Programming 1 - Assessment 2:

1. Condition to enter a password or have a random password generated:-

while password == "":

print('Please enter a password')

password = input('Create a Password: ')

else:

(Random password will be generated by the program)

* This will ensure that the user must enter a password, or the program will not continue.
* The random library must be imported, and an input should be specified for a random password to be generated e.g. enter random for random password.
  + This gives the user the freedom to choose what is most suitable for them.

1. Create Lists for all possible random characters:-

e.g. lower\_case = ‘abcdefghijklmnopqrstuvwxyz’

upper\_case =’ABCDEFGHIJKLMNOPQRSTUVWXYZ’

special\_c = ‘!@#$%^&\*()\_+<>?~|\/’

I researched diverse ways on how to provide all characters and found a more optimal way than inputting them manually. According to (Python Software Foundation, n.d.), importing the string library allows me to access all the possible characters and assigning them to variables.

**must import string library.**

lower\_case = string.ascii\_lowercase

upper\_case = string.ascii\_uppercase

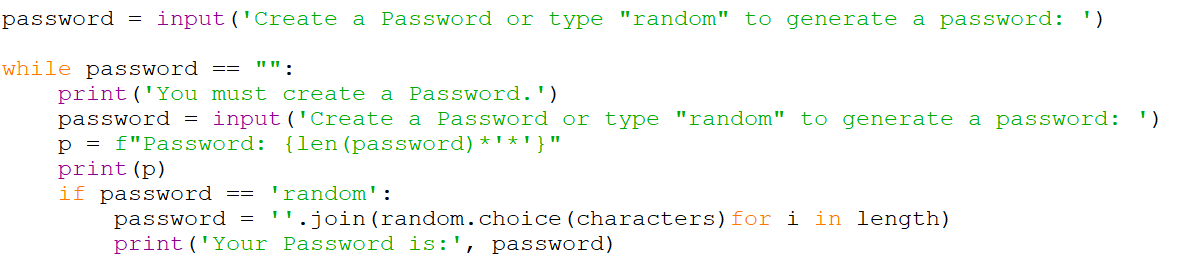
special\_c = string. Punctuation

numbers = string.digits

1. Make it a log-in / account or username to verify identity at the end with a limited number of attempts 3-5.
2. Security Questions
3. 2-Factor Authentication – Wouldn’t work because cannot/unaware on how to send text or code anywhere (**TimeStamp3: 03/11/2024)**

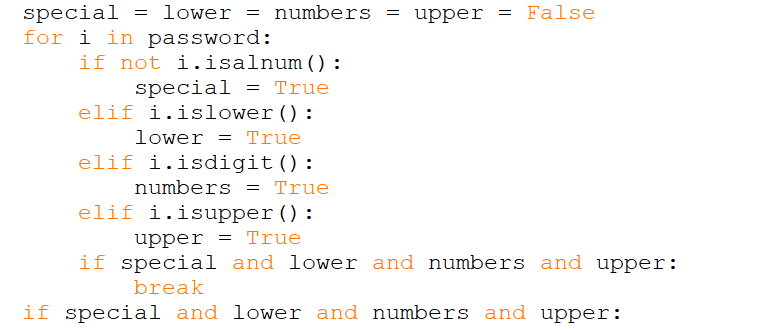
**TimeStamp1: 21/10/2024**

Issues:

* When generating a random password the word “random” was always used within the password
  + Solution: changed password = ‘random’.join(random.choice(characters)for i in length) to password = ''.join(random.choice(characters)for i in length). Also, by inputting the random library, it allows a random shuffled password to be created, along the criteria I provide, i.e. length, characters, special characters, etc…
* Initially had a really long code with many nested ifs that had repetitive conditions below each separate condition but found that it was not optimised.
  + Solution: Had to restructure the coding and conditions to make it more optimised by finding the common condition between other conditions to make it the ‘primary condition’ and I found it more logical to make the while function ‘while True’ which in sense allows me to create multiple conditions below that the program will run through separately until the truth value is confirmed. Where then the while loop will not run anymore.
* Initially the code did not work
  + Solution: had to check if any character in the string was a special character so to do this I had to break it down and check for each criterion separately. Also had to set truth value to false . So that when the criterion is true, it proceeds with the following steps, If false, then it prints a statement asking to follow the criteria.

**TimeStamp2: 21/10/2024**

* Could not optimise a for loop that was 12-13 lines due to the expression being Boolean and had to check each criterion separately



* + Solution: used any() function to optimise from 12-13 lines to 5-4 lines but still due to the Boolean property, each criterion had to be separate i.e. upper case, lower case, symbols and numbers

A close up of a text

Description automatically generated with medium confidence

**TimeStamp3: 05/11/2024**

Research:

<https://www.javatpoint.com/check-if-string-has-character-in-python>

https://docs.python.org/3.4/library/string.html#helper-functions