CODE FIRST GIRLS

[dstl]

PYTHON CHALLENGE 2023

GIVING WOMEN THE UNFAIR ADVANTAGE

Password cracker using Brute Force technique

Helen Shortland

Sara Ricchiuti

Wanjiku Kabue




Introduction

- The Brute Force technique refers to a method used to test all the possible solutions for a problem against the actual solution.
- The purpose of this presentation is to demonstrate the significance of having passwords with a considerable length, as well as why it is advised to include a specific number of characters, such as alphabets, numbers, and special characters.

Method presentation

- Used random and pyautogui library
- Pyautogui to generate user input
- random.choices() as returns random generation from population (meaning also multiples) for length specified vs itertools permutations
- Simple while loop and if/else logic to match generated password against user input password

Method presentation

- Advantage is that random character combination will always be found
- Disadvantage is that it only works well for password lengths up to 4 characters or generation of potential passwords from smaller selection of characters
- Would look to develop code further that could potentially utilise multiple computer cores to possibly crack longer length passwords

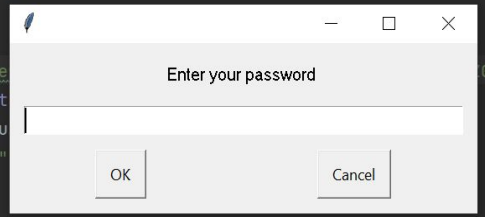
Code snapshot

Team_Gold_Python_Challenge.py

```python
import random
import pyautogui


chars = "abcdefghijklmnopqrstuvwxyzABCDEFGHIJKLMNOPQESTUVWXYZ0123456789"
allchar = list(chars)
pwd = pyautogui.password("Enter your password ")
sample_pwd = ""


while sample_pwd != pwd:
    sample_pwd = random.choices(allchar, k=len(pwd))
    if sample_pwd != list(pwd):
        continue
    else:
        print("The password is:", "".join(sample_pwd))
        break
```

Team_Gold_Python_Challenge.py

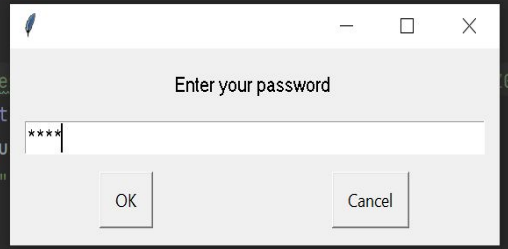
```python
import random
import pyautogui


chars = "abcde                                    0123456789"
allchar = list
pwd = pyautogu
sample_pwd = "


while sample_pwd != pwd:
    sample_pwd = random.choices(allchar, k=len(pwd))
    if sample_pwd != list(pwd):
        continue
    else:
        print("The password is:", "".join(sample_pwd))
        break
```



Code snapshot

Team_Gold_Python_Challenge.py

```python
import random
import pyautogui


chars = "abcde                                                    0123456789"
allchar = list
pwd = pyautogu
sample_pwd = "


while sample_pwd != pwd:
    sample_pwd = random.choices(allchar, k=len(pwd))
    if sample_pwd != list(pwd):
        continue
    else:
        print("The password is:", "".join(sample_pwd))
        break
```



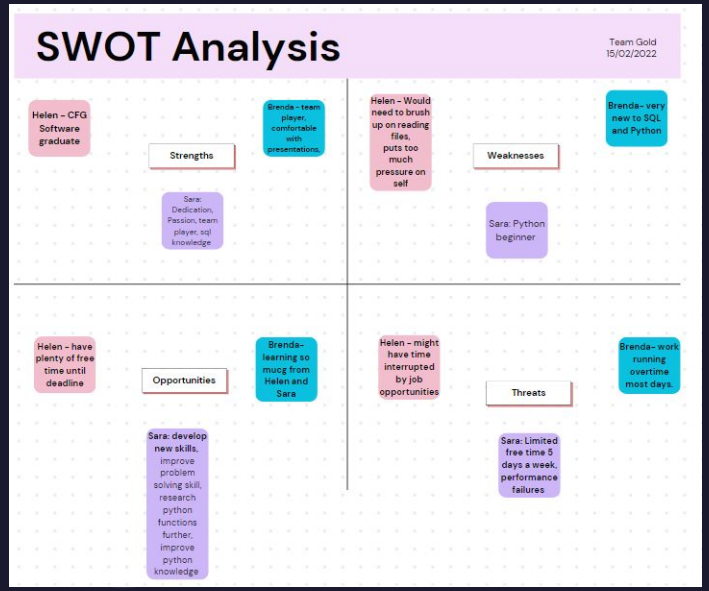
Team_Gold_Python_Challenge.py

```python
import random
import pyautogui


chars = "abcdefghijklmnopqrstuvwxyzABCDEFGHIJKLMNOPQESTUVWXYZ0123456789"
allchar = list(chars)
pwd = pyautogui.password("Enter your password ")
sample_pwd = ""


while sample_pwd != pwd:
    sample_pwd = random.choices(allchar, k=len(pwd))
    if sample_pwd != list(pwd):
        continue
    else:
        print("The password is:", "".join(sample_pwd))
        break
```

Run: Team_Gold_Python_Challenge

```
C:\Users\hssho\PycharmProjects\cfg-python\venv\Scripts\python.exe C:\Users
The password is: cat9

Process finished with exit code 0
```

Team-Gold Journey



First Meeting:
- Team bonding
- SWOT Analysis on team members
- Broad discussion of the task
- First code draft

Third Meeting:
- Decided to simplify the code for speed purpose
- Gathered presentation feedback from each team member

16th February | 20th February | 22nd February | 23rd February

Second Meeting:
- Change of strategy
- Code implementation
- First presentation draft

Fourth Meeting:
- Finalised the presentation
- Nomination the persons presenting
- Submission

Team meeting minutes:

AGENDA MEETING 1

- Discuss brute force methods for testing passwords
- Determine approach for generating permutations to test against password cases
- Determine what types of passwords to test against
- Identify library to assist with generating permutations in Python
- Plan for testing against standard and hashed passwords
- Schedule follow-up meeting

AGENDA MEETING 2

- Review code
- Determine that the new code is suitable for future use collaborated using a debugger to fix the code.
- After reviewing the group determined that the new code developed is more suitable
- The group agreed to explore adding hashing to enhance the code
- Discuss potential improvements
- Plan for follow-up meeting

AGENDA MEETING 3

- The team discussed the inclusion of hashing and decided not to include it as it would increase the length and time required to generate permutations
- The importance of password length and variety of characters was emphasized as a means to protect against brute force hacking
- Follow-up meeting the following day to discuss the code and the presentation further before the submission

AGENDA MEETING 4

- Presentation design finalised
- Submission

Conclusion

In conclusion, we applied the Brute Force method to test user-generated passwords. Through this presentation, we aimed to showcase the importance of creating passwords with a considerable length and a specific combination of characters, including alphabets, numbers, and special characters. It is crucial to prioritize password strength and use a combination of characters to enhance security

Thank You

Any Questions?