Second Largest Element in an Array

Array = {10, 20, 4, 20, 9}

Output → 10 (second largest Element)

① **Bruteforce approach:**

i) Pehle array Ko ascending mein sort Kar lo.

ii) last element Ko pakdo → ye hi sabse bada hai.

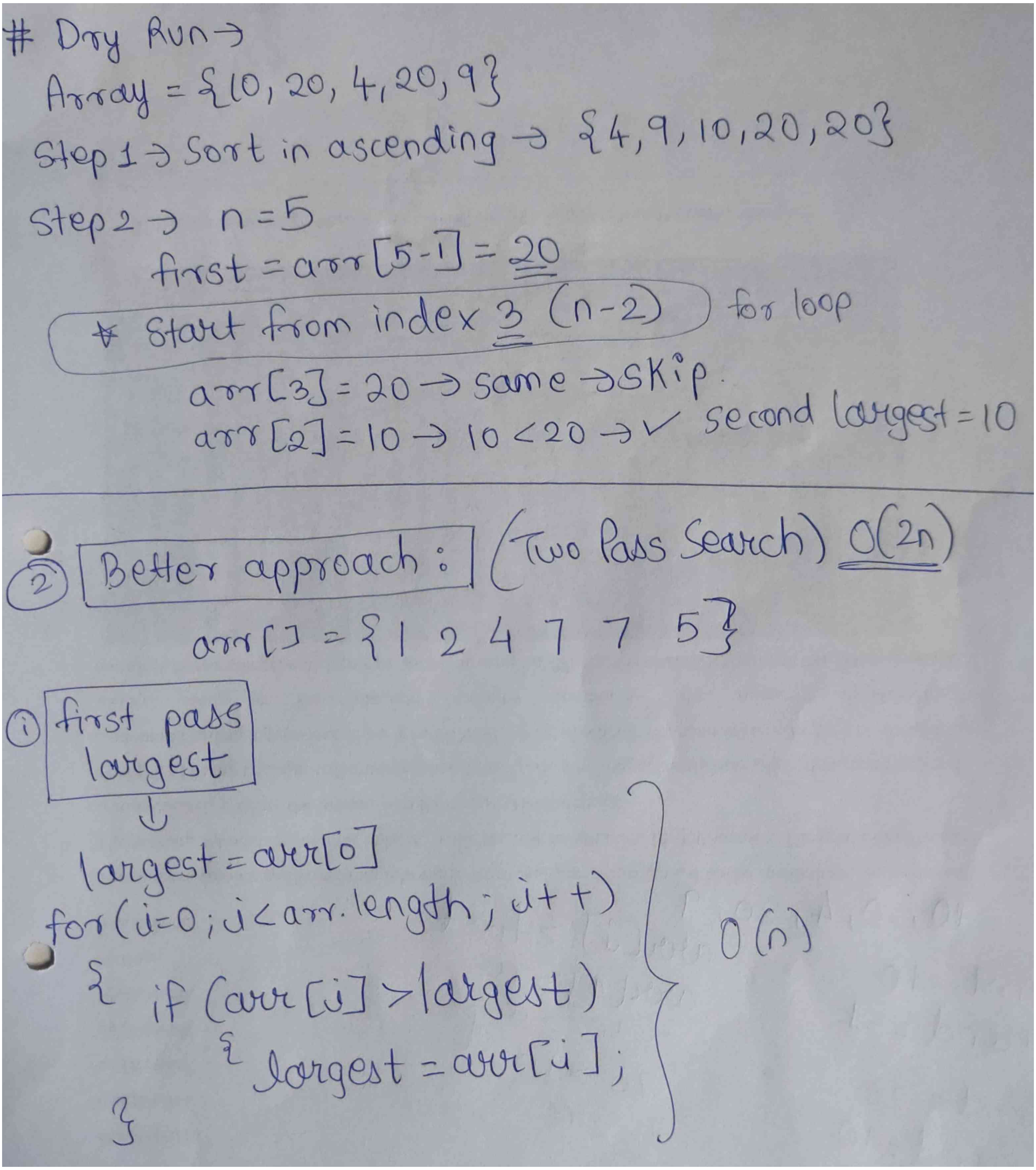
iii) Ab peeche se chalo (second last se)
→ pehla aisa element dhoondo jo is se chhota ho.

iv) Wohi second largest hoga.

v) Agar nahi mila to → "second largest nahi mila"

Soln :→
```
int[] arr = {10, 20, 4, 20, 9};
Arrays.sort(arr);
int n = arr.length;
int first = arr[n-1]; // first largest.

// Traverse backwards to find 2nd largest.
for(int i=n-2; i>=0; i--)
{ if (arr[i] < first)
      ~~sop("2nd largest →"~~ + arr[i]);
          return
}
//if not found
return -1
```



Dry Run →

Array = {10, 20, 4, 20, 9}

Step 1 → Sort in ascending → {4, 9, 10, 20, 20}

Step 2 → n = 5

first = arr[5-1] = $\underline{\underline{20}}$

* Start from index $\underline{\underline{3}}$ (n-2) for loop

arr[3] = 20 → same → skip.

arr[2] = 10 → 10 < 20 → ✓ second largest = 10

(2) Better approach : (Two Pass Search) $\underline{\underline{O(2n)}}$

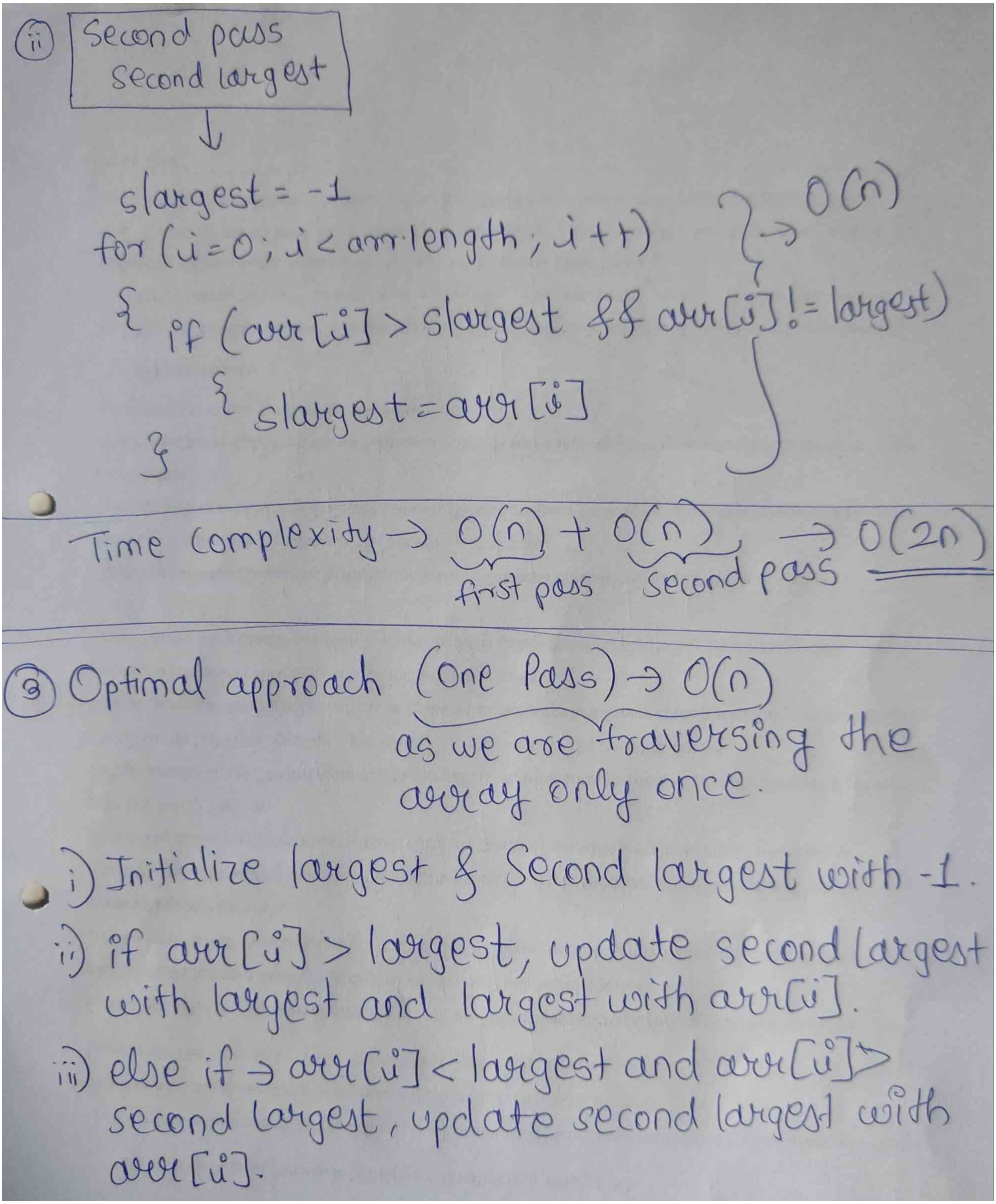
arr[] = {1 2 4 7 7 5}

(i) first pass largest

↓

```
largest = arr[0]
for(i=0; i<arr.length; i++)
{ if (arr[i] > largest)
      { largest = arr[i];
}
```

O(n)



(ii) Second pass
Second largest

↓

```
slargest = -1
for (i=0; i<arr.length; i++)            } → O(n)
{ if (arr[i] > slargest && arr[i] != largest)
      { slargest = arr[i]
}
```

$\text{Time complexity} \rightarrow \underbrace{O(n)}_{\text{first pass}} + \underbrace{O(n)}_{\text{second pass}}, \rightarrow \underline{\underline{O(2n)}}$

③ Optimal approach $\underbrace{\text{(One Pass)} \rightarrow O(n)}_{\text{as we are traversing the array only once.}}$

i) Initialize largest & Second largest with -1.

ii) if arr[i] > largest, update second largest with largest and largest with arr[i].

iii) else if → arr[i] < largest and arr[i] > second largest, update second largest with arr[i].

```
Sol^n:→ Largest = -1
        SecondLargest = -1

      for (int i = 0; i<n; i++)
      {
      → if (arr[i] > largest)
          {
            largest = arr[i];
            secondlargest = largest;
          }
      → else if (arr[i]< largest && arr[i]>secondlargest)
          {
            secondlargest = arr[i];
          }
      }
      return secondlargest;
```

**

first →
secondLargest = largest;

second →
Largest = arr[i]

Dry Run: arr =

12	35	1	10	34	1
0	1	2	3	4	5

(i) largest = -1
Secondlargest = -1

arr[i] → arr[0] = 12

if → 12 > -1
* largest = 12
Secondlargest = -1

(ii) largest = 12
Secondlargest = -1

arr[i] → arr[1] = 35

if → 35 > 12
* largest = 35
Seondlargest = 12

(iii) largest = 35
secondlargest = 12

(2) arr[i] → arr[2] = 1

if → 1 > 35 → ✗

else if → 1 < 35 && 1 > 12 → ✗

no change

(iv) largest = 35
secondlargest = 12

arr[i] → arr[3] = 10

if → 10 > 35 → ✗

else if → 10 < 35 && 10 > 12 → ✗

no change

(v) largest = 35
secondlargest = 12
arr[i] → arr[4] = 34

if → 34 > 35 → ✗

else if → 34 < 35 && 34 > 12 → ✓
condition matched

largest = 35
→ secondlargest = 34

(vi) largest = 35
secondlargest = 34

arr[i] → arr[5] = 1

if → 1 > 35 → ✗

else if → 1 < 35 && 1 > 34 → ✗

↓
no change

↓

finally we are having
largest → 35
secondlargest → 34