

Name : Prerana Lavhate.

Email : preranaslavhate1630@gmail.com

Contact No : 9373752728

Assignment

Q.1 Merge two arrays by satisfying given constraints. Given two sorted arrays X[] and Y[] of size m and n each where $m \geq n$ and X[] has exactly n vacant cells, merge elements of Y[] in their correct position in array X[], i.e., merge (X, Y) by keeping the sorted order. For example, Input: X[] = { 0, 2, 0, 3, 0, 5, 6, 0, 0 }, Y[] = { 1, 8, 9, 10, 15 } The vacant cells in X[] is represented by 0 . Output: X[] = { 1, 2, 3, 5, 6, 8, 9, 10, 15 }

Code :

```
import java.util.Arrays;

public class MergeArrays {

    public static void mergeArrays(int[] X, int[] Y) {

        int m = X.length;

        int n = Y.length;

        int end = m - 1;

        for (int i = m - 1; i >= 0; i--) {

            if (X[i] != 0) {

                X[end] = X[i];

                end--;

            }

        }

        int i = end + 1;

        int j = 0;

        int k = 0;

        while (i < m && j < n) {

            if (X[i] < Y[j]) {

                X[k] = X[i];

                i++;

            } else {

                X[k] = Y[j];

                j++;

            }

            k++;

        }

    }

}
```

```

    }
    k++;
}
while (j < n) {
    X[k] = Y[j];
    j++;
    k++;
}
}

public static void main(String[] args) {
    int[] X = {0, 2, 0, 3, 0, 5, 6, 0, 0};
    int[] Y = {1, 8, 9, 10, 15};
    mergeArrays(X, Y);
    System.out.println(Arrays.toString(X));
}
}

```

Output :

```

PS D:\Java_Programs> javac MergeArrays.java
PS D:\Java_Programs> java MergeArrays
[1, 2, 3, 5, 6, 8, 9, 10, 15]

```

Q.2 Find maximum sum path involving elements of given arrays. Given two sorted arrays of integers, find a maximum sum path involving elements of both arrays whose sum is maximum. We can start from either array, but we can switch between arrays only through its common elements. For example, Input: X = {3, 6, 7, 8, 10, 12, 15, 18, 100} Y = {1, 2, 3, 5, 7, 9, 10, 11, 15, 16, 18, 25, 50} The maximum sum path is: 1 → 2 → 3 → 6 → 7 → 9 → 10 → 12 → 15 → 16 → 18 → 100. The maximum sum is 199

Code :

```

public class MaximumSumPath {
    public static int maxSumPath(int[] X, int[] Y) {
        int sum = 0;
        int sumX = 0, sumY = 0;
    }
}

```

```

int i = 0, j = 0;

int m = X.length, n = Y.length;

while (i < m && j < n) {
    if (X[i] < Y[j]) {
        sumX += X[i++];
    } else if (X[i] > Y[j]) {
        sumY += Y[j++];
    } else {
        sum += Math.max(sumX, sumY) + X[i];
        sumX = 0;
        sumY = 0;
        i++;
        j++;
    }
}

while (i < m) {
    sumX += X[i++];
}

while (j < n) {
    sumY += Y[j++];
}

sum += Math.max(sumX, sumY);

return sum;
}

public static void main(String[] args) {
    int[] X = {3, 6, 7, 8, 10, 12, 15, 18, 100};
    int[] Y = {1, 2, 3, 5, 7, 9, 10, 11, 15, 16, 18, 25, 50};

    int maxSum = maxSumPath(X, Y);

    System.out.println("The maximum sum is: " + maxSum);
}

```

```
}
```

Output :

```
PS D:\Java_Programs> javac MaximumSumPath.java
PS D:\Java_Programs> java MaximumSumPath
The maximum sum is: 199
```

Q3: Write a Java Program to count the number of words in a string using HashMap.

Code:

```
import java.util.HashMap;
import java.util.Scanner;

public class WordCounter {

    public static HashMap<String, Integer> countWords(String str) {
        HashMap<String, Integer> wordCountMap = new HashMap<>();
        String[] words = str.split("\\s+");
        for (String word : words) {
            word = word.replaceAll("[^a-zA-Z0-9]", "");
            word = word.toLowerCase();
            wordCountMap.put(word, wordCountMap.getOrDefault(word, 0) + 1);
        }
        return wordCountMap;
    }

    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        System.out.println("Enter a string:");
        String inputString = scanner.nextLine();
        scanner.close();

        HashMap<String, Integer> wordCounts = countWords(inputString);

        System.out.println("Word Counts:");
        for (String word : wordCounts.keySet()) {
```

```

        System.out.println(word + ": " + wordCounts.get(word));
    }
}
}

```

Output :

```

PS D:\Java_Programs> javac WordCounter.java
PS D:\Java_Programs> java WordCounter
Enter a string:
Java is Programming Language. Java is used in Software Development.
Word Counts:
development: 1
java: 2
software: 1
in: 1
is: 2
language: 1
used: 1
programming: 1

```

Q4: Write a Java Program to find the duplicate characters in a string.

Code :

```

import java.util.Scanner;

public class DuplicateCharacterFinder {

    public static void findDuplicateCharacters(String str) {

        boolean[] duplicates = new boolean[128];

        char[] charArray = str.toCharArray();

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

            int asciiValue = (int) charArray[i];

            if (duplicates[asciiValue]) {

                System.out.println(charArray[i]);

            } else {

                duplicates[asciiValue] = true;

            }

        }

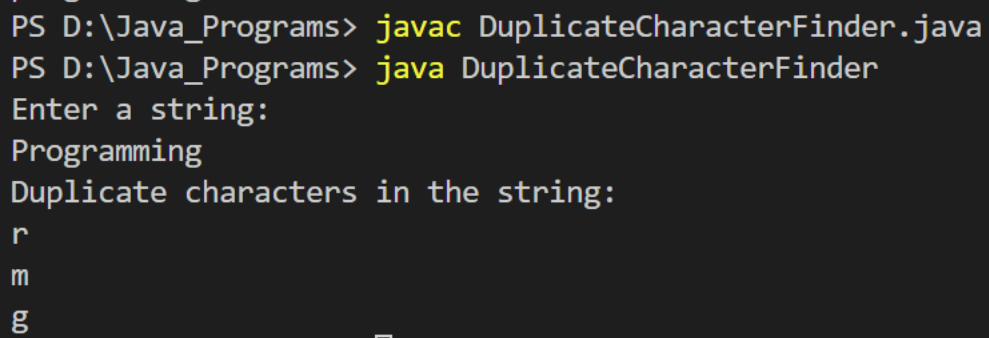
    }

}

```

```
public static void main(String[] args) {  
    Scanner scanner = new Scanner(System.in);  
    System.out.println("Enter a string:");  
    String inputString = scanner.nextLine();  
    scanner.close();  
    System.out.println("Duplicate characters in the string:");  
    findDuplicateCharacters(inputString);  
}  
}
```

Output :



```
PS D:\Java_Programs> javac DuplicateCharacterFinder.java  
PS D:\Java_Programs> java DuplicateCharacterFinder  
Enter a string:  
Programming  
Duplicate characters in the string:  
r  
m  
g
```