EQUITYZEN

Thanks for taking EquityZen's Front-End Programming Challenge

Please read this page carefully before proceeding.

1) This is a timed challenge. You have 2 hours to complete your solutions once you begin. Please do not read the actual test (next page onwards) until you have set aside 2 hours. You will time yourself and we will accept your solution on the honor system. Code can always be improved, so for your own sanity, please stop at the 2-hour mark and begin preparation to send in your solutions.
2) We expect you will check your code, test it, and time-permitting add more thorough comments (including scenarios tested). If you do not finish in the allotted time, please submit whatever you have completed.
3) While correctness is important, factors that are nearly almost as important include code clarity, design, efficiency, testing, and comments. A working answer to each required question is more important than an exceptional answer to any one question.
4) If you have any questions with the problem set, please use your best judgment, document your assumptions, and continue.
5) When you have finished, please send your response to engineering@equityzen.com as a .zip, .tar, or .tar.gz attachment. Questions done in JavaScript should also be hosted on a webpage with a link sent along with your solutions email (time spent doing this doesn't count towards the 2 hours).
6) You are welcome to include a link to a privately hosted repository (e.g. BitBucket, GitHub) along with your email; but please ensure you do not publish your code to a repository that it is publicly accessible.

Questions on next page

Question 1

Your solution should be in JavaScript.

The sentence "A quick brown fox jumps over the lazy dog" contains every single letter in the alphabet. Such sentences are called pangrams. You are to write a method `get_letters`, which takes a string, `sentence`, and returns a string with all the unique letters. You should ignore the case of the letters in sentence, and your return value should be all lowercase letters, in alphabetical order.
Your submission should be a single file which must contain the method `get_letters(sentence).`

Examples: (Note: the double quotes should not be considered part of the input or output strings.)

```
1) "A quick brown fox jumps over the lazy dog"
Returns: "abcdefghijklmnopqrstuvwxyz"

2) "A slow yellow fox crawls under the proactive dog"
Returns: "acdefghilnoprstuvwxy"

3) "Lions, and tigers, and bears, oh my!"
Returns: "abdeghilmnorsty"
```

Question 2
Here at EquityZen we leverage APIs in order to connect our app to third party services that provide pre-built functionality. While we do not currently connect to GitHub, they provide a good API for searching repositories and users (they rate-limit so don't abuse the API).

You can use any JavaScript library of your choosing for this question. Please include a link to your solution.

Create a webpage with a search bar where I can enter some text. When I "submit" the search, take this text and perform a "Users Search" using the GitHib's API's search function (`https://developer.github.com/v3/users/`).

For example, if I had typed "equity" in the search bar, you would perform the following call:
`https://api.github.com/search/users?q=equity`

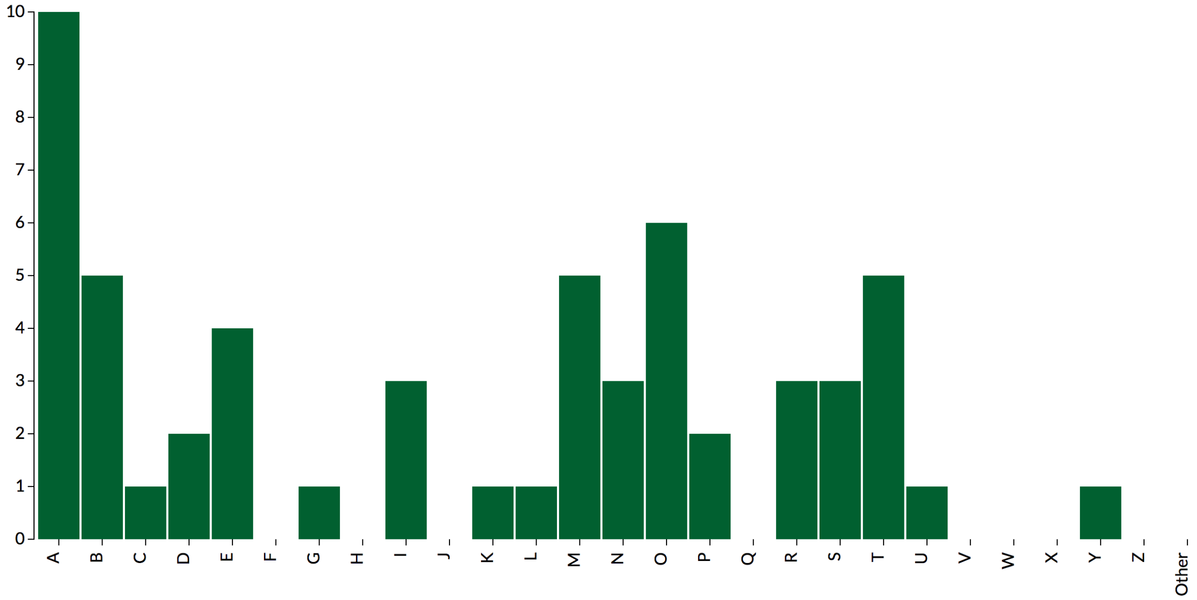
and, GitHub would return many results that look like:

```
{
  "total_count": 174,
  "incomplete_results": false,
  "items": [
    {
      "login": "equity",
      "id": 770164,
      …,
      "score": 15.0129385
    },
    {
      "login": "equityx-ltd",
      …,
      "score": 12.984634
    },
    {
      "login": "equitysplit",
      "id": 25038989,
      …,
      "score": 12.563136
    },
………
}
```

Please display a bar chart (sample on next page):
X-axis has 27 values (26 letters of the alphabet and "Other")
Y-axis has count each letter appears in the set of login attributes
(in `equitysplit` , "a" appeared 0 times, "i" 2 times, "q" 1 times)
Ignore upper and lower case
Only 5 items from the API results should be counted at a time, with a button to change the graph to show the next 5 items. E.g. items 1-5, then items 6-10

Bonus: add tool-tips so when you mouse over a given bar, it displays the count value for that letter.

The result would look something like below:



Question 3

Your solution should be in JavaScript

A collection of particles is contained in a linear chamber. They all have the same speed, but some are headed toward the right and others are headed toward the left. These particles can pass through each other without disturbing the motion of the particles, so all the particles will leave the chamber relatively quickly.

You will be given the initial conditions by a string, `init`, containing at each position an 'L' for a leftward moving particle, an 'R' for a rightward moving particle, or a '.' for an empty location. `init` shows all the positions in the chamber. Initially, no location in the chamber contains two particles passing through each other.

We would like an animation of the process. At each unit of time, we want a string showing occupied locations with an 'X' and unoccupied locations with a '.'. Create a class `Animation` that is initialized by a string, `init`, and contains an instance method `animate` that takes an int, `speed`, the number of positions each particle should move.

The constructor should display the original position of the chamber. The animate function should display the position of the particles after moving. If doing in JavaScript, use UI elements to portray the animation on a webpage (e.g. a text input and button to initialize the chamber, and a number input and button to animate the chamber)

Examples: (Note: the double quotes should not be considered part of the input or output strings.)

```
1) Python Example:
init = "..R...."
animation = Animation(init)
Prints "..X....",
animation.animate(2)
Prints "....X.."
animation.animate(1)
Prints ".....X."
animation.animate(2)
Prints "......."

The single particle starts at the 3rd position, moves to the 5th, then 6th,
and then out of the chamber.
```

```
2) JavaScript example:
var init = "RR..LRL";
var animation = new Animation(init);
Displays "XX..XXX"
animation.animate(1);
Displays ".XXX.XX"
animation.animate(1);
Displays "..XXX.."
```

After the second animate, there are actually 4 particles in the chamber, but two are passing through each other at the 3rd position.

Question 4 (BONUS)

Do question 3 in Python.