CS 101 – Lab #8 – Password Security

You probably know that passwords such as “dog” and “1234” are too easy for a hacker to guess. In today’s lab, we will practice with examining candidate passwords to see if they are relatively secure or not. This lab will also review some Python concepts we worked on earlier in the course.

You will write a short Python program that will ask the user to enter a password, and then evaluate this password to see how secure it is. We will use the following criteria for judging how good a candidate password is. It must have 4 properties:

* At least one capital letter
* At least one lowercase letter
* At least one digit
* At least one punctuation symbol.

Here is a pseudocode outline of how your program should work:

1. Ask the user to enter a candidate password, and place their response in a variable called input. Remember to use the raw\_input function since you are getting text (not numerical) input from the user.
2. Create 4 variables: hasCapital, hasLowercase, hasDigit, and hasSymbol, and set each of these variables equal to False.
3. Write a loop that looks at each character in the input.
	1. If this letter’s ASCII code is between ‘A’ and ‘Z’ inclusive, then we want to set hasCapital to True.
	2. If this letter’s ASCII code is between ‘a’ and ‘z’ inclusive, then set hasLowercase to True.
	3. If the ASCII code is between ‘0’ and ‘9’ inclusive, then set hasDigit to True.
	4. If the character is a punctuation symbol, then we want hasSymbol to become True. A nifty way to write this if-condition is to say:

if letter in “\_\_\_\_\_\_\_\_\_\_“:

and all you need to do is put all of the keyboard’s symbols inside the double quotes. There are just 2 special cases to watch out for. Inside the string, you need to type \\ to refer to the backslash, and \” to refer to a double quote symbol.

1. After the loop, we need to check our four true/false variables. If they are all equal to true, then print a message telling the user that the password is good. Otherwise, tell the user that the password is not secure enough.

After you have typed in the above program, run it and try a few sample passwords. It should accept only those passwords that have all 4 necessary types of characters.

Here is a minor enhancement to the program. Rather than giving just a yes/no output for the user, we can establish a point system. A password gets 1 point for answering each of the 4 questions correctly. So, a candidate password is rated on a 0-4 scale. At the end of your program, add code to accomplish the following:

1. Create a variable called points and set it equal to 0.
2. If the hasCapital variable equals True, add 1 to the number of points. Ditto for the other 3 boolean variables.
3. Finally, output a sentence telling the user how many points their password is worth.

Run your program to see that your point system works correctly. For example, “abC#” should be worth 3 points.

That’s it! I hope you have learned a lot in CS 101. It has been such a pleasure for me to accompany you along your journey through computer problem solving this term. I hope everyone does well on the final exam. Best of luck on the final and in all your other classes!