§ Data Structures §

Sortings ::

Insertion Sort ::
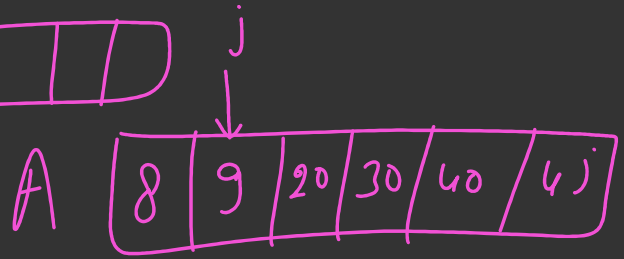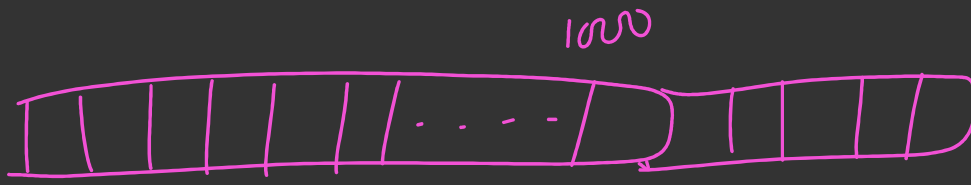
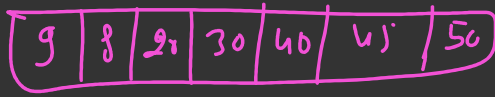Insertion (A, n)
{

$O(n^2)$

```
for (j=1; j ≤ n-1; j++)       ——→ O(n) (n
{    i= j-1;  key= A[j];
     while( A[i] > key && i ≥ 0) → (n
     {
          A[i+1] = A[i];
          i = i-1;
     } A[i+1] = key;
}
```

Worst Case = $O(n^2)$
Best Case = $O(n)$

A = | 2 | 8 | 5 | 7 | 8 | 9 |
      0   1   2   3   4   5

n = 6

Key: 8 5 8

A | 8 | 3 | 7 |

i

A[j] > key

| 7 | 8 | 9 | 11 | 5 |

1000

j

A | 8 | 9 | 20 | 30 | 40 | 4j |

Comparison based Sorting.

| 9 | 8 | 2₁ | 30 | 40 | 4j | 50 |

Stable Sortings. ✓
Implace Sorting ✓
Adaptive

Best Case

$O(n)$

A | 4 | 4 | 4 | 4 | 4 | 4 |

| 2* | 3 | 2⁺ | 5 | 9 |

| 2* | 2⁺ | 2 | 3 | 5 |

Merge Sort :-

L | 3 | 5 | 8 | | 2 | 9 | 10 | 13 |

(arrows labeled i, i, i, i ↓R above left array; j, j above right array)

A = | 2 | 3 | 5 | 8 | 9 | 10 | 13 |

K L[i] < R[k]

(arrows labeled k, k, k, k)

| 2 | 3 | 8 | 9 | ∞ | | 1 | 20 | 30 | 86 | ∞ |

(arrows labeled i, i, i, i and j, j, j, j)

| 2 | 3 | 8 | 9 | 15 | 86 | 30 | 80 |

Merging

| 2 | 8 | 6 | 9 | 3 | 1 |
  0   1   2   3   4   5

| 2 | 8 | 6 | | 9 | 3 | 1 |
  0   1   2   3   4   5

| 2 | 8 | | 6 | | 9 | 3 | | 1 |
  0   1   2   3   4   5

| 2 | 8 | | 6 | | 9 | | 3 | | 1 |

| 2 | 8 | | 3 | 9 |

| 2 | 6 | 8 | | 1 | 3 | 9 |

1, 2, 3, 6, 8, 5

A[k]=L[i]
j++;
}
else{
    A[k]=R[j]
    j++;
}
}


```
0   1   2   3
A [ 2 | 7 | 3 | 5 ]

k
q=2

n₁= 2-0
n₁= 2

L [ 2 | 7 | ]
```

$n_1 = q-P$

$q = 2$

$n_1 = 2-0$
$n_1 = 2$

merge(A,P,q,r)
{
    $n_1 = q-P$
    $n_2 = r-q+1$
    $L[1..\_\_n_1+1]$ $R[1..\_\_n_2+1]$
    for(i=0 to n_1)
    { $L[i] = A[P+i]$
    }
    for(j←0 to n_2)
        $R[j] = A[q+j]$
    $L[n_1+1] = \infty$ $R[n_2+1] = \infty$
    for(k←P to r)
    { if(L[i] ≤ R[j])

    i=0; j=0;

f_1 MS(1,6)

d_2 MS(1,3) $l=3$ f_3 ms(4,6) f_{14} M(1,3,6)

$q=2$ f_6 MS(3,3) $l=5$ f_8 MS(4,5) f_9 ms(6,6) f_{10} M(4,5,6)

f_5 MS(1,2) f_7 M(1,2,3)

f_{11} $q=1$ MS(1,1) f_{12} MS(2,2) f_{13} M(1,1,2) $l=4$ f_{14} MS(4,4) f_{15} MS(5,5) f_{16} M(4,4,5)

Space $= O(n)$
$= O($
Space $= n + \lg n = O(n)$

ms:
$$\text{Merge Sort}(A, P, r) \quad T(n)$$
{

   if$(P < r) \rightarrow O(1)$

   {

     $q \leftarrow \lfloor (P+r)/2 \rfloor \rightarrow O(1)$

     $T(n/2) \leftarrow \text{MergeSort}(A, P, q)$

     $T(n/2) \leftarrow \text{MergeSort}(A, q+1, r)$

     $n \leftarrow \text{Merge}(A, P, q, r)$

   }

} $T(n) = 2T(n/2) + n$

DFS.

$$T(n) = 2T(n/2) + n$$

$T(n)$ $T(n/2^0)$ ———————————— n

$T(n/2)$ $T(n/2)$ $T(n/2)$ ——————→ n

$T(n/4)$ $T(n/2^2)$ $T(n/2^2)$ $T(n/4)$ $T(n/4)$ $T(n/4)$ ——→ n

$T(n/2^3)$ ——————→ n

$T(n/2^k)$:

$T(1)$ $T(1)$ $T(1)$ $T(1)$ $T(1)$ $T(1)$ $T(1)$ $T(1)$ $T(1)$ $T(1)$ ——→ n

$$\frac{n}{2^k} = 1 \implies n = 2^k \implies k = \lg n$$

height of the tree = $k+1 = \lg n + 1$

$$\implies n(\lg n + 1) \implies n \lg n + n$$
$$O(n \lg n).$$