Q 1==➔

```java
public class DuplicateElement {
    public static void main(String[] args) {
        int [] arr = new int [] {1, 2, 3, 4, 2, 7, 8, 8, 3};

        System.out.println("Duplicate elements in given array: ");

        for(int i = 0; i < arr.length; i++) {
            for(int j = i + 1; j < arr.length; j++) {
                if(arr[i] == arr[j])
                    System.out.println(arr[j]);
            }
        }
    }
}
```

Q 2==➔

```java
public class Quick
{
int partition (int a[], int start, int end)
{
    int pivot = a[end];
    int i = (start - 1);

    for (int j = start; j <= end - 1; j++)
    {

        if (a[j] < pivot)
        {
            i++;
            int t = a[i];
            a[i] = a[j];
            a[j] = t;
        }
    }
    int t = a[i+1];
    a[i+1] = a[end];
```

```
21.     a[end] = t;
22.     return (i + 1);
23. }
24.
25.
26. void quick(int a[], int start, int end)
27. {
28.     if (start < end)
29.     {
30.         int p = partition(a, start, end);
31.         quick(a, start, p - 1);
32.         quick(a, p + 1, end);
33.     }
34. }
35.
36.
37. void printArr(int a[], int n)
38. {
39.     int i;
40.     for (i = 0; i < n; i++)
41.         System.out.print(a[i] + " ");
42. }
43.     public static void main(String[] args) {
44.     int a[] = { 13, 18, 27, 2, 19, 25 };
45.     int n = a.length;
46.     System.out.println("\nBefore sorting array elements are - ");
47.     Quick q1 = new Quick();
48.     q1.printArr(a, n);
49.     q1.quick(a, 0, n - 1);
50.     System.out.println("\nAfter sorting array elements are - ");
51.     q1.printArr(a, n);
52.     System.out.println();
53.     }
54. }
```

Q 3==➔

```
1.          public class Bubble {
```

```
2.      static void print (int a[]) //function to print array elements
3.  {
4.      int n = a.length;
5.      int i;
6.      for (i = 0; i < n; i++)
7.      {
8.          System.out.print(a[i] + " ");
9.      }
10. }
11. static void bubbleSort (int a[])    // function to implement bubble sort
12. {
13.     int n = a.length;
14.     int i, j, temp;
15.     for (i = 0; i < n; i++)
16.     {
17.         for (j = i + 1; j < n; j++)
18.         {
19.             if (a[j] < a[i])
20.             {
21.                 temp = a[i];
22.                 a[i] = a[j];
23.                 a[j] = temp;
24.             }
25.         }
26.     }
27. }
28. public static void main(String[] args) {
29. int a[] = {35, 10, 31, 11, 26};
30. Bubble b1 = new Bubble();
31. System.out.println("Before sorting array elements are - ");
32. b1.print(a);
33. b1.bubbleSort(a);
34. System.out.println();
35. System.out.println("After sorting array elements are - ");
36. b1.print(a);
37.
38. }
```

39.}

Q 4===➔

```
class MergeSort {

    void merge(int arr[], int l, int m, int r)
    {

        int n1 = m - l + 1;
        int n2 = r - m;

        int L[] = new int[n1];
        int R[] = new int[n2];

        for (int i = 0; i < n1; ++i)
            L[i] = arr[l + i];
        for (int j = 0; j < n2; ++j)
            R[j] = arr[m + 1 + j];



        int i = 0, j = 0;


        int k = l;
        while (i < n1 && j < n2) {
            if (L[i] <= R[j]) {
                arr[k] = L[i];
                i++;
            }
            else {
                arr[k] = R[j];
                j++;
            }
            k++;
        }


        while (i < n1) {
            arr[k] = L[i];
            i++;
            k++;
        }


        while (j < n2) {
```

```
            arr[k] = R[j];
            j++;
            k++;
        }
    }


    void sort(int arr[], int l, int r)
    {
        if (l < r) {

            int m = l + (r - l) / 2;


            sort(arr, l, m);
            sort(arr, m + 1, r);


            merge(arr, l, m, r);
        }
    }


    static void printArray(int arr[])
    {
        int n = arr.length;
        for (int i = 0; i < n; ++i)
            System.out.print(arr[i] + " ");
        System.out.println();
    }


    public static void main(String args[])
    {
        int arr[] = { 12, 11, 13, 5, 6, 7 };

        System.out.println("Given Array");
        printArray(arr);

        MergeSort ob = new MergeSort();
        ob.sort(arr, 0, arr.length - 1);

        System.out.println("\nSorted array");
        printArray(arr);
    }
}
```

Q 5===➔

```
1.  public class Selection
2.  {
3.      void selection(int a[])
4.  {
5.      int i, j, small;
6.      int n = a.length;
7.      for (i = 0; i < n-1; i++)
8.      {
9.          small = i;
10.
11.         for (j = i+1; j < n; j++)
12.         if (a[j] < a[small])
13.             small = j;
14.     int temp = a[small];
15.     a[small] = a[i];
16.     a[i] = temp;
17.     }
18.
19. }
20. void printArr(int a[])
21. {
22.     int i;
23.     int n = a.length;
24.     for (i = 0; i < n; i++)
25.     System.out.print(a[i] + " ");
26. }
27.
28.     public static void main(String[] args) {
29.     int a[] = { 91, 49, 4, 19, 10, 21 };
30.     Selection i1 = new Selection();
31.     System.out.println("\nBefore sorting array elements are - ");
32.     i1.printArr(a);
33.     i1.selection(a);
34.     System.out.println("\nAfter sorting array elements are - ");
35.     i1.printArr(a);
```

```
36.    System.out.println();
37.    }
38.}
```

Q 6===➔

```
class Launch {


    static boolean isSubset(int arr1[], int arr2[], int m,
                            int n)
    {
        int i = 0;
        int j = 0;
        for (i = 0; i < n; i++) {
            for (j = 0; j < m; j++)
                if (arr2[i] == arr1[j])
                    break;


            if (j == m)
                return false;
        }


        return true;
    }


    public static void main(String args[])
    {
        int arr1[] = { 11, 1, 13, 21, 3, 7 };
        int arr2[] = { 11, 3, 7, 1 };

        int m = arr1.length;
        int n = arr2.length;

        if (isSubset(arr1, arr2, m, n))
            System.out.print("arr2[] is "
                             + "subset of arr1[] ");
        else
            System.out.print("arr2[] is "
                             + "not a subset of arr1[]");
    }
}
```