) and swap it with 4 (the last) in the remaining list
2 1 4 5 6 8 9
The number 5 now is in the correct position and thus no longer need to be considered.
4 is the largest and last in the list. No swap is necessary
swap
2 1 4 5 6 8 9
The number 4 now is in the correct position and thus no longer need to be considered.
Select 2 (the largest) and swap it with 1 (the last) in the remaining list
1 2 4 5 6 8 9
The number 2 now is in the correct position and thus no longer need to be considered.
Since there is only one number in the remaining list, sort is completed

참 고 Code Review(선택정렬)

```
void selectionSort(double list[ ], int arraySize)
{  for (int i = arraySize - 1; i >= 1; i--)
  {    // Find the maximum in the list[0..i]
     double currentMax = list[0];
     int currentMaxIndex = 0;
     for (int j = 1; j <= i; j++)
     {
       if (currentMax < list[j])
       {
         currentMax = list[j];
         currentMaxIndex = j;      }
     }
     // Swap list[i] with list[currentMaxIndex] if necessary;
     if (currentMaxIndex != i)
     { list[currentMaxIndex] = list[i];
       list[i] = currentMax;    }
}
```

삽입 정렬

int[] myList = {2, 9, 5, 4, 8, 1, 6}; // Unsorted

Step 1: Initially, the sorted sublist contains the first element in the list. Insert 9 to the sublist.

Step2: The sorted sublist is {2, 9}. Insert 5 to the sublist.

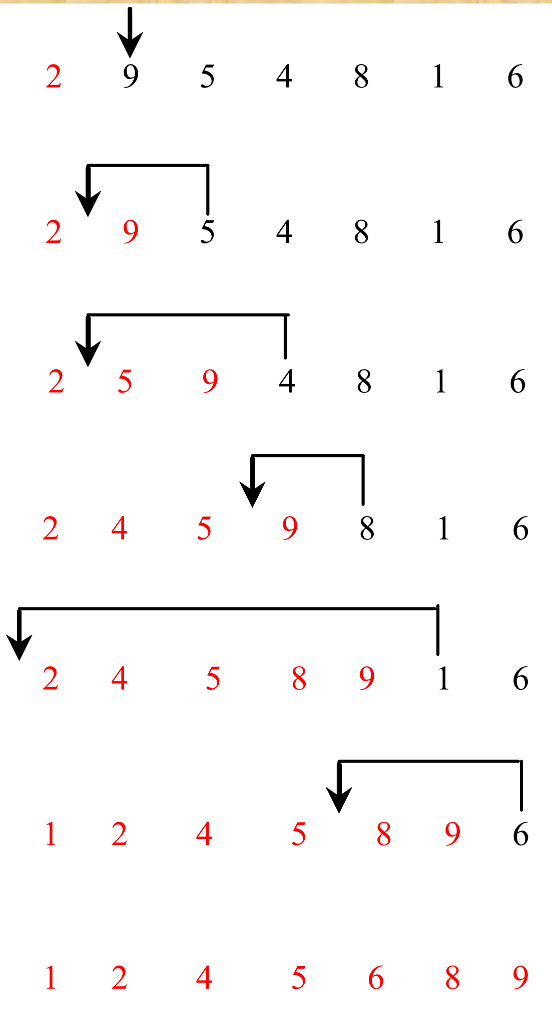
Step 3: The sorted sublist is {2, 5, 9}. Insert 4 to the sublist.

Step 4: The sorted sublist is {2, 4, 5, 9}. Insert 8 to the sublist.

Step 5: The sorted sublist is {2, 4, 5, 8, 9}. Insert 1 to the sublist.

Step 6: The sorted sublist is {1, 2, 4, 5, 8, 9}. Insert 6 to the sublist.

Step 7: The entire list is now sorted



참고 Code Review(삽입정렬)

```
void insertionSort(double list[ ], int arraySize)
 {
 for (int i = 1; i < arraySize; i++)
 {
   /* insert list[i] into a sorted sublist list[0..i-1] so that
      list[0..i] is sorted. */
   double currentElement = list[i];
   int k;
   for (k = i - 1; k >= 0 && list[k] > currentElement; k--)
   {
     list[k + 1] = list[k];
   }
   // Insert the current element into list[k+1]
   list[k + 1] = currentElement;
 }
```

Bubble Sort

- Simplest sorting algorithm
- Idea:
  - 1. Set flag = false
  - 2. Traverse the array and compare pairs of two consecutive elements
    - 1.1 If E1 ≤ E2 -> OK (do nothing)
    - 1.2 If E1 > E2 then Swap(E1, E2) and set flag = true
  - 3. repeat 1. and 2. while flag=true.

Bubble Sort

1	1	23	2	56	9	8	10	100
2	1	2	23	56	9	8	10	100
3	1	2	23	9	56	8	10	100
4	1	2	23	9	8	56	10	100
5	1	2	23	9	8	10	56	100

---- finish the first traversal ----

1	1	2	23	9	8	10	56	100
2	1	2	9	23	8	10	56	100
3	1	2	9	8	23	10	56	100
4	1	2	9	8	10	23	56	100

---- finish the second traversal ----

…

Bubble Sort

```
public void bubbleSort (Comparable[] arr)
  {
  boolean isSorted = false;
  while (!isSorted) {
    isSorted = true;
    for (i = 0; i<arr.length-1; i++)
     if (arr[i].compareTo(arr[i+1]) > 0)
  {
        Comparable tmp = arr[i];
        arr[i] = arr[i+1];
        arr[i+1] = tmp;
        isSorted = false;
     }
  }
}
```