Insertion Sort

⇒ In each pass of an insertion sort, one or more pi
of data are inserted into their correct location in an
ordered list.

let us see, two insertion sorts

- straight insertion sort
- Shell sort.

Straight Insertion Sort:

of data are inserted into their correct location in
ordered list.

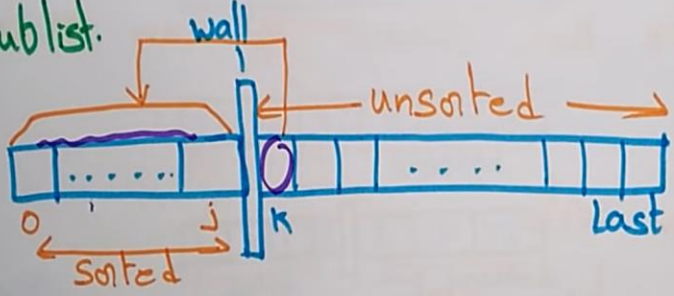
⇒ let us see, two insertion sorts

- straight insertion sort
- Shell sort.

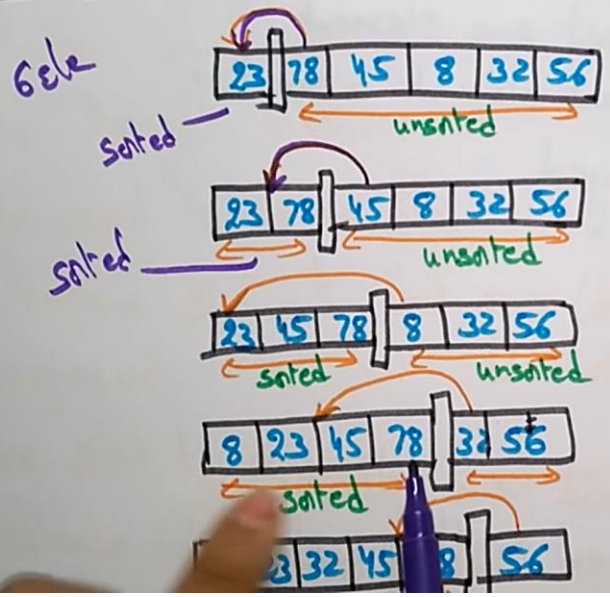
Straight Insertion Sort:

- The list at any moment is divided into sorted and unsorted sublist

- In each pass the first element of the unsorted sublist is inserted into the sorted sublist.



traight Insertion Sort example



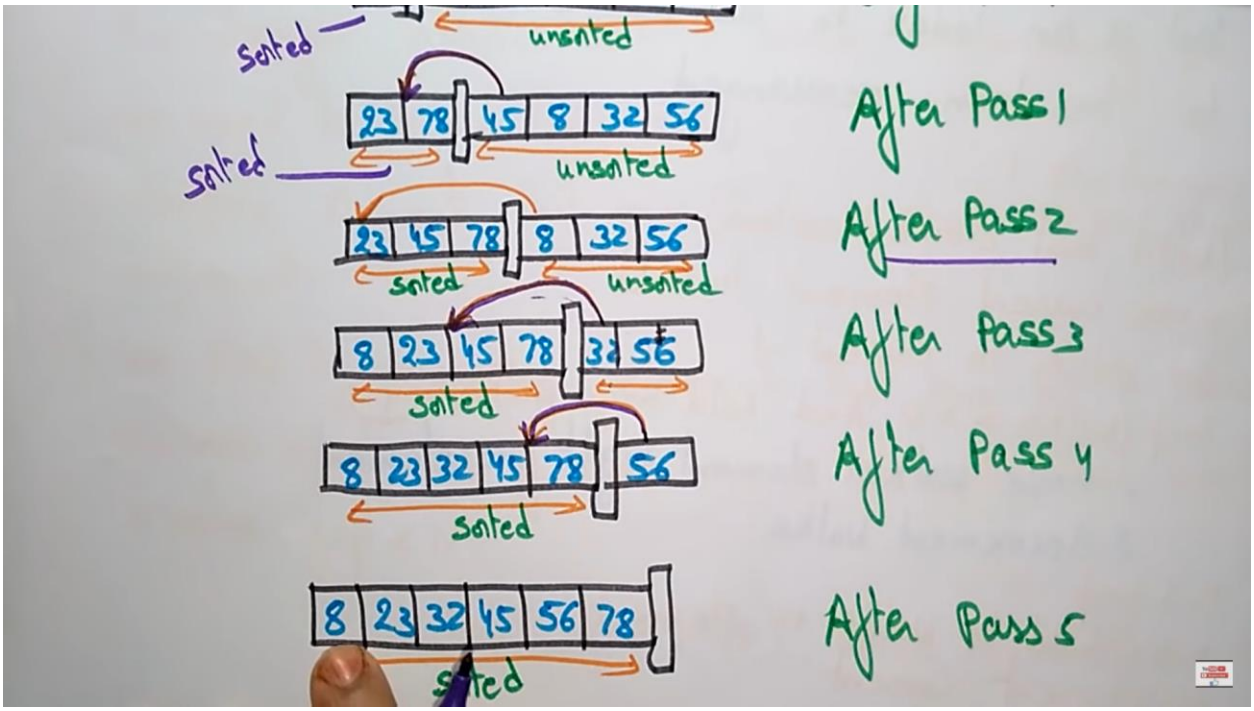
Original list

After Pass 1

After Pass 2

After Pass 3

After Pass 4

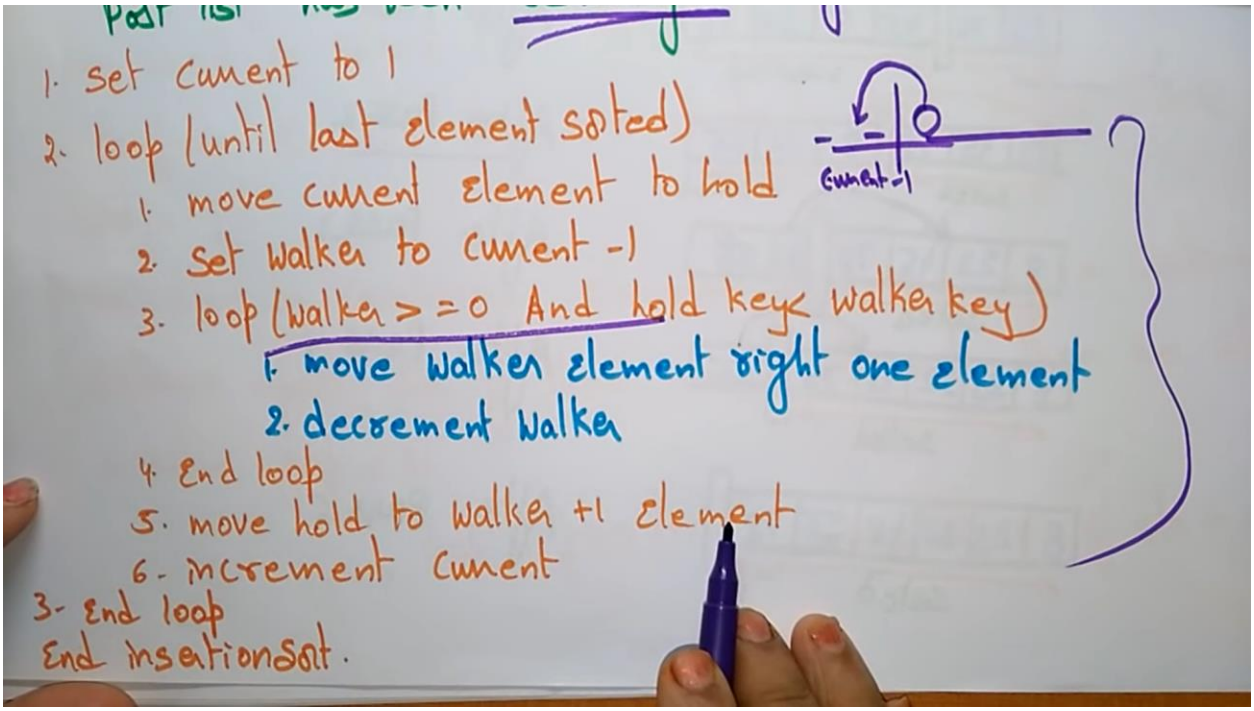


→ Algorithm insertion sort (list, last)

Pre list must contain at least one element
last is an index to last element in the list

Post list has been rearranged

1. set current to 1
2. loop (until last element sorted)
   1. move current element to hold
   2. set walker to current - 1
   3. loop (walker >= 0 And hold key < walker key)
      1. move walker element right one element
      2. decrement walker



5	2	4	6	1	3	
2	5	4	6	1	3	
2	4	5	6	1	3	
2	4	5	6	1	3	No change
1	2	4	5	6	3	
1	2	3	4	5	6	

Insertion sort

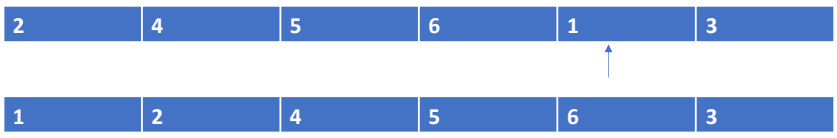
2	4	5	6	1	3
1	2	3	4	5	6


```
key = a[5];    i= 4;
while ( i > 0 and a[   i] > key)
{
 a[i+1] = a[ i];
 i=i-1;
}
a[i+1] = key
```

```
Algorithm insertion_sort (a, n)
{
for j = 2 to n
{
        key = a[j];    i= j-1;
        while ( i > 0 and a[   i] > key)
        {
                a[i+1] = a[ i];
                i=i-1;
        }
        a[i+1] = key
}
}
```


INSERTION-SORT(A)

```
1  for j = 2 to A.length
2      key = A[j]
3      // Insert A[j] into the sorted sequence A[1 .. j − 1].
4      i = j − 1
5      while i > 0 and A[i] > key
6          A[i + 1] = A[i]
7          i = i − 1
8      A[i + 1] = key
```



1	2	3	4	5	6

Elements are in ascending order : _____n-1_____ (best case)

```
INSERTION-SORT(A)
1  for j = 2 to A.length
2      key = A[j]
3      // Insert A[j] into the sorted sequence A[1 .. j − 1].
4      i = j − 1
5      while i > 0 and A[i] > key
6          A[i + 1] = A[i]
7          i = i − 1
8      A[i + 1] = key
```

Insertion analysis

6	5	4	3	2	1

Elements are in descending order : 1+2+3+4+5 = 1+2+3+4+….n -1 = n(n-1)/2 =(n*n - n)/2 -> worst case
Average case : (n+1)/2 * (n-1) = O(n*n)

```
INSERTION-SORT(A)
1  for j = 2 to A.length
2      key = A[j]
3      // Insert A[j] into the sorted sequence A[1 .. j − 1].
4      i = j − 1
5      while i > 0 and A[i] > key
6          A[i + 1] = A[i]
7          i = i − 1
8      A[i + 1] = key
```

Insertion analysis