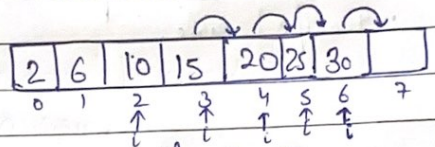
2) Insertion Sort

↳ Inserting element in a already sorted array at a sorted Position.

Algo → For insertion in Array ↳ Go on shifting larger elements wherever to get ~~space~~ element less than element to be inserted. then insert the element at its next index.

This is insertion Method

Array A

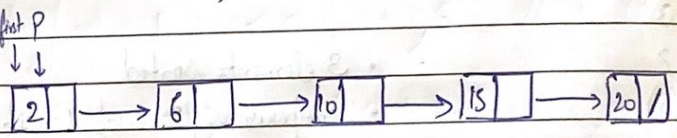


Max $O(n)$ Min $O(1)$

30 > 12 move 30
25 > 12 move 25
20 > 12 move 20
15 > 12 move 15

Now, 10 < 12 so we will insert 12 at (i+1).

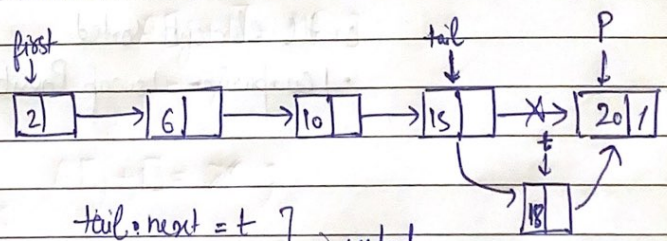
LinkedList



element = 18

Keep on moving tail & P till you find an element that is > than the elmt to be inserted.

After traversing



tail.next = t
t.next = P
inserted

Max $O(n)$ Min $O(1)$

Algo for insertion Sort →

A → | 8 | 5 | 7 | 3 | 2 |

Take element & insert this side

→ Here 1 element sorted 4 elements not sorted.

1) Insert 5

| 8 | | 7 | 3 | 2 |

(1 comparison) (1 shift)

I st Pass

8 > 5 so shift 8 & insert 5

Result → | 5 | 8 | 7 | 3 | 2 |

Sorted | unsorted

2) Insert 7

| 5 | 8 | | 3 | 2 |

(2 comparisons) (Max 2 shifts)

2nd Pass

8 > 7, so shift 8

5 < 7, so no need to shift + insert 7

Result → | 5 | 7 | 8 | 8 | 2 |

Sorted | unsorted

3) Insert 3

| 5 | 7 | 8 | | 2 |

(3 comparisons) (Max 3 shifts)

3rd Pass

8 > 3, so shift 8

7 > 3, " " 7

5 > 3, " " 5 + Insert 3

Result → | 3 | 5 | 7 | 8 | 2 |

Sorted | unsorted

4) Insert 2

| 2 | 3 | 5 | 7 | 8 |

(4 comparisons) (Max 4 shifts)

4th Pass

8 > 2, shift 8

7 > 2, " 7

5 > 2, " 5

3 > 2, " 3, insert 2

Result → | 2 | 3 | 5 | 7 | 8 |

Sorted

	Min	Max
Comparison →	$O(n)$	$O(n^2)$
Swaps →	$O(1)$	$O(n^2)$

Total → 4 Passes N elements (N-1) Passes

No. of Comparisons → $1+2+3+4 \ldots (n-1)$ $\frac{n(n-1)}{2} \Rightarrow O(n^2)$ Time complexity

No. of swaps → $O(n^2)$

→ In this intermediate Passes do not give any useful Result as we get largest element at last in Bubble Sort after 1st Pass.

→ In LL we do not have to shift element, so more compatible with LL than array.

Algo/Pseudo Code →

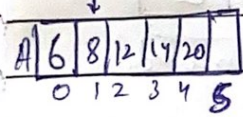
```
void insertionSort(int A[], int n){
    for(int i=1; i<n; i++){          // No. of Passes
        int j = i-1;                 // start checking from this element.
        int x = A[i];                // the element to be inserted
        while(j>-1 && A[j]>x)        // if this keep shifting
        {
            A[j+1] = A[j];
            j--;
        }
        A[j+1] = x;
    }
```

Start from 2nd element

Traverse whole array.

1, 2, 3, 4

5 elements 4 Passes



x = 10