SELECTION SORT

1) loop on I [0, Array.lenght]

2) Asume Array [I] is minimun Value.

3) loop on J [I+1, Array.lenght]

   if (Array [J] < Array [I])

   SWAP

---

minVola Pos

| 99 | 38 | 80 | 48 |

| 0 |

i
J

| 38 | 99 | 80 | 48 |

| 1 |

i

| 2 |

| 38 | 99 | 80 | 48 |

i   J

| 38 | 80 | 99 | 48 |

| 38 | 80 | 99 | 48 |

| 38 | 80 | 98 | 99 |

— END.
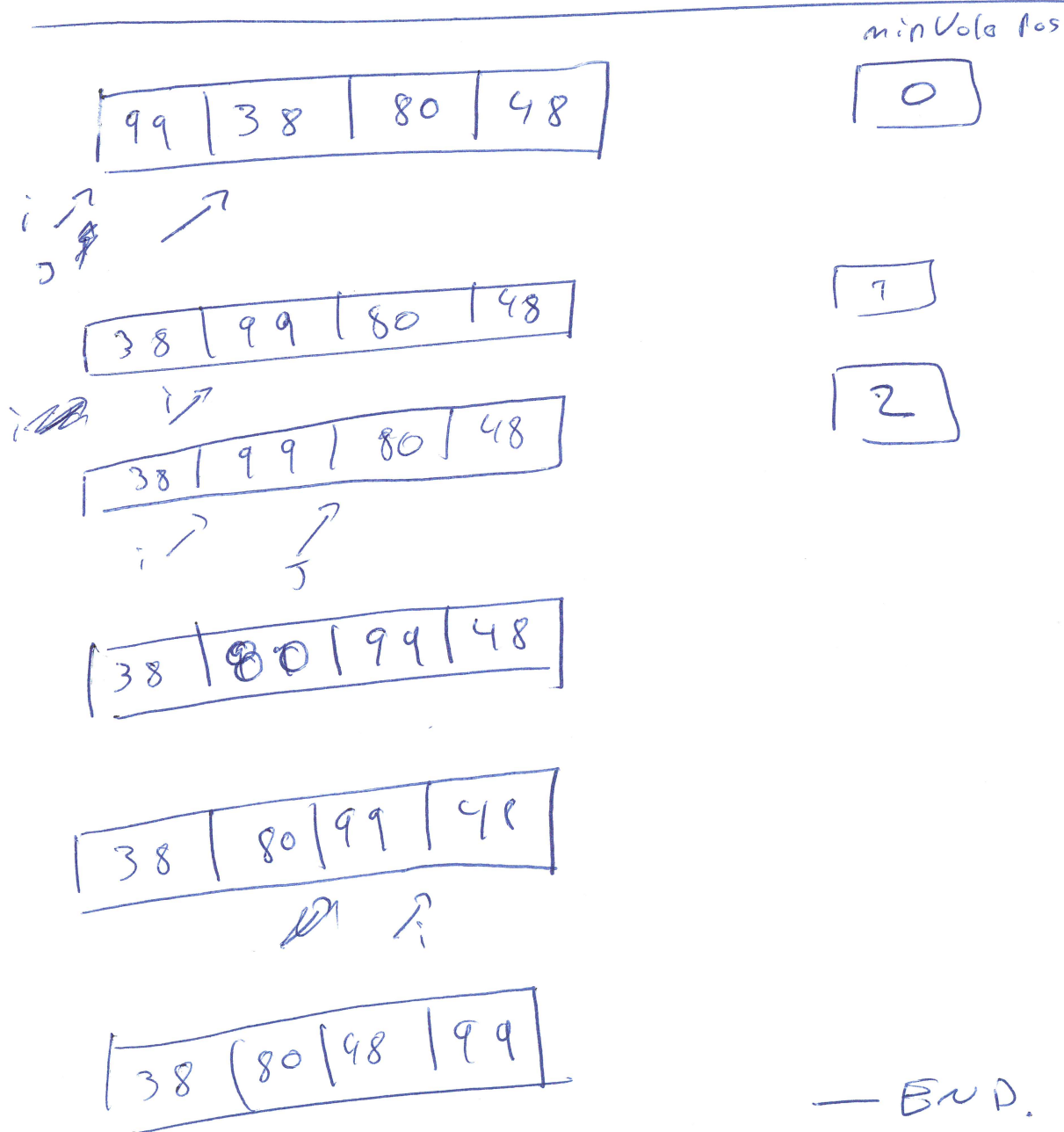
Buble Sort

1) Loop on I (0, array-lenght)

2) loop on J (0, array lenght -1)

3) if (a[J] < a[J+1])
   swap.

J
| 99 | 38 | 80 | 48 |
i

J
| 38 | 99 | 80 | 48 |
i

J
| 38 | 99 | 80 | 48 |
i

J
| 38 | 80 | 99 | 48 |
i

J
| 38 | 80 | 99 | 48 |
i

J
| 38 | 80 | 99 | 48 |
i

J
| 38 | 80 | 99 | 48 |
i

J
| 38 | 80 | 48 | 99 |
i

J
| 38 | 80 | 48 | 99 |
i

- - - - -
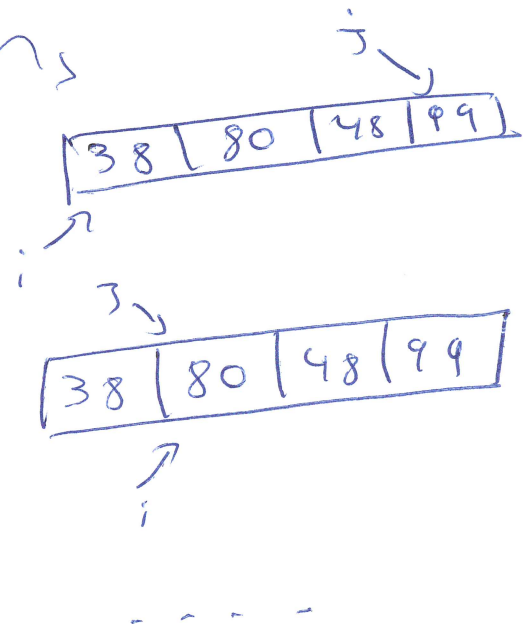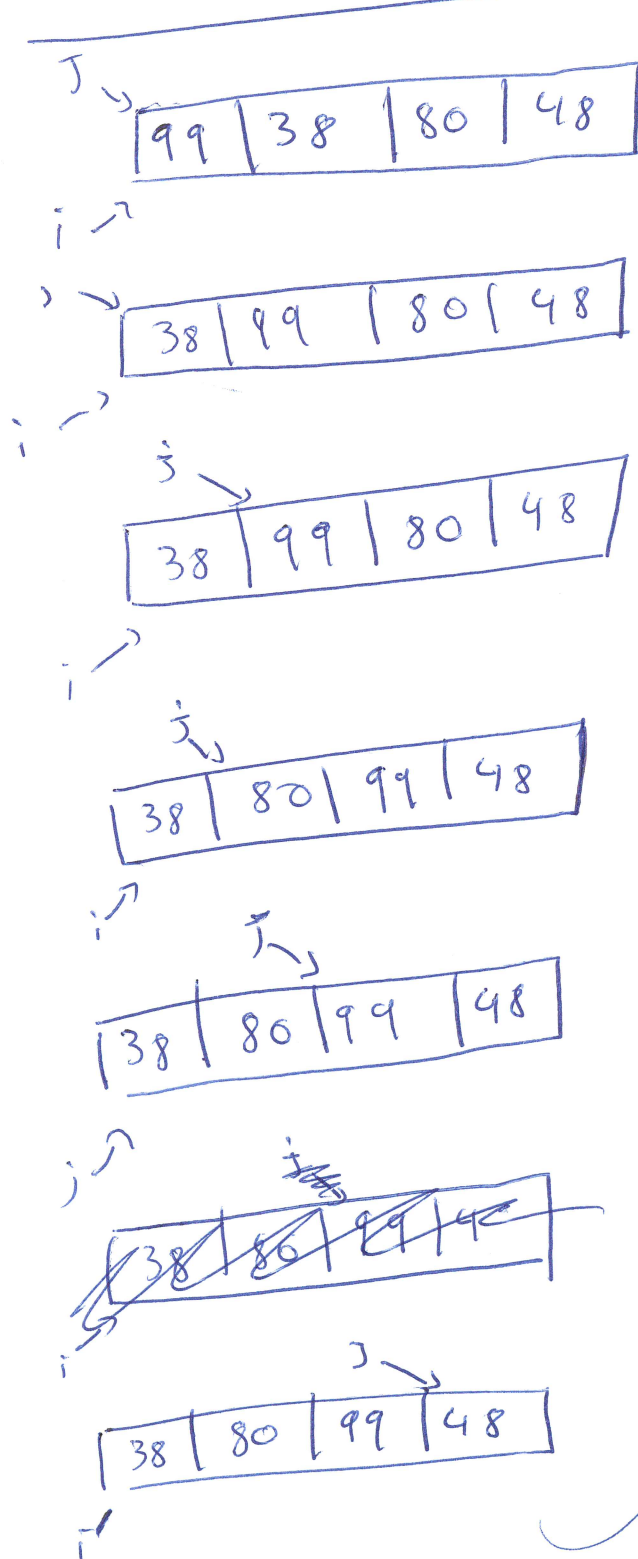
Insert Sort

1) loop on I[1, lenght]

2) Store eside array[I] ~ k

3) Compare from I to 0
   if k > on I —> insert k in on I.

---

	vale k	Pos to insert
99 38 80 48	38	1
99 99 80 48	38	0 — End 3
38 99 80 48		
38 99 80 48	80	2
38 99 99 48	80	1
38 80 99 48		— End 3
38 80 99 48	48	3

i[1, lenght]

Pivot | lou | hg
38 | 0 | 3

| 99 | 38 | 80 | 48 |

low | hi
38 | 0 | 3

| 99 | 38 | 80 | 48 |

38 | 0 | 3

| 99 | 38 | 80 | 48 |

| 38 | 99 | 80 | 48 |

low < J

=

i > high

quicksort (1, 3)

Pivot | lou | hg
80 | 1 | 3

| 38 | 99 | 80 | 48 |

| 38 | 48 | 80 | 99 |

80 | 1 | 3

| 38 | 48 | 80 | 99 |

low < j

=

i < hg

quicksort (

=

Quick Sort

Note: Recursive Not Iterative (possibly !-))

1) Select Sort scope [o, array left] / for full array

2) Select Pivot [Average Value 4 instance]

< Recursion >

   3) Save tree values → Pivot : ara[AVG(loo, My]

                   → i : low

                   → j : high.

   3.1) loop on i until j (increasing)

       loop from j until i (decreasing)

       - until ~~value~~ array[i] < pivot

       - until array[j] > pivot.

       if (i <= j) ⟹ SWAP(i, j)

   3.2) if [low < j) ← recu. Sort (low, j)

       if (hig < i) ← recursi Sort (i, hig)