Assignment number – 2

NAME-AJAY KUMAR

E-MAIL:-ajaykr4598@gmail.com

1. WAP to find the duplicates present in an array.

```
Sol:- public class DuplicateElement {

        public static void main(String[] args) {
            int[] a= {7,5,2,2,3,1,1,4,4,6,6};
            System.out.print("Dublicate elements are :");
            for(int i=0;i<a.length;i++)
            {
                for(int j=i+1;j<a.length;j++)
                {
                    if(a[i]==a[j])
                    {
                        System.out.print(a[j]+" ");
                        break;
                    }
                }
            }

        }

    }
```

2. WAP to sort an array using Quick Sort Algorithm.

```
Sol:- public class QuickSorting {
        int partition(int arr[], int low, int high)
     {

        int pivot = arr[high];
        int i = (low-1);
        for (int j=low; j<high; j++)
        {

            if (arr[j] <= pivot)
            {
                i++;

                int temp = arr[i];
                arr[i] = arr[j];
                arr[j] = temp;
            }
        }


        int temp = arr[i+1];
        arr[i+1] = arr[high];
        arr[high] = temp;

        return i+1;
     }
```

```
    void sort(int arr[], int low, int high)
    {
        if (low < high)
        {

            int pi = partition(arr, low, high);


            sort(arr, low, pi-1);
            sort(arr, pi+1, high);
        }
    }

      public static void main(String[] args) {
                    int arr[]= {60,44,55,77,88};
                    int l=0;
                    int h=arr.length-1;
                    QuickSorting ob = new QuickSorting();
                     ob.sort(arr, l, h);
                   System.out.println("Sorted array");
                     for(int elem:arr) {
                            System.out.print(elem+" ");
  }

      }

}
```

3. WAP to sort an array using Bubble Sort Algorithm.

```
Sol:- public class BubbleSorting {

      public static void main(String[] args) {
             int[] a= {7,5,2,3,1,4,6};

             for(int i=0;i<a.length;i++)
             {
                    for(int j=1;j<a.length-i;j++)
                    {
                          if(a[j]<a[j-1])
                          {
                                int tempVar=a[j];
                                a[j]=a[j-1];
                                a[j-1]=tempVar;
                          }
                    }
             }
             System.out.println("Sorted array");
             for(int elem:a)
             {
                    System.out.print(elem+" ");
             }
      }
}
```

4. WAP to sort an array using Merge Sort Algorithm.

Sol:-

```
public 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);
```

```
            }
        }
          public static void main(String[] args) {
                 int arr[] = { 12, 11, 13, 5, 6, 7 };
                 MergeSort ob = new MergeSort();
                  ob.sort(arr, 0, arr.length - 1);
                  System.out.println("Sorted Array");
                  for(int elem:arr) {
                       System.out.print(elem+" ");
                  }

          }

}
```

5. WAP to sort an array using Selection Sort Algorithm.

```
Sol:- public class SelectionSorting {

        public static void main(String[] args) {
                int[] a= {7,5,2,3,1,4,6,8};

                for(int i=0;i<a.length-1;i++)
                {
                        int smallest=i;
                        for(int j=i+1;j<a.length;j++)
                        {
                                if(a[smallest]>a[j])
                                {
                                        smallest=j;

                                }
                        }
                        int tempVar=a[smallest];
                        a[smallest]=a[i];
                        a[i]=tempVar;
                }
                System.out.println("Sorted array");
                for(int elem:a)
                {
                        System.out.print(elem+" ");
                }
        }

        }
```

6. WAP to check whether an array is a subset of another array.

```
Sol:- public class SubsetArray {

        public static void main(String[] args) {
                int[] ar= {10,11,21,34,46};
                int[] arr= {10,11,3};
                int count=0;
                for(int i=0;i<ar.length;i++) {
                        for(int j=0;j<arr.length;j++) {
                                if(ar[i]==arr[j]) {
                                        count++;
                                        continue;
                }
            }
        }
        if(count==arr.length) {
            System.out.println("arr is a subset of ar");
        }
        else
        {
            System.out.println("arr is not a subset of ar");
        }
 }
}
```