```java
Arrays.sort(arr);

int distinctCount = 0;
int distinctElem = Integer.MIN_VALUE;

for(int i = n-1; i >= 0; i-){
    if(arr[i] != distinctElem){
        distinctCount++;
        distinctElem = arr[i];
    }
    if(distinctCount == 3){
        System.out.println(arr[i]);
        return;
    }
}

// distinct elems fewer than 3
System.out.println(arr[n-1]);
```

arr = | 7 | 3 | 3 | 1 | 7 | 8 | n = 6

0	1	2	3	4	5
1	3	3	7	7	8

arr =

i = 5 ≥ 0

if(8 != -∞) T

i = 4 ≥ 0

if(7 != 8) T

i = 3 ≥ 0

if(7 != 7) F

i = 2 ≥ 0

if(3 != 7) T

count = ~~0~~ 1 2 3

elm = -~~∞~~ 8 7 3

count == 3

3 == 3 T

o/p → 3

c = 1 2

c = 1

1 | 1 | 2

[TC → O(n log n) + n]

Maximum of 3 products

Arrays.sort(arr)

-7	-5	2	3	4

1 2 3 4 5

-7 -5 2 4

```
int p1 = arr[n-1] * arr[n-2] * arr[n-3];
int p2 = arr[0] * arr[1] * arr[n-1];
return Math.max(p1, p2);
```

2 x 3 x 4 = 24

-7 x -5 x 2 = 70

-7 x -5 x 4 = 140

-5 | -4 | -3 | -2 | -1

-3 x -4 x -5 = -60

-1 x -2 x -5 = -10

arr →

-7	5	4	3	1

→

-7	1	3	4	5

p1 = 5 x 4 x 3 = 60

p2 = -7 x 1 x 5 = -35

Max(60, -35)

O/P → 60