**Java Program to find Repeated Characters of String**

The standard way to solve this problem is to get the character array from String, iterate through that and build a Map with character and their count. Then [iterate through that Map](http://java67.blogspot.sg/2013/08/best-way-to-iterate-over-each-entry-in.html) and print characters which have appeared more than once. So you actually need two loops to do the job, the first loop to build the map and second loop to print characters and counts.  
  
If you look at below example, there is only one static method called printDuplicateCharacters(), which does both this job. We first got the character array from String by calling toCharArray().  
  
Next we are using [HashMap](http://java67.blogspot.com/2013/02/10-examples-of-hashmap-in-java-programming-tutorial.html)to store characters and their count. We use containsKey() method to check if key, which is a character already exists or not, if already exists we get the old count from HashMap by calling get() method and store it back after incrementing it by 1.  
  
Once we build our Map with each character and count, next task is to [loop through Map](http://java67.blogspot.sg/2013/08/best-way-to-iterate-over-each-entry-in.html) and check each entry, if count, which is the value of Entry is greater than 1, then that character has occurred more than once. You can now print duplicate characters or do whatever you want with them.

**import** **java.util.HashMap**;

**import** **java.util.Map**;

**import** **java.util.Scanner**;

**import** **java.util.Set**;

/\*\*

\* Java Program to find duplicate characters in String.

\*

\*

\* @author http://java67.blogspot.com

\*/

**public** **class** **FindDuplicateCharacters**{

**public** **static** **void** **main**(String args[]) {

printDuplicateCharacters("Programming");

printDuplicateCharacters("Combination");

printDuplicateCharacters("Java");

}

/\*

\* Find all duplicate characters in a String and print each of them.

\*/

**public** **static** **void** **printDuplicateCharacters**(String word) {

**char**[] characters = word.toCharArray();

// build HashMap with character and number of times they appear in String

Map<Character, Integer> charMap = **new** HashMap<Character, Integer>();

**for** (Character ch : characters) {

**if** (charMap.containsKey(ch)) {

charMap.put(ch, charMap.get(ch) + **1**);

} **else** {

charMap.put(ch, **1**);

}

}

// Iterate through HashMap to print all duplicate characters of String

Set<Map.Entry<Character, Integer>> entrySet = charMap.entrySet();

System.out.printf("List of duplicate characters in String '%s' %n", word);

**for** (Map.Entry<Character, Integer> entry : entrySet) {

**if** (entry.getValue() > **1**) {

System.out.printf("%s : %d %n", entry.getKey(), entry.getValue());

}

}

}

}

Output

List of duplicate characters in String 'Programming'

g : **2**

r : **2**

m : **2**

List of duplicate characters in String 'Combination'

n : **2**

o : **2**

i : **2**

List of duplicate characters in String 'Java'