Arrays programs

1)WAP on intersection of two array?

output:[2, 5]

```
package Array_prgm;

import java.util.ArrayList;

public class intersection_2arrays {
 public static void main(String[] args) {
  int[] arr1= {1,2,4,5};
  int[] arr2= {6,2,5,3};
  ArrayList<Integer> intersection = new ArrayList<Integer>();
  for (int i = 0; i < arr1.length; i++) {
   for (int j = 0; j < arr2.length; j++) {
     if (arr1[i] == arr2[j] && !intersection.contains(arr1[i])) {
                    intersection.add(arr1[i]);
   }
  }
 }
  System.out.println(intersection);
}
}
```

2)WAP on union of two array?

output:[1, 2, 4, 5, 6, 3]

```
package Array_prgm;

import java.util.ArrayList;

public class union {

 public static void main(String[] args) {
  int[] arr1= {1,2,4,5};
  int[] arr2= {6,2,5,3};
  ArrayList<Integer> union = new ArrayList<Integer>();
  for (int i = 0; i < arr1.length; i++) {
   if(!union.contains(arr1[i])) {
    union.add(arr1[i]);
   }
  }
  for (int j = 0; j < arr2.length; j++) {
   if(!union.contains(arr2[j])) {
    union.add(arr2[j]);
   }
  }
  System.out.println(union);
 }

}
```

3)Merge two unsorted array and sort it with out sort inbuilt method?

output:Sorted Merged Array: [1, 2, 3, 4, 6, 6, 7, 8, 9]

```
package Array_prgm;

import java.util.Arrays;
```

```
public class unsorted_to_sorted {
public static void main(String[] args) {
 int[] arr1= {7,6,8,3,6};
 int[] arr2= {1,2,4,9};
 int[] merge=new int[arr1.length+arr2.length];
 System.arraycopy(arr1, 0, merge, 0, arr1.length);
 System.arraycopy(arr2, 0, merge, arr1.length, arr2.length);
  //bubblesort
 int n = merge.length;
 for (int i = 0; i < n - 1; i++)
 for (int j = 0; j < n - i - 1; j++)
 if (merge[j] > merge[j + 1]) {
    int temp = merge[j];
    merge[j] = merge[j + 1];
    merge[j + 1] = temp;
   }
   System.out.println("Sorted Merged Array: " + Arrays.toString(merge));
 }
}
```

4)WAP on linear serach?
output:Element found at index: 2

```
package Array_prgm;

public class LinearSearch {
    public static void main(String[] args) {
        int[] arr = {10, 20, 30, 40, 50};
        int target = 30;
        boolean found = false;

        for (int i = 0; i < arr.length; i++) {
            if (arr[i] == target) {
                System.out.println("Element found at index: " + i);
                found = true;
                break;
            }
        }

        if (!found) {
            System.out.println("The element is not present");
        }
    }
}
```

5)WAP on bubble sort?
output: Sorted Array: [1, 4, 5, 6, 8, 9]

```
package Array_prgm;

import java.util.Arrays;

public class BubbleSort {
  public static void main(String[] args) {
   int[] arr= {1,4,6,8,5,9};
   int n = arr.length;
   for (int i = 0; i < n - 1; i++)
   for (int j = 0; j < n - i - 1; j++)
   if (arr[j] > arr[j + 1]) {
      int temp = arr[j];
      arr[j] = arr[j + 1];
      arr[j + 1] = temp;
     }
     System.out.println("Sorted Array: " + Arrays.toString(arr));
   }
 }
```

6)WAP to remove duplicate elements in array?
output:[1,2,3,4]

```
package Array_prgm;

import java.util.Set;
import java.util.TreeSet;

public class Remove_duplicates {
public static void main(String[] args) {
 int[] arr= {1,2,3,2,4};
 Set<Integer> s1=new TreeSet<Integer>();
 for (int i = 0; i < arr.length; i++) {
  s1.add(arr[i]);
 }
 System.out.println(s1);
 }
}
```

7)WAP to print duplicates in array?
output: 2,3

```
package Array_prgm;

public class duplicates {
    public static void main(String[] args) {
        int[] arr = {1, 2, 3, 2, 3,4};
        boolean isPrinted = false;
        for (int i = 0; i < arr.length; i++) {
            for (int j = i + 1; j < arr.length; j++) {
                if (arr[i] == arr[j] && !isPrinted) {
                    System.out.println(arr[i]);
                    isPrinted = true;
                    break;
                }
            }
            isPrinted = false;
        }
    }
}
```

8)WAP to print unique array elements in array?
output:1 3 4

```
package Array_prgm;

public class unicElement {
public static void main(String[] args) {
 int[] arr= {1,2,3,2,4};
 for (int i = 0; i < arr.length; i++) {
  boolean flag=true;
  for (int j = 0; j < arr.length; j++) {
   if(i!=j&&arr[i]==arr[j]) {
    flag=false;
    break;
   }
  }
  if(flag) {
   System.out.print(arr[i]+" ");
  }
 }
}
}
```

9)WAP to print the 1st non repeated element index in unsorted array?
output:The first non-repeated element is 6 at index 3

```
package Array_prgm;
import java.util.HashMap;

public class first_NonRepeated_Element {

public static void main(String[] args) {
 int[] arr = {4, 5, 4, 6, 7, 5, 8};
 HashMap<Integer, Integer> map = new HashMap<>();
  for (int num : arr) {
    map.put(num, map.getOrDefault(num, 0) + 1);
  }

  for (int i = 0; i < arr.length; i++) {
    if (map.get(arr[i]) == 1) {
      System.out.println("The first non-repeated element is " + arr[i] + " at index " + i);
      return;
      }
  }

 System.out.println("No non-repeated element found.");
 }
 }
```

10)WAP to find 1st min and max elements sum in unsorted array?

```
import java.util.Arrays;

public class MinMaxSum {
    public static void main(String[] args) {
        int[] arr = {3, 1, 4, 1, 5, 9, 2, 6};

        Arrays.sort(arr);
        int min = arr[0];
        int max = arr[arr.length - 1];

        int sum = min + max;

        System.out.println("1st Minimum: " + min);
        System.out.println("1st Maximum: " + max);
        System.out.println("Sum of 1st Min and Max: " + sum);
    }
}
```

11)WAP to find Nth min and max elements sum in unsorted array?

```
import java.util.Arrays;

public class NthMinMaxSum {
    public static void main(String[] args) {
        int[] arr = {3, 1, 4, 1, 5, 9, 2, 6};
        int N = 3;

        Arrays.sort(arr);

        int nthMin = arr[N - 1];
        int nthMax = arr[arr.length - N];
        int sum = nthMin + nthMax;

        System.out.println(N + "th Minimum: " + nthMin);
        System.out.println(N + "th Maximum: " + nthMax);
        System.out.println("Sum of " + N + "th Min and Max: " + sum);
    }
}
```

12)WAP to find the frequency of an array?

```
output: 1 1
 2 2
 3 1
 4 1
```

```
package sorting_array;

public class frq_arr {
 public static void main(String[] args) {
  int[] arr= {1,2,2,3,4};
  int[] arr1=new int[127];
  for (int i = 0; i < arr.length; i++) {
   int num=arr[i];
   arr1[num]++;
  }
  for (int i = 0; i < arr1.length; i++) {
   if(arr1[i] != 0) {
    System.out.println(i+" "+arr1[i]);
   }
  }
 }
}
```

13)WAP to find Max frequency in array?
output:2

```
package sorting_array;

public class most_frequent {
public static void main(String[] args) {
 int[] arr= {1,2,2,3,4};
 int[] arr1=new int[127];
 for (int i = 0; i < arr.length; i++) {
  int num=arr[i];
  arr1[num]++;
 }
 int maxfreq=0;
 int mostfrequentientnum=-1;
 for (int i = 0; i < arr1.length; i++) {
  if(arr1[i]>maxfreq) {
   maxfreq=arr1[i];
   mostfrequentientnum=i;
  }

 }
 System.out.println(mostfrequentientnum);
}
}
```

14)WAP to rotate the array towards right with Nth value?
input=>{1,2,3,4,5} if n=3 output 3,4,5,1,2

```
package Rotation_prgms;

public class right_rotation {
 public static void main(String[] args) {
  int[] arr= {1,2,3,4,5};
  int n=3;
  n=n%arr.length;
  for (int i = 0; i < arr.length; i++) {
   if(i<n) {
    System.out.print(arr[arr.length+i-n]+" ");
   }else {
    System.out.print(arr[i-n]+" ");
   }
  }
 }
}
```

15)WAP to rotate the array towards left with Nth value?
input=>{1,2,3,4,5} if n=2 output 3,4,5,1,2

```
package Rotation_prgms;

public class left_rotation {
 public static void main(String[] args) {
  int[] arr= {1,2,3,4,5};
  int n=2;
  n=n%arr.length;
  int j=0;

  for (int i = 0; i < arr.length; i++) {
   if(n<arr.length) {
    System.out.print(arr[n++]+" ");
   }else {
    System.out.print(arr[j++]+" ");
   }
  }
 }
}
```

16)WAP to test if an array contains a specific value.

```
package Array_prgm;

public class SpecificValue {
    public static void main(String[] args) {
        int[] arr = {3, 1, 4, 1, 5, 9, 2, 6};
        int target = 5;

        boolean found = false;

        for (int num : arr) {
            if (num == target) {
                found = true;
                break;
            }
        }

        if (found) {
            System.out.println("The array contains the value: " + target);
        } else {
            System.out.println("The array does not contain the value: " + target);
        }
    }
}
```

17)WAP to remove a specific element from an array.

```
package Array_prgm;

import java.util.Arrays;

public class Remove_Spefic_Element {
    public static void main(String[] args) {
        int[] arr = {3, 1, 4, 1, 5, 9, 2, 6};
        int target = 4;
        int count = 0;
        for (int num : arr) {
            if (num == target) {
                count++;
            }
        }
        if (count > 0) {
            int[] newArr = new int[arr.length - count];
            int index = 0;
            for (int num : arr) {
                if (num != target) {
                    newArr[index++] = num;
                }
            }
            System.out.println("Array after removing " + target + ": " + Arrays.toString(newArr));
        } else {
            System.out.println("Element " + target + " not found in the array.");
        }
    }
}
```

18)WAP to copy an array by iterating the array

```
package Array_prgm;

import java.util.Arrays;

public class ArrayCopy {
    public static void main(String[] args) {
        int[] arr = {3, 1, 4, 1, 5, 9, 2, 6};
        int[] copiedArr = new int[arr.length];

        System.arraycopy(arr, 0, copiedArr, 0, arr.length);

        System.out.println("Original Array: " + Arrays.toString(arr));
        System.out.println("Copied Array: " + Arrays.toString(copiedArr));
    }
}
```

19)WAP to find common elements between two arrays?
output [2,5]

```
package Array_prgm;

import java.util.ArrayList;

public class intersection_2arrays {
 public static void main(String[] args) {
  int[] arr1= {1,2,4,5};
  int[] arr2= {6,2,5,3};
  ArrayList<Integer> intersection = new ArrayList<Integer>();
  for (int i = 0; i < arr1.length; i++) {
   for (int j = 0; j < arr2.length; j++) {
     if (arr1[i] == arr2[j] && !intersection.contains(arr1[i])) {
                 intersection.add(arr1[i]);
   }
  }
 }
  System.out.println(intersection);
}
}
```

20)WAP to add two matrices of the same size.

```
package Array_prgm;

public class MatrixAddition {
    public static void main(String[] args) {
        int[][] matrix1 = {
            {1, 2, 3},
            {4, 5, 6},
            {7, 8, 9}
        };

        int[][] matrix2 = {
            {9, 8, 7},
            {6, 5, 4},
            {3, 2, 1}
        };

        //both matrices are of the same size
        int rows = matrix1.length; //r=3
        int cols = matrix1[0].length; //3 first row c=3

        int[][] result = new int[rows][cols];
              //3  3
        for (int i = 0; i < rows; i++) {
            for (int j = 0; j < cols; j++) {
                result[i][j] = matrix1[i][j] + matrix2[i][j];
            }
        }

        for (int i = 0; i < rows; i++) {
            for (int j = 0; j < cols; j++) {
                System.out.print(result[i][j] + " ");
            }
            System.out.println();
        }
    }
}
```

21)WAP to convert an array to an ArrayList

```
package Array_prgm;

import java.util.ArrayList;
import java.util.Arrays;

public class ArrayToArrayList {
    public static void main(String[] args) {
        Integer[] arr = {1, 2, 3, 4, 5};
        ArrayList<Integer> arrayList = new ArrayList<>(Arrays.asList(arr));
        System.out.println("ArrayList: " + arrayList);
    }
}
```

22)WAP to find all pairs of elements in an array whose sum is equal to a specified number

```
package Array_prgm;

public class PairSum {
    public static void main(String[] args) {
        int[] arr = {1, 4, 6, 8, 9, 2};
        int targetSum = 10;

        System.out.println("Pairs with sum " + targetSum + ":");

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

23)WAP to find a missing number in an array.?

```
package Array_prgm;

public class MissingNumber {
    public static void main(String[] args) {
        int[] arr = {1, 2, 4, 6, 3, 7, 8};
        int sum=0;
        for (int i = 0; i < arr.length; i++) {
   sum+=arr[i];
  }
        int n=arr.length+1;
        int missingnum=n*(n+1)/2;
        System.out.println(missingnum-sum);
    }
}
```

24)WAP to find a missing numbers sequence in an array.

```
package Array_prgm;

import java.util.ArrayList;
import java.util.Collections;

public class MissingNumbers {
    public static void main(String[] args) {
        int[] arr = {1, 2, 4, 6, 7};

        ArrayList<Integer> missingNumbers = new ArrayList<>();
        for (int n : arr) {
```

```
   missingNumbers.add(n);
  }
        //              1             7
        for (int i = arr[0]; i < arr[arr.length - 1]; i++) {
   if(!missingNumbers.contains(i)) {
    missingNumbers.add(i);
   }
  }
        Collections.sort(missingNumbers);
        for (int n : missingNumbers) {
   System.out.print(n+" ");
  }
    }
}
```

25)WAP to find common elements in three sorted (in non-decreasing order) arrays.

```
public class CommonElements {
    public static void main(String[] args) {
        int[] arr1 = {1, 5, 10, 20, 40, 80};
        int[] arr2 = {6, 7, 20, 80, 100};
        int[] arr3 = {3, 4, 15, 20, 30, 70, 80};

        int i = 0, j = 0, k = 0;

        // Loop through all arrays to find common elements
        while (i < arr1.length && j < arr2.length && k < arr3.length) {
            if (arr1[i] == arr2[j] && arr2[j] == arr3[k]) {
                System.out.print(arr1[i] + " ");
                i++;
                j++;
                k++;
            } else if (arr1[i] < arr2[j]) {
                i++;
            } else if (arr2[j] < arr3[k]) {
                j++;
            } else {
                k++;
            }
        }
    }
}
```

26)WAP to move all o’s to the end of an array. Maintain the relative order of the other (non-zero) array elements

```
public class MoveZeros {
    public static void main(String[] args) {
        int[] arr = {0, 1, 9, 0, 3, 12};

        int nonZeroIndex = 0;

        // Move non-zero elements to the beginning
        for (int i = 0; i < arr.length; i++) {
            if (arr[i] != 0) {
                arr[nonZeroIndex++] = arr[i];
            }
        }

        // Fill the remaining places with 0's
        while (nonZeroIndex < arr.length) {
            arr[nonZeroIndex++] = 0;
        }

        // Print the result
        System.out.println("Array after moving 0's to the end: ");
        for (int num : arr) {
            System.out.print(num + " ");
        }
    }
}
```

27)WAP to get the difference between the largest and smallest values in an array of integers. The array must have a length of at least 1.

```
public class MinMaxDifference {
    public static void main(String[] args) {
        int[] arr = {3, 1, 4, 1, 5, 9, 2, 6};

        // Initialize min and max with the first element
        int min = arr[0];
        int max = arr[0];

        // Loop through the array to find the min and max values
        for (int num : arr) {
            if (num < min) {
                min = num;
            }
            if (num > max) {
                max = num;
            }
        }

        // Calculate the difference
        int difference = max - min;
        System.out.println("The difference between largest and smallest values is: " + difference);
    }
}
```

28)WAP to compute the average value of an array of integers except the largest and smallest values

```
public class AverageExcludingMinMax {
    public static void main(String[] args) {
        int[] arr = {3, 1, 4, 1, 5, 9, 2, 6};

        int min = arr[0];
        int max = arr[0];
        int sum = 0;
        int count = arr.length;

        // Find min, max, and total sum
        for (int num : arr) {
            if (num < min) {
                min = num;
            }
            if (num > max) {
                max = num;
            }
            sum += num;
        }

        // Calculate the sum excluding min and max
        sum -= (min + max);
        count -= 2;

        // Calculate the average
        double average = (double) sum / count;
        System.out.println("The average excluding the largest and smallest values is: " + average);
    }
}
```

29)WAP to check if the sum of all the 10's in the array is exactly 30. Return false if the condition does not satisfy, otherwise true

```
public class SumOfTens {
    public static void main(String[] args) {
        int[] arr = {10, 5, 10, 10, 3, 10};

        int sum = 0;

        // Sum up the occurrences of 10
        for (int num : arr) {
            if (num == 10) {
                sum += num;
            }
        }

        // Check if the sum is exactly 30
        if (sum == 30) {
            System.out.println("True");
        } else {
            System.out.println("False");
        }
    }
}
```