# day-016-string-assignment

## GitHub Link:

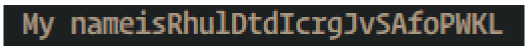
https://github.com/irahuldutta02/pw-skills-jdsd-assignments/tree/main/day-016-string-assignment/codes

## 1. WAP(Write a Program) to remove Duplicates from a String.(Take any String example with duplicates character)

```
Question_01.java
```

```java
// WAP(Write a Program) to remove Duplicates from a String.
import java.util.*;

class Question_01 {

  static void removeDuplicate(char str[], int length) {
    int index = 0;
    for (int i = 0; i < length; i++) {
      int j;
      for (j = 0; j < i; j++) {
        if (str[i] == str[j]) {
          break;
        }
      }
      if (j == i) {
        str[index++] = str[i];
      }
    }
    System.out.println(String.valueOf(Arrays.copyOf(str,
index)));
  }

  public static void main(String[] args) {
    String info = "My name is Rahul Dutta and I am currently
learing Java DSA and System Design from PW SKILL";
    char str[] = info.toCharArray();
    int len = str.length;
    removeDuplicate(str, len);
  }
}
```

```
Output :
```



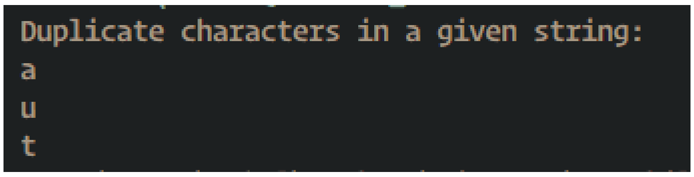
```
My nameisRhulDtdIcrgJvSAfoPWKL
```

## 2. WAP to print Duplicates characters from the String

```
Question_02.java
```

```java
// 2. WAP to print Duplicates characters from the String

public class Question_02 {

  public static void main(String[] args) {
    String string1 = "My name is Rahul Dutta";
    int count;
    char string[] = string1.toCharArray();

    System.out.println("Duplicate characters in a given
string: ");
    for (int i = 0; i < string.length; i++) {
      count = 1;
      for (int j = i + 1; j < string.length; j++) {
        if (string[i] == string[j] && string[i] != ' ') {
          count++;
          string[j] = '0';
        }
      }
      if (count > 1 && string[i] != '0')
System.out.println(string[i]);
    }
  }
}
```

```
Output :
```



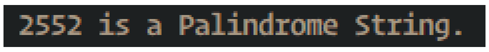
```
Duplicate characters in a given string:
a
u
t
```

**3. WAP to check if "2552" is palindrome or not.**

```
Question_03.java
```

```java
// 3. WAP to check if "2552" is palindrome or not.

public class Question_03 {
    public static void main(String[] args) {

        String str = "2552", reverseStr = "";

        int strLength = str.length();

        for (int i = (strLength - 1); i >=0; --i) {
          reverseStr = reverseStr + str.charAt(i);
        }

        if
(str.toLowerCase().equals(reverseStr.toLowerCase())) {
          System.out.println(str + " is a Palindrome
String.");
        }
        else {
          System.out.println(str + " is not a Palindrome
String.");
        }
      }
}
```

```
Output :
```



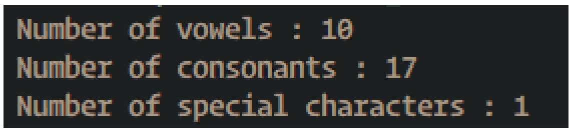
```
2552 is a Palindrome String.
```

## 4. WAP to count the number of consonants, vowels, special characters in a String.

```
Question_04.java
```

```
// WAP to count the number of consonants, vowels, special
characters in a String.

public class Question_04 {

  public static void main(String[] args) {
    int vCount = 0, cCount = 0, specialChar = 0;
    String str = "This is a really simple sentence &";
    str = str.toLowerCase();

    for (int i = 0; i < str.length(); i++) {
      if (
        str.charAt(i) == 'a' ||
        str.charAt(i) == 'e' ||
        str.charAt(i) == 'i' ||
        str.charAt(i) == 'o' ||
        str.charAt(i) == 'u'
      ) {
        vCount++;
      } else if (str.charAt(i) >= 'a' && str.charAt(i) <= 'z')
{
        cCount++;
      } else if (str.charAt(i) >= '0' && str.charAt(i) <= '9'
|| str.charAt(i) == ' ') {
        continue;
      } else {
        specialChar++;
      }
    }
    System.out.println("Number of vowels : " + vCount);
    System.out.println("Number of consonants : " + cCount);
    System.out.println("Number of special characters : " +
specialChar);
  }
}
```

```
Output :
```



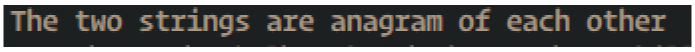
```
Number of vowels : 10
Number of consonants : 17
Number of special characters : 1
```

## 5. WAP to implement Anagram Checking least inbuilt methods being used.

```
Question_05.java
```

```
// WAP to implement Anagram Checking least inbuilt methods
being used

import java.util.Arrays;

class Question_05 {

  static char[] stringToArray(String str) {
    str = str.toLowerCase();
    char[] ch = new char[str.length()];
    for (int i = 0; i < str.length(); i++) {
      ch[i] = str.charAt(i);
    }
    return ch;
  }

  static boolean areAnagram(char[] str1, char[] str2) {
    int n1 = str1.length;
    int n2 = str2.length;
    if (n1 != n2) return false;
    Arrays.sort(str1);
    Arrays.sort(str2);
    for (int i = 0; i < n1; i++) {
      if (str1[i] != str2[i]) {
        return false;
      }
    }
    return true;
  }

  public static void main(String args[]) {
    String s1 = "silent";
    String s2 = "listen";
    char str1[] = stringToArray(s1);
    char str2[] = stringToArray(s2);
    if (areAnagram(str1, str2)) System.out.println(
      "The two strings are" + " anagram of each other"
    ); else System.out.println(
      "The two strings are not" + " anagram of each other"
    );
  }
}
```

```
Output :
```



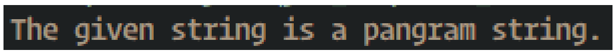
```
The two strings are anagram of each other
```

**6. WAP to implement Pangram Checking with least inbuilt methods being used.**

```
Question_06.java
```

```
// WAP to implement Pangram Checking with least inbuilt
methods being used
public class Question_06 {
  static int size = 26;
  static boolean isLetter(char ch) {
    if (!Character.isLetter(ch)) return false;
    return true;
  }
  static boolean containsAllLetters(String str, int len) {
    str = str.toLowerCase();
    boolean[] present = new boolean[size];
    for (int i = 0; i < len; i++) {
      if (isLetter(str.charAt(i))) {
        int letter = str.charAt(i) - 'a';
        present[letter] = true;
      }
    }
    for (int i = 0; i < size; i++) {
      if (!present[i]) return false;
    }
    return true;
  }
  public static void main(String args[]) {
    String str = "Abcdefghijklmnopqrstuvwxyz";
    int len = str.length();
    if (containsAllLetters(str, len)) System.out.println(
      "The given string is a pangram string."
    ); else System.out.println("The given string is not a
pangram string.");
  }
}
```

```
Output :
```



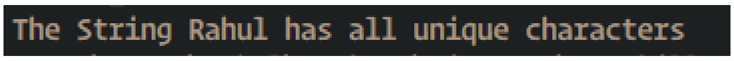
```
The given string is a pangram string.
```

**7. WAP to find if String contains all unique characters.**

```
Question_07.java
```

```java
// 7. WAP to find if String contains all unique characters.

class Question_07 {

  boolean uniqueCharacters(String str) {
    for (int i = 0; i < str.length(); i++) for (
      int j = i + 1;
      j < str.length();
      j++
    ) if (str.charAt(i) == str.charAt(j)) return false;
    return true;
  }

  public static void main(String args[]) {
    Question_07 obj = new Question_07();
    String input = "Rahul";

    if (obj.uniqueCharacters(input)) System.out.println(
      "The String " + input + " has all unique characters"
    ); else System.out.println(
      "The String " + input + " has duplicate characters"
    );
  }
}
```

```
Output :
```



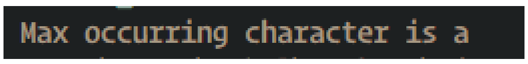
```
The String Rahul has all unique characters
```

**8. WAP to find the maximum occurring character in a String**

```
Question_08.java
```

```java
// 8. WAP to find the maximum occurring character in a String

public class Question_08 {

  static final int ASCII_SIZE = 256;
  static char getMaxOccurringChar(String str) {
    int count[] = new int[ASCII_SIZE];
    int len = str.length();
    for (int i = 0; i < len; i++) count[str.charAt(i)]++;

    int max = -1;
    char result = ' ';
    for (int i = 0; i < len; i++) {
      if (max < count[str.charAt(i)]) {
        max = count[str.charAt(i)];
        result = str.charAt(i);
      }
    }

    return result;
  }

  public static void main(String[] args) {
    String str = "Rahul Dutta";
    System.out.println(
      "Max occurring character is " + getMaxOccurringChar(str)
    );
  }
}
```

```
Output :
```



```
Max occurring character is a
```