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

public class duplicateArrayElement {

public static void main(String[] args) {

int [] nums = new int [] {1, 2, 3, 4, 2, 7, 8, 8, 3};

System.out.println("Duplicate Elements: ");

for(int i = 0; i < nums.length; i++) {

for(int j = i + 1; j < nums.length; j++) {

if(nums[i] == nums[j])

System.out.print(nums[j]+" ");

}

}

}

}

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

class QuickSort {

static int partition(int array[], int low, int high) {

int pivot = array[high];

int i = (low - 1);

for (int j = low; j < high; j++) {

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

int temp = array[i];

array[i] = array[j];

array[j] = temp;

}

}

int temp = array[i + 1];

array[i + 1] = array[high];

array[high] = temp;

return (i + 1);

}

static void quickSort(int array[], int low, int high) {

if (low < high) {

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

quickSort(array, low, pi - 1);

quickSort(array, pi + 1, high);

}

}

}

class Main {

public static void main(String args[]) {

int[] arr = { 0, 8, 97, 24, 11, 10, 96, 2 };

System.out.println("Given Array: "+Arrays.toString(arr));

int size = arr.length;

QuickSort.quickSort(arr, 0, size - 1);

System.out.println("Sorted Array: "+Arrays.toString(arr));

}

}

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

import java.util.\*;

public class BubbleSort {

static void bubbleSort(int[] arr) {

for(int i=0; i < arr.length; i++){

for(int j=1; j < (arr.length-i); j++){

if(arr[j-1] > arr[j]){

int temp = arr[j-1];

arr[j-1] = arr[j];

arr[j] = temp;

}

}

}

}

public static void main(String[] args) {

int arr[] ={0, 8, 97, 24, 11, 10, 96, 2 };

System.out.println("Given Array: "+Arrays.toString(arr));

bubbleSort(arr);

System.out.println("\nSorted Array: "+Arrays.toString(arr));

}

}

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

import java.util.\*;

class MergeSort {

static 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, 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++;

}

}

static 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[] = { 0, 8, 97, 24, 11, 10, 96, 2 };

System.out.println("Given Array: "+Arrays.toString(arr));

sort(arr, 0, arr.length - 1);

System.out.println("\n Sorted array: "+Arrays.toString(arr));

}

}

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

import java.util.\*;

public class SelectionSort{

public static void selectionSort(int[] arr){

for (int i = 0; i < arr.length - 1; i++) {

int index = i;

for (int j = i + 1; j < arr.length; j++){

if (arr[j] < arr[index]){

index = j;

}

}

int smallerNumber = arr[index];

arr[index] = arr[i];

arr[i] = smallerNumber;

}

}

public static void main(String a[]){

int[] arr = {0, 8, 97, 24, 11, 10, 96, 2 };

System.out.println("Given Array: "+Arrays.toString(arr));

selectionSort(arr);

System.out.println("\n Sorted Array: "+Arrays.toString(arr));

}

}

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

class SubsetArray {

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[] = { 0, 8, 97, 24, 11, 10, 96, 2 };

int arr2[] = { 11, 2, 97, 24 };

if (isSubset(arr1, arr2, arr1.length, arr2.length))

System.out.print("Array 2 is subset of Array 1 ");

else

System.out.print("Array 2 is not a subset of Array 1 ");

}

}