Date. __/__/____

Selection sort

① Select min & swap from first.

② Select min & swap with arr[1] to arr[n]

13	14	24	52	20	9

① | 9 | 14 | 24 | 52 | 20 | 13 |

② | 9 | 13 | 24 | 52 | 20 | 14 |

③ | 9 | 13 | 14 | 24 | 20 | 24 |

④ | 9 | 13 | 14 | 20 | 52 | 24 |

⑤ | 9 | 13 | 14 | 20 | 24 | 52 |

pseudo code

```
for(int i=0; i<=n-2; i++) → min=i
    for(int j=i; j<=n-1; j++)
        if arr[j]<arr[min]
            min=j
    swap(arr[min], arr[i])
}
```

Date __/__/____

```
void selection_sort(int arr[], int n){
    for(int i=0; i<=n-2; i++){
        int min=i;
        for(int j=i; j<=n-1; j++){
            if arr[min] < arr[i]
            if arr[j] < arr[min]
                min=j
        }
        swap(arr[min], arr[i]);
    }
}
```

13	46	24	52	20	9
0	1	2	3	4	5

(1) i=0
min=0

j=0	j=1	j=2	j=3	j=4	j=5
13≠13	46≠13	24≠13	52≠13	20≠13	9<13
					min=5

9	46	24	52	20	13

(2) i=1
min=1

j=1	j=2	j=3	j=4	j=5
46≠46	24<46	52≠24	20<24	13<20
	min=2		min=4	min=5

9	13	24	52	20	46

③ i=2
j=2
min=2
j=2 j=3 j=4 j=5
24≠24 50≠24 20<24 46≠20
min=4

min=4

9|13|20|50|24|46

④ i=3
min=3
j=3 j=4 j=5
50≠50 24<50 46≠24
min=24

min=4

9|13|20|24|50|46

⑤ i=4 (n-2)
min=4
j=4 j=5
56≠50 46<56
min=5

9|13|20|24|46|50 → Ans

Date __/__/____

j loop

$n, n-1, n-2 \cdots 1$

$= \frac{n^2}{2} + \frac{n}{2}$

$= n^2$

Ex

$\Rightarrow O(n^2)$

Bubble Sort

Date: __/__/____

→ Push the largest to last by adjacent swaps

(1)

13, 46, 24, 52, 20, 9

↓ sorted

13, 46, 24, 52, 20, 9

↓ not s

13, 24, 46, 52, 20, 9

↓ s

13, 24, 46, 52, 20, 9

↓ not s

13, 24, 46, 20, 52, 9

↓ not s

13, 24, 46, 20, 9, 52

0 to n-1 for size n

(2)

13, 24, 46, 20, 9, 52

↓ s

13, 24, 46, 20, 9, 52

↓ s

13, 24, 46, 20, 9, 52

↓ ns

13, 24, 20, 46, 9, 52

↓ ns

13, 24, 20, 9, 46, 52

0 to n-2

Date: __/__/____

(3) 13, 24, 20, 9, 46, 52

13, 24, 20, 9, 46, 52 (ns)

13, 20, 24, 9, 46, 52 (ns)

13, 20, 9, 24, 46, 52

0 to n-3

(4) 13, 20, 9, 24, 46, 52

13, 20, 9, 24, 46, 52 (ns)

13, 9, 20, 24, 46, 52

0 to n-2

(5) 9, 13, 20, 24, 46, 52 } 0 to n-5 or 0-1

```
for(int i = n-1 ; i >= 1 ; i--)
    and did swap = 0;
    for(int j = 0; j <= i-1; j++)
        if (arr[j] > a[j+1])
            swap(a[j+1], a[j])
            did swap = 1
    if (did swap == 0)
        break
```

worst $O(n^2)$ → same best O(N) as selection sort

* If the array is in already ascending order we don't need to swap the rest so hence did swap.

Date: __/__/____

```
// Recursive
void bubble(int arr[], int n) {
    if (n == 1)
        return;
    for (int j = 0; j <= n-2; j++) {
        if (arr[j] > arr[j+1])
            swap(arr[j], arr[j+1]);
    }
    bubble(arr, n-1);
}
```

insertion sort

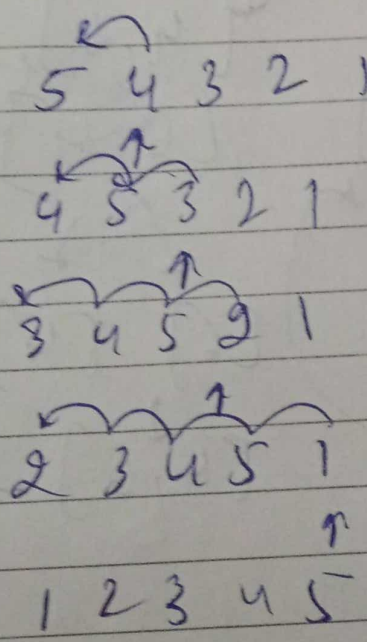
Date __/__/____

→ takes an element & places it in its correct order

```
for (i=0; i<=n-1; i++){
    j=i;
    while (j>0 && arr[j-1] > arr[j]){
        swap(arr[j+1], arr[j])
        j--;
    }
}
```

worst $O(n^2)$

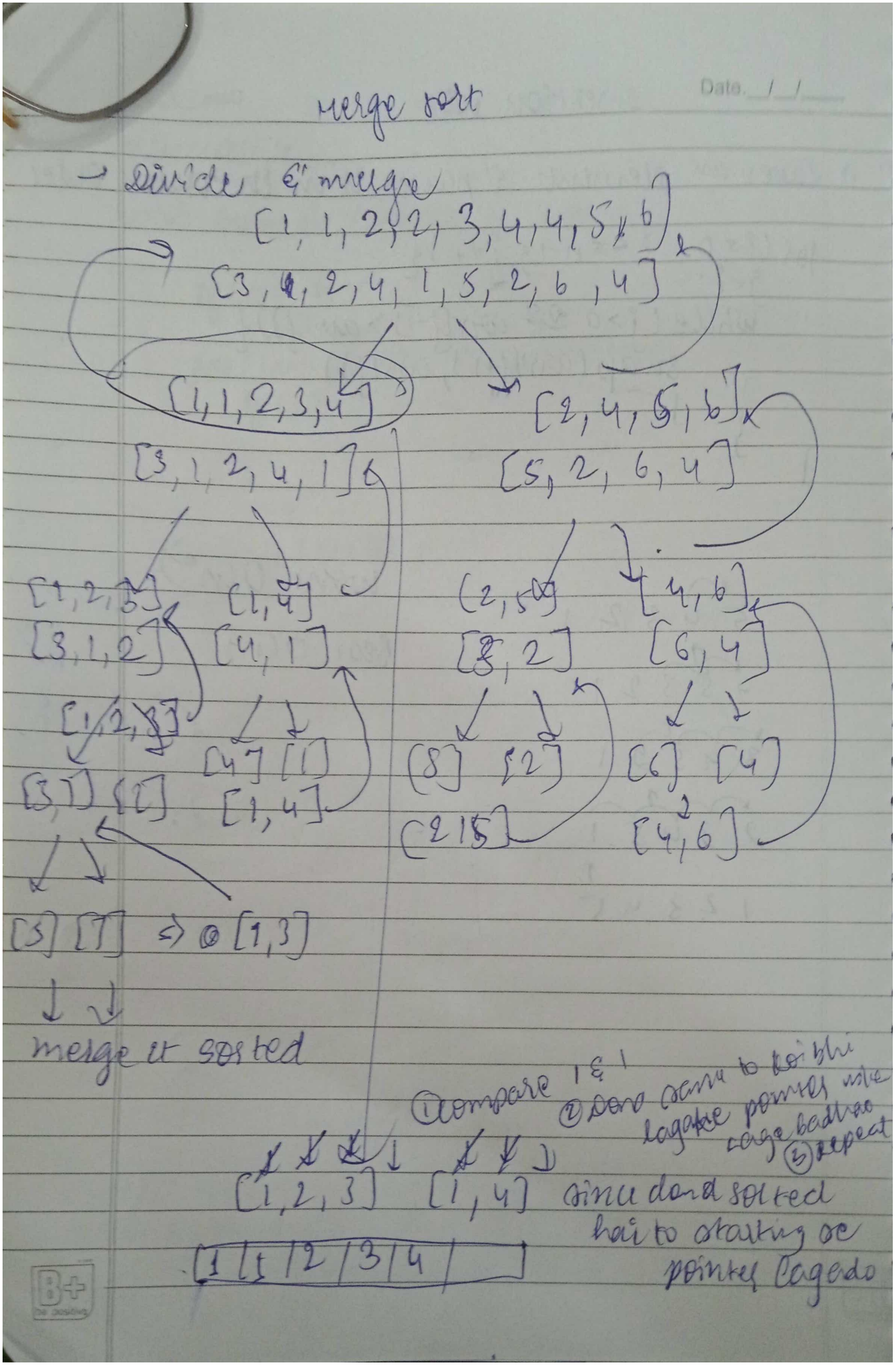
Best $O(n)$



merge sort

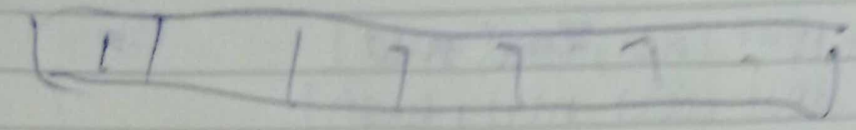
Date: / /

→ Divide & merge

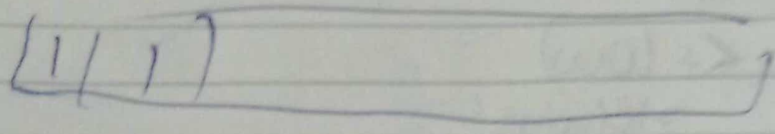


Date __/__/____

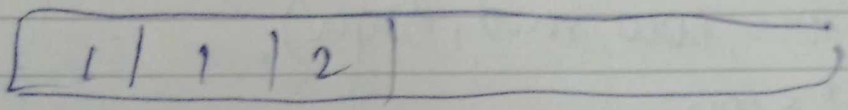
① [1 1 2 3 4] [2 4 5 6]



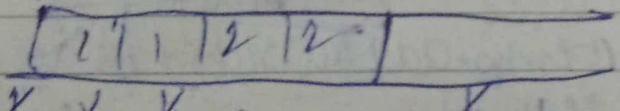
② (1, 1, 2, 3, 4) [2, 4, 5, 6]



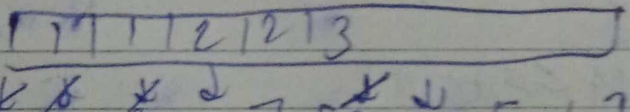
③ [1, 1, 2, 3, 4] [2, 4, 5, 6]



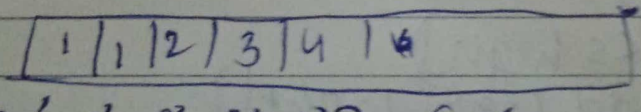
④ [1, 1, 2, 3, 4] [2, 4, 5, 6]



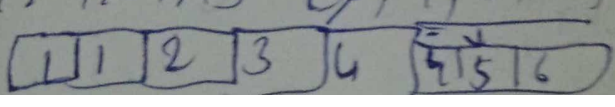
④ [1, 1, 2, 3, 4] [2, 4, 5, 6]



⑤ [1, 1, 2, 3, 4] [2, 4, 5, 6]



⑥ [1, 1, 2, 3, 4] [2, 4, 5, 6]



once any of the array ends just simply add the remaining elements

Date.__/__/____

```
                                    low  high
mergesort (arr, low, high)
  mid = (high + low)/2

  mergesort (arr, low, mid)
  mergesort (arr, mid+1, high)
  merge (arr, low, mid, high)
```

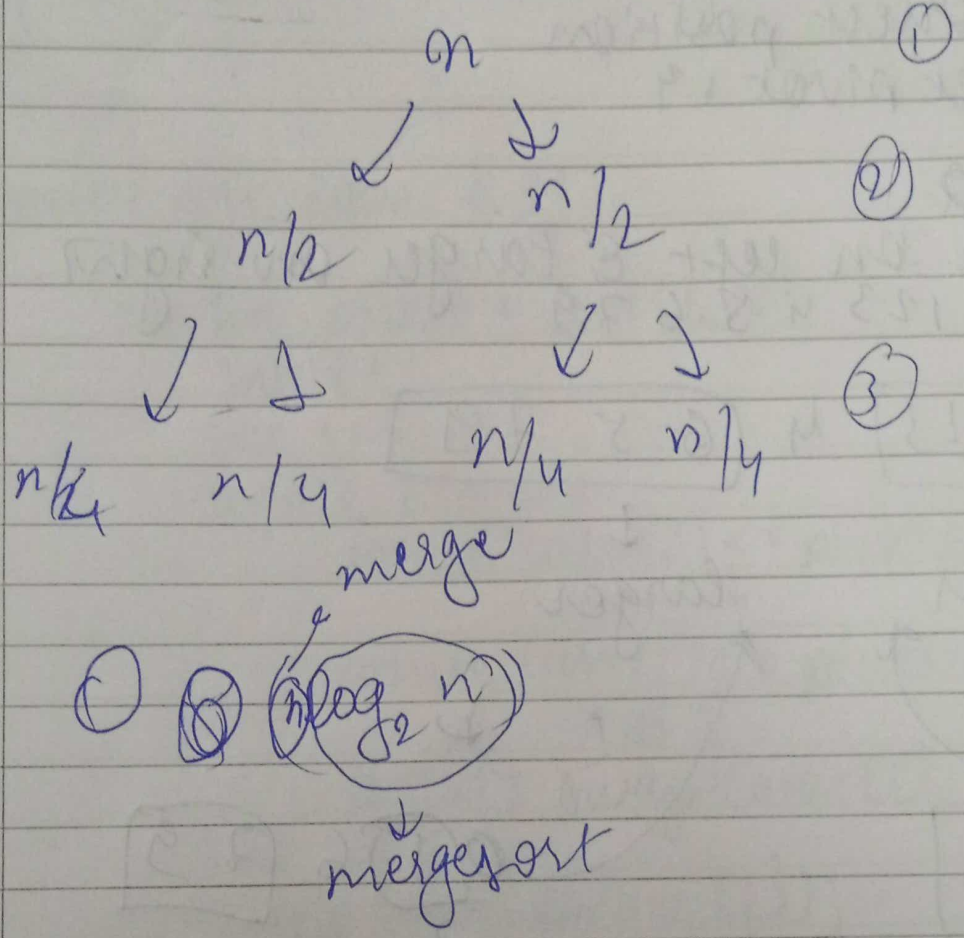
Base case :=

```
if (high <= low)
        return;
```

```
merge (arr, low, mid, high)
    left = low
    right = mid+1
temp = []
    while (left <= mid && right <= mid)
            if arr[left] <= arr[right]
                temp.add (arr[left])
                left ++;
            temp.add (arr[right])
            right ++;
    }
    while (left <= mid)
        temp.add (arr[left]);
            left ++;
    while (right <= high)
        temp.add (arr[right]);
        right++;
```

Date __/__/____

```
for(i=low ; i<=high ; i++){
    arr[i] = temp[i-low];
}
```



Space complexity : $O(n)$

↓

as introducing temp

Quick sort

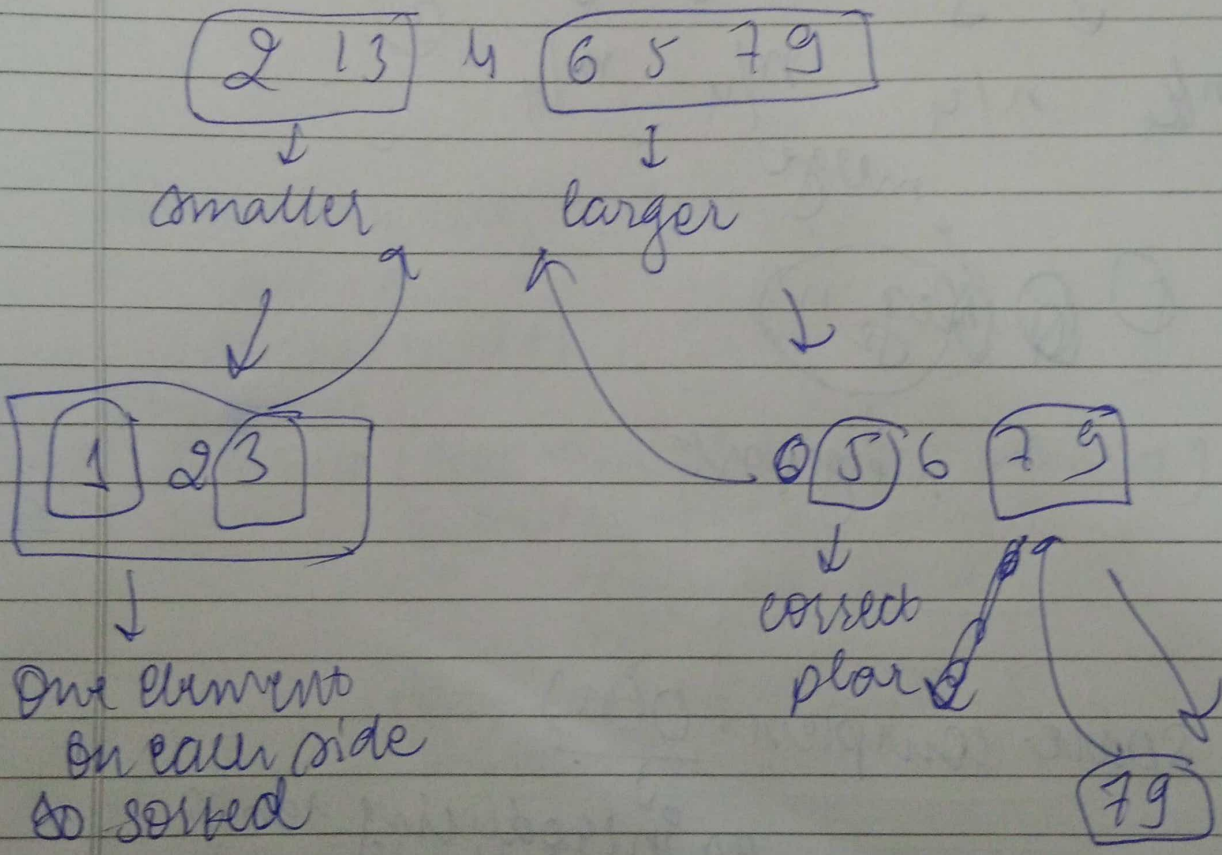
Date: __/__/____

$T \leq O(n \log n)$
$S = O(1)$

4 6 2 5 7 9 13

① Pick a pivot (any element) and place it in its correct position
Let pivot = 4

② Smaller on left & larger on right
1 2 3 4 5 6 7 9



```
qs(arr[], int l, int h)
    if (low < h)
        int parti = part(arr, l, h);
        qs(arr, low, parti-1);
        qs(arr, parti+1, high);
}
}

part(arr, low, h){
    int pivot = arr[low];
    int i = l;
    int j = h;
    while(i<j){
        while(arr[i] <= pivot && i<=high-1){
            i++;}
        while(arr[j] > pivot && j>=low+1){
            j--;}
        if(i<j) swap(arr[i], arr[j]);
    }
    swap(arr[l], arr[j]);
    return j;
}
```