Find the first non-repeating character in the word or String.

First Non-Repeating character in String.

Let us consider the word

Example: java

Output j

Example mississippi

Output m

So the first character which is not repeated again in the String needs to be printed out.

So there are couple of ways to do it.

We will discuss few ways to do so with time and space trade offs.

1) We can use LinkedHashMap<> and insert K as character and V as count of characters. Good thing about LinkHashMap<> is that it mainstains the order of insertion. Now we can iterate through the Map and get the K whose V is 1.

2) Use of HashMap<>. HashMap<> does not insert the data into Map in ordered way. But nevertheless we can use it for this problem. Iterate through the String and insert K as character and V as count of characters. Now Again iterate thought the String and check for every character is corresponding V is 1. If yes, print it out and break from loop.

3) Use an int[] of 256. increment the index of character by 1. Now Iterate through the string and then return the first index whose value is 1.

All of the above solutions has time complexity of O(n) and space complexity of O(n).

Below is the program for the 3 methods described above:

**package** strings;

**import** java.util.HashMap;

**import** java.util.LinkedHashMap;

**import** java.util.Map;

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

**import** java.util.Set;

**public** **class** FirstNonrepeatingCharacter {

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

*firstNonRepeatingCharacterByArray*(**null**);

*firstNonRepeatingCharacterByArray*("a");

*firstNonRepeatingCharacterByArray*("aa");

*firstNonRepeatingCharacterByArray*("abcdef");

*firstNonRepeatingCharacterByArray*("aab");

}

/\*\*

\* In this method we use LinkedHashMap which maintains the insertion order.

\* \*/

**public** **static** **void** firstNonRepeatingByLinkHashMap(String str) {

/\*\*

\* If string is null or ots length is 1 then no need to procees it.

\* Just print it.

\* \*/

**if** (str == **null** || str.length() == 1) {

System.***out***.println(str);

**return**;

}

//define new LinkedHashMap<>

Map<Character, Integer> map = **new** LinkedHashMap<Character, Integer>();

**char**[] ch = str.toCharArray();

/\*\*

\* process the char[] ch

\* place each character in map.

\* if the character is repeated then get old value and add 1 to it.

\* else the character is new then insert 1 as value

\* \*/

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

**if** (map.containsKey(ch[i])) {

map.put(ch[i], map.get(ch[i]) + 1);

} **else** {

map.put(ch[i], 1);

}

}

Set<Entry<Character, Integer>> entries = map.entrySet();

**for** (Entry<Character, Integer> entry : entries) {

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

System.***out***.println("First non repeating character is "+entry.getKey());

**return**;

}

}

System.***out***.println(str+" "+"All characters are unique");

}

/\*\*

\* In this method we use HashMap.

\* Remember HashMap does not maintain insertion ordered as it is unordered collection.

\*

\* \*/

**public** **static** **void** firstNonRepeatingByHashMap(String str) {

/\*\*

\* If string is null or ots length is 1 then no need to procees it.

\* Just print it.

\* \*/

**if** (str == **null** || str.length() == 1) {

System.***out***.println(str);

**return**;

}

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

**char**[] ch = str.toCharArray();

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

**if** (map.containsKey(ch[i])) {

map.put(ch[i], map.get(ch[i]) + 1);

} **else** {

map.put(ch[i], 1);

}

}

Set<Entry<Character, Integer>> entries = map.entrySet();

**for** (**char** chr : str.toCharArray()) {

**if** (map.get(chr) == 1) {

System.***out***.println("First non repeating character is "+chr);

**return**;

}

}

System.***out***.println(str+" "+"All characters are unique");

}

/\*\*

\* This method uses an array of 256 to mark entry o character.

\* \*/

**public** **static** **void** firstNonRepeatingCharacterByArray(String str) {

**if** (str == **null** || str.length() == 1) {

System.***out***.println(str);

**return**;

}

**int**[] chars = **new** **int**[256];

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

chars[str.charAt(i)]++;

}

**for** (**char** ch : str.toCharArray()) {

**if** (chars[ch] == 1) {

System.***out***.println("First non repeating character is "+ch);

**return**;

}

}

System.***out***.println(str+" "+"No Non-Repeating Character found.");

}

}

Output:

null

a

aa No Non-Repeating Character found.

First non repeating character is a

First non repeating character is b