SORTING ALGORITHMS

1) Bubble Sort:

Theory:

Let us assume a set of elements

20, 35, -15, 7, 55, 1, -22 ← Indices
0, 1, 2, 3, 4, 5, 6

→ Consider a variable "LastUnsortedIndex" that tracks till where the array has been sorted

→ Now, the array is not sorted yet, therefore the LastUnsortedIndex = 6

→ Now, Let us use a traversal variable i, which is initialized to 0

20	35	-15	7	55	1	-22
0	1	2	3	4	5	6
↑						↑
i, Traversal Variable						LastUnsortedIndex

LastUnsortedIndex = 6; Iterations:

i) [20 35] -15 7 55 1 -22 ; 20<35, ignore
ii) 20 [-15 35] 7 55 1 -22 ; 35>-15; swap
iii) 20 -15 [7 35] 55 1 -22 ; 35>7; swap
iv) 20 -15 7 [35 55] 1 -22 ; 35<55; ignore
v) 20 -15 7 35 [1 55] -22 ; 55>1; swap
vi) 20 -15 7 35 1 [-22 55] ; 55>-22; swap

At the end of first iteration, the Largest element is at it's correct position

Now, LastUnsortedIndex - -;
LastUnsortedIndex = 5;

	20	-15	7	35	1	-22	55	
i)	-15	20	7	35	1	-22	55	20 > -15; swap
ii)	-15	7	20	35	1	-22	55	20 > 7; swap
iii)	-15	7	20	35	1	-22	55	20 < 35; ignore
iv)	-15	7	20	1	35	-22	55	35 > 1; swap
v)	-15	7	20	1	-22	35	55	35 > -22; swap

At the end of this iteration, the second largest element is at its correct position

Now, lastUnsortedIndex--;
lastUnsortedIndex = 4;

	-15	7	20	1	-22	35	55
i)	-15	7	20	1	-22	35	55 ; -15 < 7; ignore
ii)	-15	7	20	1	-22	35	55 ; 7 < 20; ignore
iii)	-15	7	1	20	-22	35	55 ; 20 > 1; swap
iv)	-15	7	1	-22	20	35	55 ; 20 > -22; swap

At the end of this iteration, the third largest element is at its correct position

Now, lastUnsortedIndex--;
lastUnsortedIndex = 3;

-15	7	1	-22	20	35	55

[-15 7] 1 -22 [20] [35] [55] ; -15<7; ignore

-15 [1 7] -22 [20] [35] [55] ; 7>1; swap;

-15 1 [-22 7] [20] [35] [55] ; 7>-22; swap

At the end of this iteration, the fourth largest element is at its correct position

Now, lastUnsortedIndex--;
lastUnsortedIndex = 2;

-15 1 -22 [7] [20] [35] [55]

[-15 1] -22 [7] [20] [35] [55] ; -15<1; ignore

-15 [-22 1] [7] [20] [35] [55] ; 1>-22; swap

At the end of this iteration, the fifth largest element is at its correct position

Now, lastUnsortedIndex--;
lastUnsortedIndex = 1;

-15 -22 [1] [7] [20] [35] [55]

[illegible] ; -15>-22; swap

[-22 -15] [1] [7] [20] [35] [55] ; -15>-22; swap

At the end of this iteration, sixth largest element is at its correct position

Now, the condition lastUnsortedIndex > 0 is failed and loop is exited.

Implementation of Bubblesort:

```
import java.util.Scanner;
class Bubble_Sort
{
 Public Static Void main(String[] args)
 {
   //Collection of data
    /* To manually collect the data from the
       User, remove these comments

       int array[] = new int[10];
       Scanner in = new Scanner(System.in);
       System.out.println("Enter size of array:");
       int n = in.nextInt();
       System.out.println("Enter elements:");
       for(int i=0; i<n; i++)
       {
        array[i] = in.nextInt();
       }
    */
      // Hard code
      int array[]={20,35,-15,7,55,-1,-22};

  // Bubble sort implementation
```

```
for(int lastUnsortedIndex = array.length - 1;
lastUnsortedIndex >0;  lastUnsortedIndex - -)
{
  for(int i=0; i< lastUnsortedIndex ; i++)
   {
     if(array[i] >  array[i+1])
       {
          swap(array, i, i+1);
        }//if
    } // inner for
  } // outer for
// Printing elements using for-each loop
 for(int element : array)
 {
   System.out.println(element);
  } // for
}//main
Public static void swap(int[] array, int i, int j)
{
   if(i==j) {return ;}
   int temp = array[i];
   array[i] = array[j];
   array[j] = temp;
 } // swap
} class - Bubble sort.
```