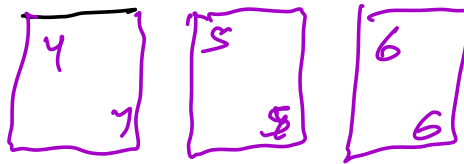


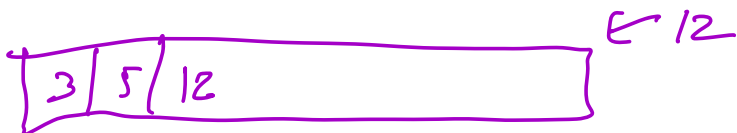
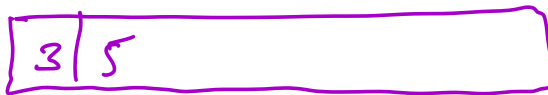
Sorting 3

Deck of cards

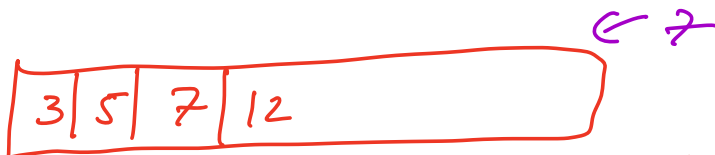
Insertion sort



← 3



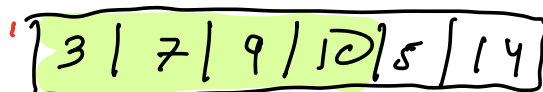
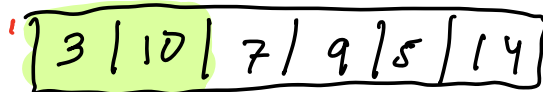
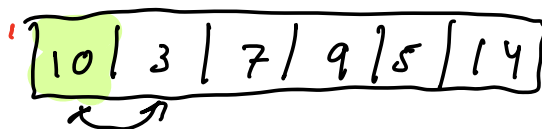
← 12



← 7



← 4



3 | 5 | 7 | 9 | 10 | 14

```
for( i=1; i < N; i++)
    j = i-1
    while( j >= 0 && A[j] > A[j+1])
        swap(A[j], A[j+1])
        j--;
```

$O(N^2)$
 $O(1)$

\Rightarrow Given array of N ele, rearrange the array s.t.

All ele $\leq A[0]$ go to left

All ele $> A[0]$ go to right

{ 10, 3, 0, 15, 6, 12, 2, 10, 7, 15, 14 }

{ 3, 0, 6, 2, 7, 10, 15, 12, 10, 15, 14 }

{ 10, 3, 0, ~~15~~, 6, ~~12~~, ~~2~~, 10, ~~7~~, 15, 14 }

$\downarrow^x \quad \downarrow^e$
~~7~~ ~~2~~ ~~12~~ ~~15~~
 2 10 12 15

swap($A[0]$, $A[x]$)

```

partition (A[], int N)
{
    l = 1; r = N-1;
    while (l <= r)
    {
        if (A[l] <= A[r])
            l++;
        else if (A[r] > A[l])
            r--;
        else
            swap(A[l], A[r]);
            l++; r--;
    }
    swap(A[l], A[r]);
    return r;
}

```

\Rightarrow Given array of N ele, rearrange the array s.t. s to e
 All ele $\leq A[s]$ go to left
 All ele $> A[s]$ go to right

```

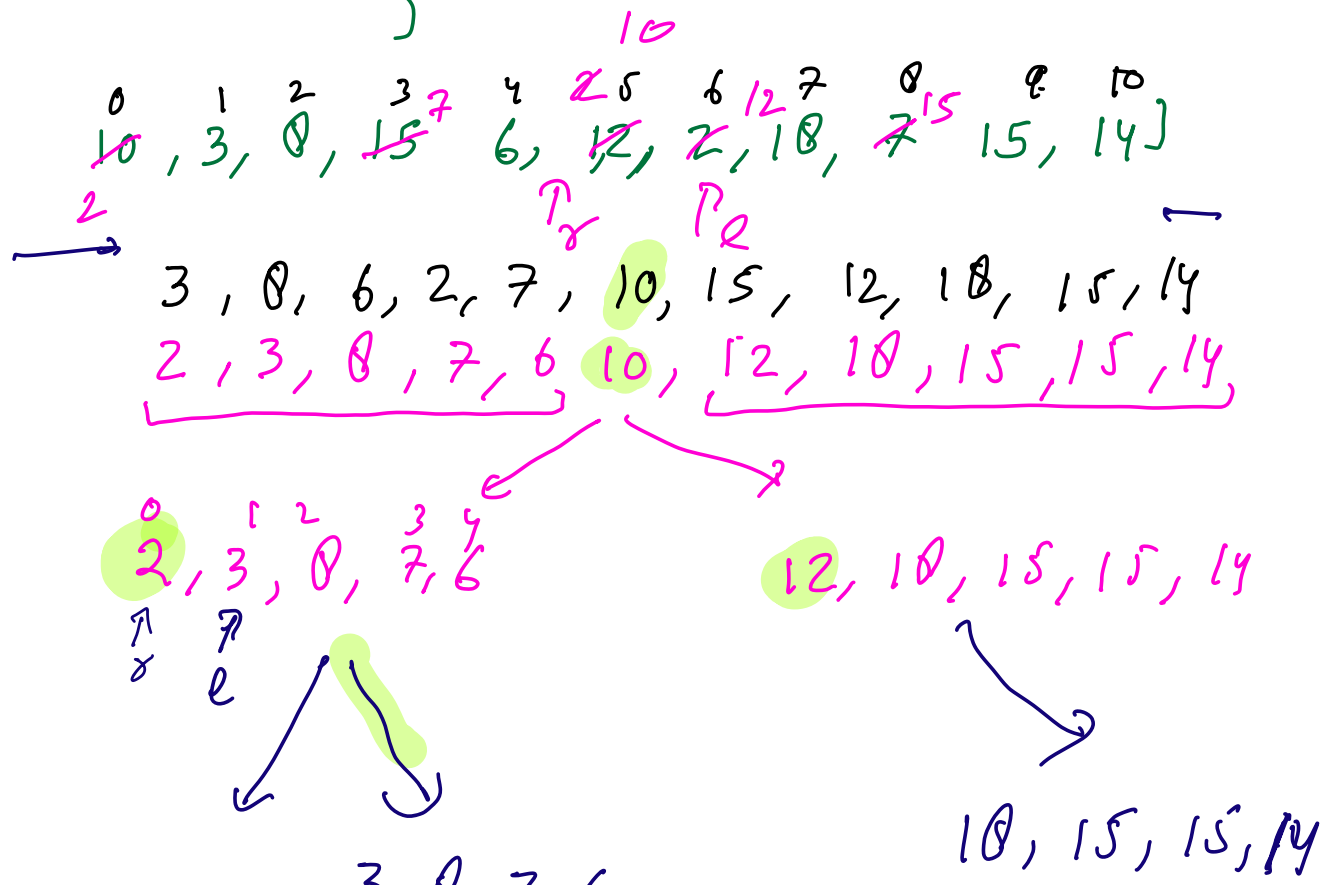
partition (A[], s, e)
{
    l = s+1; r = e;

```

```

while ( l <= r )
{
    if ( A[l] <= A[r] )
        l++;
    else if ( A[l] > A[r] )
        r--;
    else
        swap(A[l], A[r]);
        l++; r--;
}
swap( A[l], A[r] );
return r;
}

```



$$s, v, t, b$$

```
void QuickSort (int A[], int s, int e)
```

```
if ( s == e ) return;
int p = partition( A, s, e )
```

```

    quicksort(A, s, p-1);
    quicksort(A, p+1, se);
}

```

\downarrow \downarrow^e

$\{10, 3, 0, 15, 6, 12, 2, 18, 7, 15, 14\}$

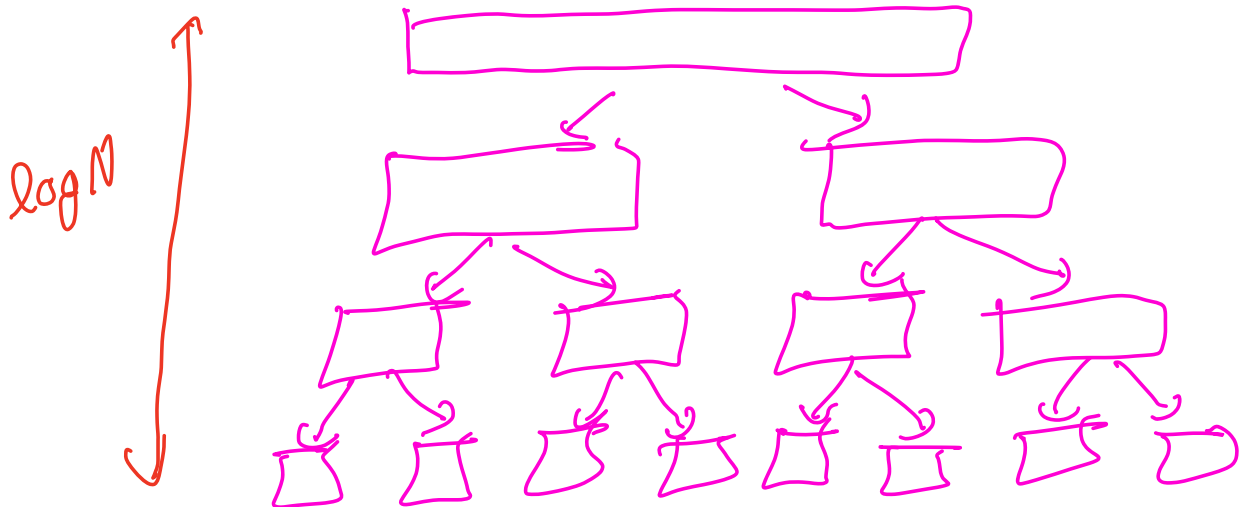
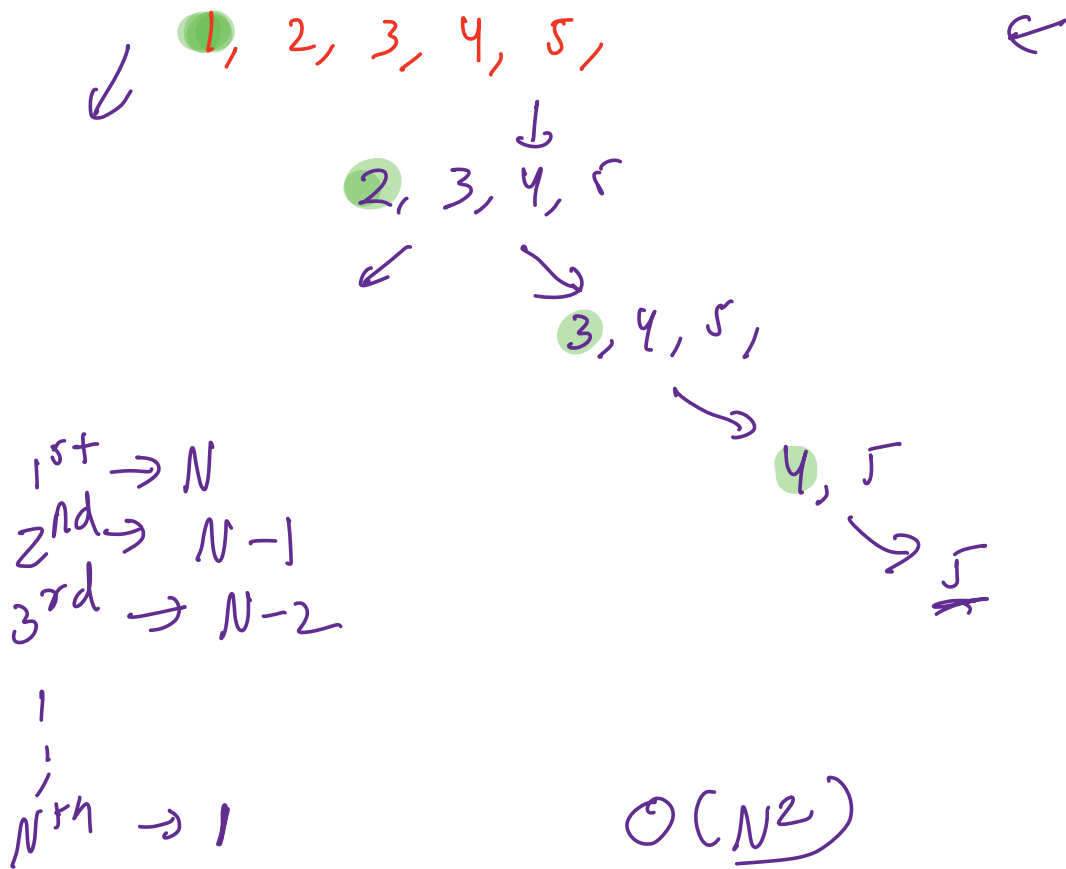
3, 0, 6, 2, 7, 10, 15, 12, 18, 15, 14

Diagram illustrating the selection of the root node for a Huffman tree. The initial list of frequencies is 2, 3, 6, 0, 7. The two smallest values, 2 and 3, are selected and combined into a new node with frequency 5. The updated list is 5, 6, 0, 7. The two smallest values, 0 and 5, are selected and combined into a new node with frequency 5. The final list is 5, 6, 7. The two smallest values, 5 and 6, are selected and combined into a new node with frequency 11. The final list is 11, 7. The two smallest values, 7 and 11, are selected and combined into the root node with frequency 18.

2, 3, 6, 7, 8, 10

15, 12, 14, 15, 18
 14, 12, 15 18
 12, 14
 12

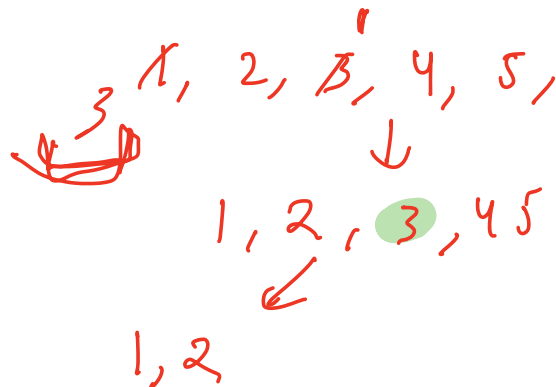
Break: 9:10



$$1 * N + 2 * \frac{N}{2} + 4 * \frac{N}{4} + \dots$$

$$N + N + N$$

$$O(N \log N)$$



$$\frac{1}{N} * \frac{1}{N-1} * \frac{1}{N-2} * \frac{1}{N-3} \dots \rightarrow \text{verrr}$$

① Choose random ele as pivot ele
 $p_i \Rightarrow \text{rand}()$ // [5, 6]

② swap (A[0], A[p_i])

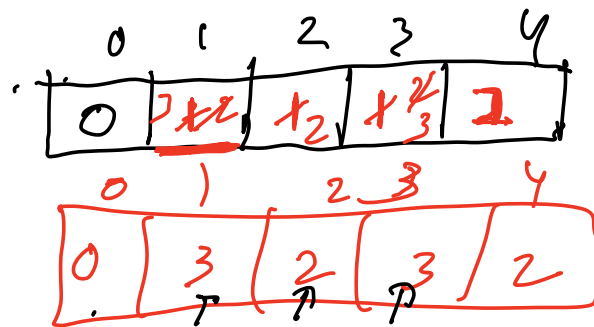
Best : $O(N \log N)$
 Avg : $O(N \log N)$
 Worst : $O(N^2)$

SC: $O(\log N)$
 $O(N)$

	Best	Worst	Avg	In place	Stable	Swap
Selection						
Bubble						
Merge						
Insertion						
Quicksort						

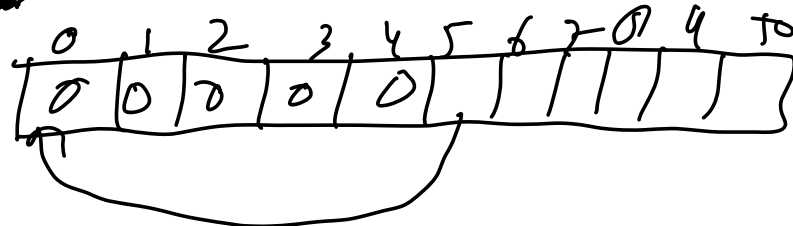
Count sort

ACD = { 3, 1, 4, 4, 2, 1, 3, 3, 2, 1 }



1, 1, 1, 2, 2, 3, 3, 3, 4, 4

5-10



$$5 - 5 = 0$$

$$6 - 5 = 1$$

$$10 - 6 = 4$$

$$[-10, 10]$$

$$-10 - (-10) = 0$$

$$-9 - (-10) = 1$$

$$-8 - (-10) = 2$$

1

⋮

$$10 \rightarrow 10 - (-10) = 20$$