Challenge 1: Algorithm Test

Problem Overview: String Compression

No libraries for challenge 1.

Implement a string compression using python. For example, aaaabbbccddddddee would become a4b3c2d6e2. If the length of the string is not reduced, return the original string.

Requirements

Implement a compress function which accepts an input string and returns a compressed string.
Code must be implemented using python 3.6 and must follow strictly pep8 rules.
Provide comments regarding the implementation.

Test Cases

```
assert compress(‘bbcceeee’) == ‘b2c2e4’
assert compress(‘aaabbbcccaaa’) == ‘a3b3c3a3’
assert compress(‘a’) = a
```

Explanation

Explain time complexity of the compression written.

Problem Overview: Network Failure Point

We have a mesh network connected by routers labeled from 1 to 6 in a directed manner. Write an algorithm that can detect the routers with the highest number of connections so we might know which routers will cause a network failure if removed. Multiple connections between the same routers should be treated as separate connections. A tie for the most number of connections is possible.

Requirements

Implement a identify_router function that accepts an input graph of nodes representing the total network and identifies the node with the most number of connections.
Return the label of the node.

Implement a directed **graph** data structure using Python 3.6 and up.

Each node is unique thus there will be no cases of having multiple nodes with the same label.
Each node will have an infinite number of both inbound and outbound links.

Test Cases

1 -> 2 -> 3 -> 5 -> 2 -> 1 = 2 ***since router 2 has 2 inbound links and 2 outbound links**

1 -> 3 -> 5 -> 6 -> 4 -> 5 -> 2 -> 6 = 5 *** since router 5 has 2 inbound links and 2 outbound link**

2 -> 4 -> 6 -> 2 -> 5 -> 6 = 2, 6 *** since router 2 has 1 inbound link and 2 outbound links and 6 has 2 inbound links and 1 outbound link**

Explanation

Explain time complexity of the identify_router function written.

Challenge 2: Django Test

Problem Overview

Libraries allowed for challenge 2.

Write a simple application that allows a user to manage subscriptions using Stripe and Python Django 4.

Problem Details

1. Upon user signup, we want the user to sign up for a free trial of a Aben Premium membership which includes 7 days of premium service. Afterwards the user will be charged $49.99 per month.
2. When a user decides to sign up, the Subscription must be activated and we must track trial expiration. After 7 days, the initial charge of $49.99 must be charged on the test credit card.
3. On the Stripe interface, this must appear as a trial subscription.
4. The user can cancel their subscription at any time. Upon cancellation, the subscription is marked as cancelled and would also appear on the Stripe interface as a cancelled subscription.

Technical Requirements

- Use django registration (https://django-registration.readthedocs.io/en/3.1/) for the user signup.
- Databases can be MySQL or PostgreSQL.
- Design a Subscription Model with state fields that represent different stages of the subscription process. (You can name the different stages in your own naming convention).
- Code must be pep8 compliant.
- Code must have comments explaining functionality.
- You can use standard bootstrap code for user interface.

Code Submission for both Challenges

Please sign up to Bitbucket and create a code repository. Please put both challenges into individual folders named challenge1 and challenge2 respectively. Share the code with jordan@whatsbusy.com