**Studio: Counting Characters**

Let's revisit the [Counting Characters studio](https://runestone.launchcode.org/runestone/static/thinkcspy/Studios/counting-characters.html) from unit 1.

Write a program that calculates the number of times each character occurs in a string and prints these counts to the console. You could get the string as input from the user, but for the sake of simplicity, you may also hard-code the string into your program as the value of a variable. Here’s a test string, for your convenience:

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Nunc accumsan sem ut ligula scelerisque sollicitudin. Ut at sagittis augue. Praesent quis rhoncus justo. Aliquam erat volutpat. Donec sit amet suscipit metus, non lobortis massa. Vestibulum augue ex, dapibus ac suscipit vel, volutpat eget massa. Donec nec velit non ligula efficitur luctus.

Your approach to this problem should be the same as with Python: Loop through the string one character at a time, storing and/or updating the count for a given character using an appropriate data structure.

You can turn a String object into an array of characters using:

char[] charactersInString = myString.toCharArray();

Some questions to ponder as you start this studio:

* Which type of data structure (ArrayList, HashMap, or Array) should you use to store character counts?
* You'll need to "initialize" the counts for the characters in some fashion. It's probably better to do this as you go through the string, as opposed to doing so before you loop through the string. (Why?)

If you're having trouble recalling how to create a new class, revisit the instructions in [Studio: Area of a Circle](https://education.launchcode.org/skills-back-end-java/studios/area/). And don't forget to check out the [Bonus Missions](https://education.launchcode.org/skills-back-end-java/studios/counting-characters/#bonus-missions) below.

For the example string above, your output should look something like:

A: 1

D: 2

L: 1

N: 1

P: 1

U: 1

V: 1

: 50

a: 22

b: 3

c: 17

d: 4

e: 26

f: 2

g: 7

h: 1

i: 27

j: 1

l: 17

,: 4

m: 11

n: 14

.: 8

o: 15

p: 7

q: 3

r: 9

s: 29

t: 29

u: 28

v: 4

x: 1

**Bonus Missions**

Here are few ways you might modify your solution:

* Make the counts case-insensitive
* Exclude non-alphabetic characters
* Get the string as input from the user at the command line
* Read the string in from a file. *Note:* This is a hard one, and we won't talk about reading from files in Java in this course, so be ready for a tough challenge if you bite this one off.