Random Password Generator and Password Strength Checker

Antonio Shields

School of Cybersecurity, Old Dominion University

CYSE 250: Basic Cybersecurity Programming and Networking

Prof. Shobha Vatsa

December 8, 2022

Random Password Generator and Password Strength Checker

Problem statement

The purpose of this project is to create a random password generator using Python that is able to make stronger passwords faster and easier than creating a password manually and to test the password strength using a password strength checker also created in Python.

Hardware and Software Details

The hardware that was used to conduct this group project was a Dell Inspiron 15 5000 laptop with 11th Gen Intel(R) Core(TM) i5-1135G7 @ 2.40GHz 1.38 GHz processor and 64-bit operating system. The operating system used was Windows 11 pro version 22H2. The software that was used to conduct the programming for both the Random Password Generator and the Password Strength Checker was PyCharm 2022.2.2 (Community Edition) Build #PC-222.4167.33

Results and Discussion

Password Generator Program (Python):

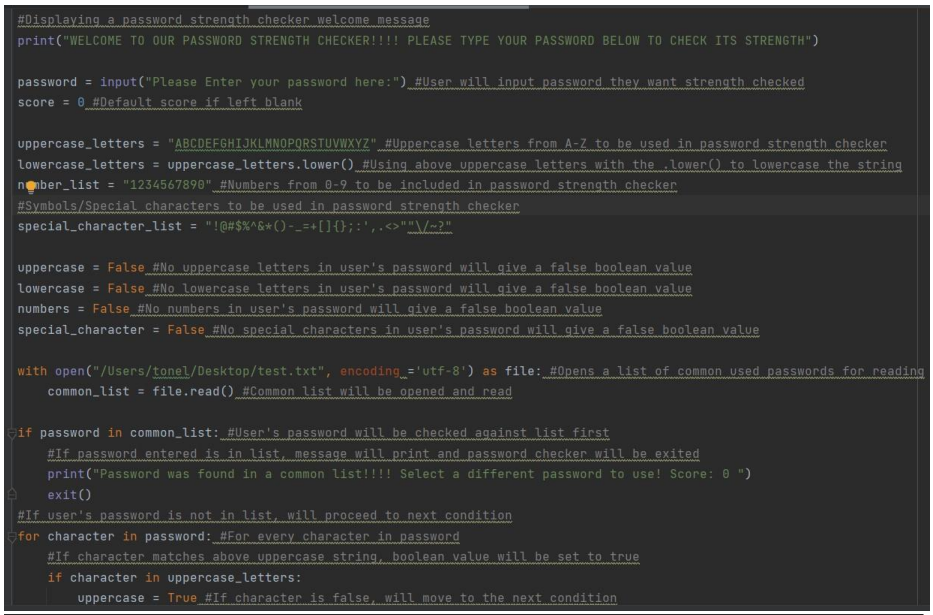
```python
#Import the random module
import random

#Displaying a password generator welcome message
print("WELCOME TO OUR PASSWORD GENERATOR!!!! PLEASE ANSWER THE QUESTIONS BELOW TO RECEIVE YOUR PASSWORD(S).")

uppercase_letters = "ABCDEFGHIJKLMNOPQRSTUVWXYZ" #Uppercase letters from A-Z to be used in generator
lowercase_letters = uppercase_letters.lower() #Using above uppercase letters with the .lower() to lowercase the string
numbers = "1234567890" #Numbers from 0-9 to be included in password generator
symbols = "!@#$%^&+()-_=+[]{};:',.<>""\/~?" #Symbols/Special characters to be used in password generator

#All above characters that will be used for password generator simplified
all_characters = (uppercase_letters + lowercase_letters + numbers + symbols)

#While loop that will generate passwords based on user's input
while True:
    #User will input how many passwords they want generated
    password_count = int(input("How many passwords would you like generated? : "))
    if password_count == 0: #If user inputs "0" as the amount of passwords needed, they will break out of loop
        break #breaking out of loop if user input for amount of passwords needed is "0"
    else: #Any input other than "0" will continue through the loop
        #User will input how long they want the password to be
        password_length = int(input("How long would you like your password to be? (Number of Characters) : "))
        #Uses input of how many passwords are needed to determine how many times the loop will run
        for number in range(password_count):
            password = "" #password that will be created
            #Uses input of the password length to determine how many times each loop will run to generate full password
            for number in range(password_length):
                #randomly choices a character from the all_characters list
                password_characters = random.choice(all_characters)
                password = password + password_characters #generates password from loop results
            print ("Here is your generated password : ", password) #prints generated passwords

#Message will be printed when while loop is broken
print("Thank you for using our Password Generator! Have a nice day!")
```

Password Generator Run Results (Python):

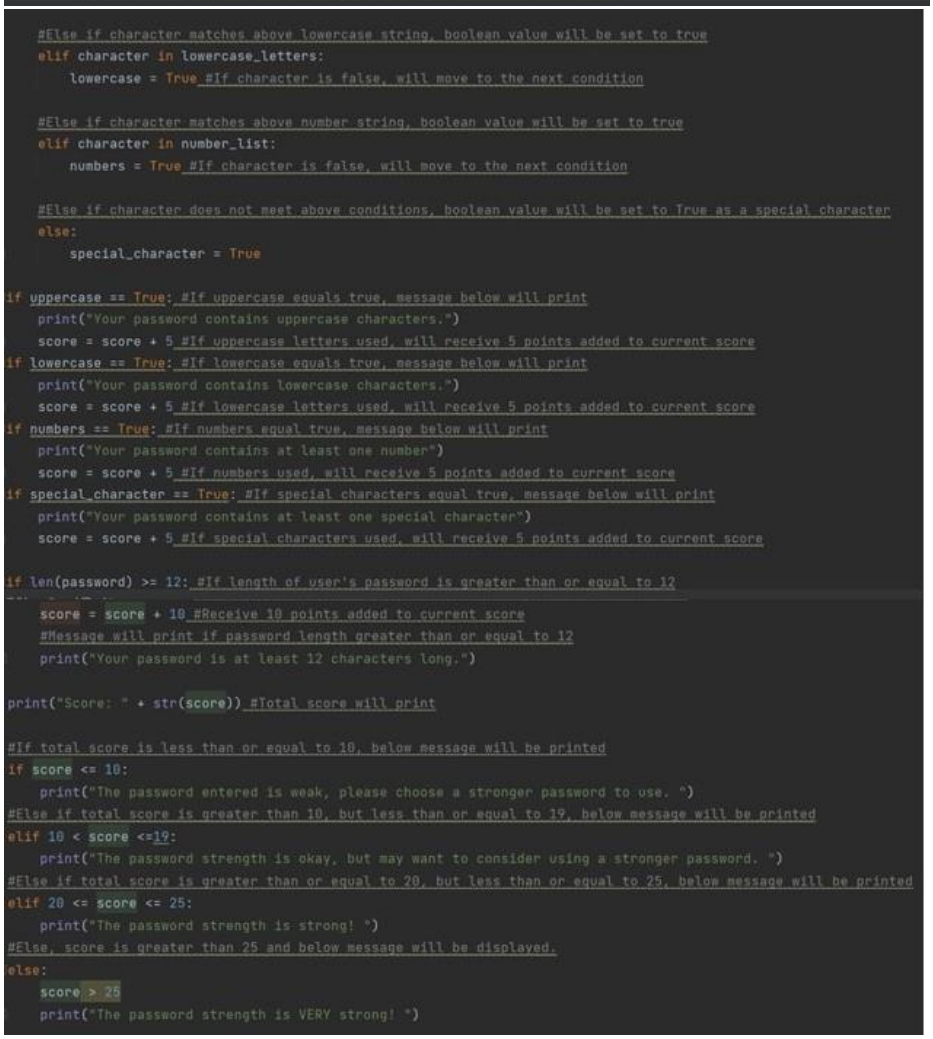
```
WELCOME TO OUR PASSWORD GENERATOR!!!! PLEASE ANSWER THE QUESTIONS BELOW TO RECEIVE YOUR PASSWORD(S).
How many passwords would you like generated? : 5
How long would you like your password to be? (Number of Characters) : 16
Here is your generated password :  Rye\/VqFJdPT~dF0
Here is your generated password :  q\X=F@)@!*Al3R0V
Here is your generated password :  ?Ex%p6^hZagVIM&#
Here is your generated password :  0HrA,U11^8c.}%78
Here is your generated password :  sE,E]r?YYO#I<-+0
How many passwords would you like generated? : 8
How long would you like your password to be? (Number of Characters) : 20
Here is your generated password :  mg=~#B'NWpyFyO-(4eC,
Here is your generated password :  %{uM:q[Jvri;6MtGtm\2
Here is your generated password :  oFDLUGDRezMK&2>1[.\-
Here is your generated password :  ,5qgM/C5jkL{C84vX)*l
Here is your generated password :  2!Ui#6G}ONS/v[!i8Bmh
Here is your generated password :  9*N6<^X.'=2=<@Uj6D:>
Here is your generated password :  _z\5x1V{?p![qi+l%s7+
Here is your generated password :  5qw*cLPYf:;wCE.qF2KV
How many passwords would you like generated? : 0
Thank you for using our Password Generator! Have a nice day!

Process finished with exit code 0
```

Random Password Generator and Password Strength Checker

Password Strength Checker Program (Python):



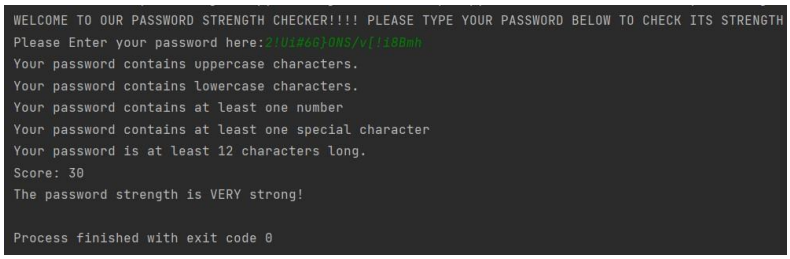
```python
#Displaying a password strength checker welcome message
print("WELCOME TO OUR PASSWORD STRENGTH CHECKER!!!! PLEASE TYPE YOUR PASSWORD BELOW TO CHECK ITS STRENGTH")

password = input("Please Enter your password here:") #User will input password they want strength checked
score = 0 #Default score if left blank

uppercase_letters = "ABCDEFGHIJKLMNOPQRSTUVWXYZ" #Uppercase letters from A-Z to be used in password strength checker
lowercase_letters = uppercase_letters.lower() #Using above uppercase letters with the .lower() to lowercase the string
number_list = "1234567890" #Numbers from 0-9 to be included in password strength checker
#Symbols/Special characters to be used in password strength checker
special_character_list = "!@#$%^&*()-_=+[]{};:',.<>""\/~?"

uppercase = False #No uppercase letters in user's password will give a false boolean value
lowercase = False #No lowercase letters in user's password will give a false boolean value
numbers = False #No numbers in user's password will give a false boolean value
special_character = False #No special characters in user's password will give a false boolean value

with open("/Users/tonel/Desktop/test.txt", encoding='utf-8') as file: #Opens a list of common used passwords for reading
    common_list = file.read() #Common list will be opened and read

if password in common_list: #User's password will be checked against list first
    #If password entered is in list, message will print and password checker will be exited
    print("Password was found in a common list!!!! Select a different password to use! Score: 0 ")
    exit()
#If user's password is not in list, will proceed to next condition
for character in password: #For every character in password
    #If character matches above uppercase string, boolean value will be set to true
    if character in uppercase_letters:
        uppercase = True #If character is false, will move to the next condition
```



```python
    #Else if character matches above lowercase string, boolean value will be set to true
    elif character in lowercase_letters:
        lowercase = True #If character is false, will move to the next condition

    #Else if character matches above number string, boolean value will be set to true
    elif character in number_list:
        numbers = True #If character is false, will move to the next condition

    #Else if character does not meet above conditions, boolean value will be set to True as a special character
    else:
        special_character = True

if uppercase == True: #If uppercase equals true, message below will print
    print("Your password contains uppercase characters.")
    score = score + 5 #If uppercase letters used, will receive 5 points added to current score
if lowercase == True: #If lowercase equals true, message below will print
    print("Your password contains lowercase characters.")
    score = score + 5 #If lowercase letters used, will receive 5 points added to current score
if numbers == True: #If numbers equal true, message below will print
    print("Your password contains at least one number")
    score = score + 5 #If numbers used, will receive 5 points added to current score
if special_character == True: #If special characters equal true, message below will print
    print("Your password contains at least one special character")
    score = score + 5 #If special characters used, will receive 5 points added to current score

if len(password) >= 12: #If length of user's password is greater than or equal to 12
    score = score + 10 #Receive 10 points added to current score
    #Message will print if password length greater than or equal to 12
    print("Your password is at least 12 characters long.")

print("Score: " + str(score)) #Total score will print

#If total score is less than or equal to 10, below message will be printed
if score <= 10:
    print("The password entered is weak, please choose a stronger password to use. ")
#Else if total score is greater than 10, but less than or equal to 19, below message will be printed
elif 10 < score <=19:
    print("The password strength is okay, but may want to consider using a stronger password. ")
#Else if total score is greater than or equal to 20, but less than or equal to 25, below message will be printed
elif 20 <= score <= 25:
    print("The password strength is strong! ")
#Else, score is greater than 25 and below message will be displayed.
else:
    score > 25
    print("The password strength is VERY strong! ")
```

Random Password Generator and Password Strength Checker

Password Strength Checker Run Results (Python):



The random password generator program successfully created passwords based on the above results. The generator was run three times, the first run with five passwords created with a length of 16, the second run with eight passwords created with a length of 20, and the third run was with zero passwords created to test the exit function from the loop.

The password strength checker program successfully checked and scored the inputted password based on the above results. One of the passwords that was created from the random password generator was inputted into the password strength checker and scored a 30 out of 30 for password strength. The results showed and demonstrated that the random password generator and the password strength checker both were programmed correctly and executed their respective programs successfully. Using the class material that was given and also taught throughout the semester, along with the internet as a resource, this project's level of complexity was moderate. The reason this project is considered moderate is because of the creation of two separate programs, the password generator and the checker to test the password's strength. Each program has its differences, but being that we were taught these concepts during the semester and allowed to practice these concepts during class made understanding each line of code needed easier and easier to explain. One of either program could have been done easily and verified using an online resource, but executing both programs allowed for the opportunity to continue to become more proficient in using Python.

References

Python 3.11.1 documentation. 3.11.1 Documentation. (n.d.). Retrieved December 5, 2022, from https://docs.python.org/3/

NeuralNine. (2020, October 26). *Simple password generator in Python*. YouTube. Retrieved December 5, 2022, from https://www.youtube.com/watch?v=rHTwjV1ORUQ

Godinho, J. (2020, August 13). *Python tutorial : How to create a random password generator using Python for Beginners*. YouTube. Retrieved December 5, 2022, from https://www.youtube.com/watch?v=qgwEs36D_Xc

Learn python by example. PythonForBeginners.com. (2022, January 2). Retrieved December 5, 2022, from https://www.pythonforbeginners.com/

Real Python. (n.d.). *Python tutorials*. Real Python. Retrieved December 5, 2022, from https://realpython.com/

Random Password Generator and Password Strength Checker

Python archives. PYnative. (n.d.). Retrieved December 5, 2022, from https://pynative.com/python/

g0tmi1k. (n.d.). *PASSWORDS/COMMON-CREDENTIALS/10-MILLION-PASSWORD-LIST-TOP-1000000.TXT · Kali/master · Kali Linux / packages / seclists · GITLAB*. GitLab. Retrieved December 5, 2022, from https://gitlab.com/kalilinux/packages/seclists/-/blob/kali/master/Passwords/Common-Credentials/10-million-password-list-top-1000000.txt

Password checker | 101 computing. (n.d.). Retrieved December 5, 2022, from https://www.101computing.net/password-checker/

How secure is my password?: Password strength checker. Security.org. (2022, October 5). Retrieved December 5, 2022, from https://www.security.org/how-secure-is-my-password/

Python tutorial. (n.d.). Retrieved December 5, 2022, from https://www.w3schools.com/python/default.asp

Python programming language. GeeksforGeeks. (n.d.). Retrieved December 5, 2022, from https://www.geeksforgeeks.org/python-programming-language/?ref=shm