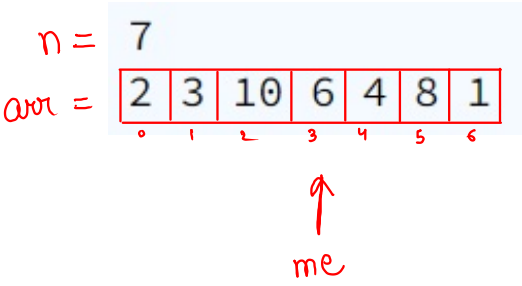
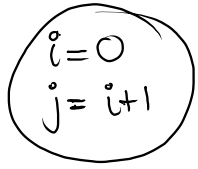
Possibilities

Code Structure

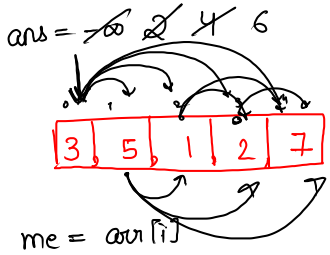
- Combination without Repetation — (i = 0, j = i+1)
- Combination with Repetation — (i = 0, j = i)
- Permutation without Repetation — (i = 0, j = 0, if (i != j))
- Permutation with Repetation — (i = 0, j = 0)

maximum difference between the two elements






```java
public static void main(String[] args) {
    Scanner scn = new Scanner(System.in);
    int n = scn.nextInt();
    int[] arr = new int[n];
    for (int i = 0; i < n; i++) {
        arr[i] = scn.nextInt();
    }

    System.out.println(maxDiff(arr));
}
public static int maxDiff(int[] arr) {
    int n = arr.length;
    int ans = Integer.MIN_VALUE;
    for (int i = 0; i < n; i++) {
        for (int j = i + 1; j < n; j++) {
            if ( arr[i] < arr[j] ) {
                int diff = arr[j] - arr[i];
                if ( diff > ans ) {
                    ans = diff;
                }
            }
        }
    }
    return ans;
}
```



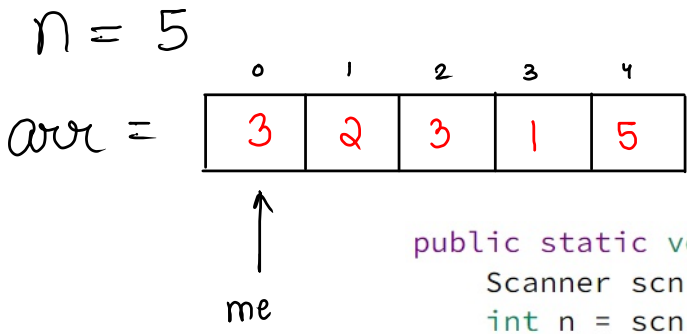
me = 3
diff = 5 - 3 = 2
diff = 7 - 3 = 4

me = 5
diff = 7 - 5 = 2

me = 1
diff = 2 - 1 = 1
diff = 7 - 1 = 6

me = 2
diff = 7 - 2 = 5

Find Duplicate 3




```java
public static void main(String[] args) {
    Scanner scn = new Scanner(System.in);
    int n = scn.nextInt();
    int[] arr = new int[n];
    for (int i = 0; i < n; i++) {
        arr[i] = scn.nextInt();
    }

    System.out.println(findDuplicates(arr));
}
public static boolean findDuplicates(int[] arr) {
    int n = arr.length;
    for (int i = 0; i < n; i++) {
        for (int j = i + 1; j < n; j++) {
            if ( arr[i] == arr[j] ) { // myself == other
                return true;
            }
        }
    }
    return false;
}
```

Double Occurence

$n = 5$

arr1 =

0	1	2	3	4
2	3	5	1	4

(unique)

$m = 9$

arr2 =

0	1	2	3	4	5	6	7	8
3	5	7	1	8	1	4	3	9

dry run

i=0, count=0

i=1, count=0 X 2

i=2, count=0 1

i=3, count=0 X 2

i=4 count=0 1

i=5 X

O/P

3

1

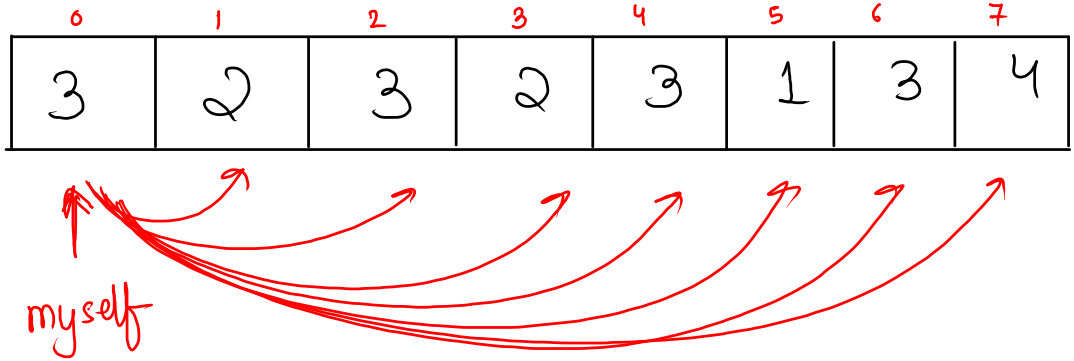
Code

```java
public static void main(String[] args) {
    Scanner scn = new Scanner(System.in);
    int n = scn.nextInt();
    int[] arr1 = new int[n];
    for (int i = 0; i < n; i++) {
        arr1[i] = scn.nextInt();
    }

    int m = scn.nextInt();
    int[] arr2 = new int[m];
    for (int i = 0; i < m; i++) {
        arr2[i] = scn.nextInt();
    }

    doubleOccurance(arr1, n, arr2, m);
}
public static void doubleOccurance(int[] arr1, int n, int[] arr2, int m) {
    for (int i = 0; i < n; i++) {
        int count = 0;
        for (int j = 0; j < m; j++) {
            if ( arr1[i] == arr2[j] ) {
                count++;
            }
        }
        if ( count == 2 ) {
            System.out.print( arr1[i] + " " );
        }
    }

}
```

Max Count 3

(here, we need to store number along with count

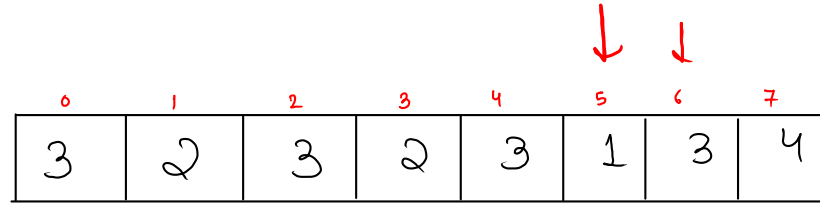
arr =



count = ~~1 2 3~~ 4

num = 3

```java
public static void main(String[] args) {
    Scanner scn = new Scanner(System.in);
    int n = scn.nextInt();
    int[] arr = new int[n];
    for (int i = 0; i < n; i++) {
        arr[i] = scn.nextInt();
    }

    System.out.println(maxCount(arr));
}
public static int maxCount(int[] arr) {
    int n = arr.length;
    int max_count = 0;
    int num = 100001;
    for (int i = 0; i < n; i++) {
        int count = 0;
        for (int j = i; j < n; j++) {
            if ( arr[i] == arr[j] ) {
                count++;
            }
        }

        if ( count > max_count ) {
            max_count = count;
            num = arr[i];
        }
    }
    return num;
}
```



max_count = ~~0~~ 4

num = ~~100001~~ 3

i = 0, count = ~~0~~ ~~1~~ ~~2~~ ~~3~~ 4

i = 1, count = ~~0~~ 2

i = 2, count = 3

i = 3, count = 1

i = 4, count = 2

i = 5, count = 1

i = 6, count = 1

i = 7, count = 1