CIS-2235 Advanced Sys Admin
Lab #7 Assignment

Objective

Practice developing basic scripts in Python

Introduction

In this lab, you'll write three python scripts. Your submission for the lab should include the scripts as 3 separate script files with .py extensions plus a writeup/lab report. It is easier for me if you submit your files as 4 separate files vs. a compressed archive (tarball, zip, etc).

Your writeup should demonstrate your three programs in action -- specifics are given in each section below. You do not need to include any details about how you wrote the scripts (IDE, docs, etc.), or the code in your actual scripts – I will review your code files directly and offer tips on how they could be improved. If your code has any major functional flaws it may cost you points, but basic formatting/naming/etc. choices will not.

Tasks

1. Write a math game which asks the user 10 addition questions by randomly selecting two integers each time. Compare the users input to the correct answer and tell them if they are right or wrong. Collect the scores during the game and at the end display the total number of correct answers and the percentage. Do not throw an error on bogus (non-numeric) answers, just count them as incorrect and ask the next question. Your solution should use at least one function. In your writeup, be sure to show some good answers and some incorrect answers. Your incorrect answers should include a non-numeric answer to demonstrate you've handled exceptions correctly.

2. Write a python script to look for discrepancies between /etc/passwd and dirs in /home (they should match). Report mismatches by generating a message pointing out a user who doesn't have a /home/<userdir> *or* a /home/<userdir> without a valid user entry in passwd. Notice there are *two* tests here. For Ubuntu, users start at uid=1000, so there is no need to print messages for system id's with uids < 1000. For your testing and writeup, you should add a user who doesn't have a home directory to your passwd file, and also create a directory in /home that does not match a valid user. Also make sure you don't give a false alert on "lost+found", since that is an expected directory in /home.

   Hint: look at str.split() which will split strings up based on a delimiter. You can use this to split the lines in the password file. split() returns a list of elements corresponding to the various fields.

3. CGI is the common gateway interface which allows you to invoke scripts from within your

webserver. You should look for reasonable resources to help you figure out how to make CGI work within Apache/Nginx. Is CGI already enabled? How would you enable it if not? (if you find mention of CGI versus CGID, either is fine).

In order to demonstrate your Python CGI setup works, write a script that lists all the valid users on the system with a number in front of them. You may use either /etc/passwd or /home to extract user names.

For example:

1. red
2. frank
3. lisa

A couple of notes about writing CGI scripts:

- The actual text emitted needs to be valid HTML which your webserver will return to the requestor's web browser -- so you'll need to print appropriate HTML tags.
- The first line you emit needs to be: "Content-type: text/html\n\n" which sets the content type header, followed by a blank line. Then proceed with whatever data and HTML tags are appropriate.

Your writeup for this task should show:

- the resources you used to figure out how to make CGI work
- the steps you took to make CGI work
- the URL you used to invoke the script
- the result of your script displayed in a browser