Betriebswirtschaftslehre

Herr Baun

Passwort Manager

Unsere Gruppe hat sich für die Aufgabenstellung Password-Manager entschieden.

Einer der festgelegten Aufgabenstellungen ist es, dass der Nutzer selbst entscheiden kann, ob er ein eigenes Passwort für eine bestimmte Webseite erstellen möchte oder das Programm eins für ihn bereitstellt.
Jedoch wird hier abgefragt, wie lang sein Passwort sein muss, wie viele Großbuchstaben, Kleinbuchstaben und Sonderzeichen es enthalten solle.

Unser Quellcode beginnt mit dem zufälligen Generieren des Passwortes.
Dafür wurden Methoden erstellt, die die Länge des Passwortes überprüfen. Falls der Nutzer die Anzahl der Groß-und Kleinbuchstaben, sowie die Anzahl der Sonderzeichen eingibt, diese in der Summe gleich sein müssen, wie die Passwortlänge.
Wenn es nicht der Fall sei, wird der Nutzer darauf hingewiesen und es kommt zu der erneuten Frage, es diesmal in der richtigen Anzahl anzugeben. Sonst erscheint ein Befehl noch, falls der Nutzer das Programm verlassen will, er „off“ in der Benutzereingabe schreiben kann.
Bei der nächsten Methode namens def passGenerieren(): werden die einzelnen Variablen angezeigt, die beim Erstellen des Passwortes wichtig sind. Damit das Programm weiß, mit was er das Passwort generieren soll, haben wir 3 Variabel erstellt.
Einmal für die Groß- und Kleinbuchstaben nutzen wir den Ascii Code. (Für Großbuchstaben = ascii_uppercase und für die Kleinbuchstaben ascii_lowercase.
Für die Variabel Zeichen erstellten wir selbst ein ???.
Demnächst wurde dann ein Input zu den einzelnen ??? erstellt. Hier kommt die Eingabe des Nutzers. Und im Hintergrund wird ein zufälliges Passwort für die Person zusammengewürfelt, nach seinen Wünschen.
Zahra
Zusätzlich fügten wir die Funktion Pyperclip ein. Diese dient dazu, dass das Passwort im Clipboard gespeichert wird. Dafür muss man zuerst Pyperclip importieren. Dies allein reicht nicht aus, sondern man muss beim es Terminal noch hinzukommend installieren. Dazu gibt man nur den Befehl: pip Install pyperclip ein.

Als nächstes beschäftigten wir uns mit dem Masterpasswort.
Das Masterpasswort ist eine wichtige Funktion, denn durch das Masterpasswort gelangt der Nutzer den Zugriff auf sein Passwort-Manager.
Dem Benutzer wird hier als aller erstes angefordert sich ein Masterpasswort zu erstellen. Diese steht ihm frei zu ohne jegliche Anforderungen.
Demnächst wird von dem Nutzer wieder verlangt sein erstelltes Masterpasswort einzugeben.

Wir erstellten hier eine While Schleife, denn erst mit der richtigen Eingabe werden ihm die 5 verschiedene Möglichkeiten angezeigt. Falls das falsche Passwort eingegeben wurde, bekommt der Benutzer eine Ausgabe mit „Falsches Passwort“ und das Programm wird auch schon beendet.
Wir erstellten fünf verschiedene Optionen in unserem Passwort-Manager.
Diese werden ihm angezeigt und ihm steht die freie Wahl und wurden mit if, elif,else Befehlen durchgeführt.

Die erste Möglichkeit ist es, seine Passwörter angezeigt zu bekommen mit der dazugehörigen Webseite und dem Benutzernamen.
Wir haben uns dafür entschieden die ganzen Daten in einem File zusichern. Es kamen auch Versuche die Datei in einer Datenbank speichern zu können, aber nach stundenlanger Recherche, kamen wir nicht viel weiter und wählten uns das Speichern in einer File aus, namens password.txt.
Am Anfang vom File steht, noch ein Hinweis, dass falls dem Benutzer nichts angezeigt wird, es noch keine Passwörter gespeichert worden sind.
Zur Anzeige der Dateien vom Nutzer mussten wir den File zuerst öffnen mit dem Befehl
with open (‚password.txt', ‚r') as f:
‚r' dient zum Lesen der File mit f.read(), welches dann mit f_contents gleich gesetzt wurde und der Befehl aufkam, f_contents auszugeben mit print().
Nachdem die Daten des Nutzers abgebildet worden sind, kann sich die Person sein weiteren Zug auswählen. Hierfür werden ihm wieder die 5 Auswahlmöglichkeiten angezeigt.

In der zweiten Option geht es um die Passwort Erstellung.
Zum generieren erstellten wir, wie oben schon detailliert genannt, Methoden, damit es passend erstellt wird.
Hier werden die einzelnen Dinge vom Nutzer abgefragt.
Zuallererst nach dem Namen der Webseite. Wir fügten noch inter dem Input das Befehl .lower() hinzu. Dies dient dazu, falls der Benutzer den Namen der Webseite groß schreiben sollte, wird durch .lower() daraus Kleinbuchstaben werden und unser Programm erkennt, um welche Webseite es sich hier handelt, denn durch die falsche Groß- und Kleinbuchstabe diese Eingabe später mit dem gespeichertem nicht erkannt wurde.
Im folgendem wird nach dem Benutzernamen gefragt.
Daraufhin stellt man dem Nutzer die Möglichkeit sein Passwort vom Programm generieren zu lassen oder sich sein eigenes auszuwählen.
Dafür muss er entweder den Buchstaben g (für generieren des Passwortes) oder e für (eigenes Passwort).
Wir haben auch hierfür eine if, elif, else Befehl eingefügt.
Wenn der Nutzer den Buchstaben g eingibt, wird der Befehl passGenerieren() aufgerufen.
Ihm werden dann nacheinander die einzelnen Fragen zum Passwortgenerieren gefragt.

laenge = int(input(”Bitte geben Sie die Länge des Passwortes ein: ”))
grosbuchstaben = int(input(„Bitte geben Sie die Anzahl der Großbuchstaben ein:“))
grosbuchstaben=int(pruefGroßBuchstaben(grosbuchstaben,laenge))
pw = pw + (,‘.join(secrets.choice(buchstabenGroß) for _ in range (großbuchstaben)))
kleinbuchstaben = int(input(input(„Bitte geben Sie die Anzahl der Kleinbuchstaben ein:"))
kleinbuchstaben=int(pruefKleineBuchstaben(kleinebuchstaben,laenge,grosbuchstaben))
pw = pw + (,‘.join(secrets.choice(buchstabenKlein) for _ in range (kleinbuchstaben)))
sonderzeichen = int(input(„Bitte geben Sie die Anzahl der Sonderzeichen ein:“))
sonderzeichen =int(pruefSonderZeichen(sonderzeichen,laenge))

Zum besseren Verständnis habe ich die Befehle aufgezählt. Wir fügten vor dem input ein integer hinzu, damit Python weiß, dass nur Zahlen als Anwortmöglichkeiten gelten.
Falls der Benutzer doch ein String eingibt, ihm ein Fehler angezeigt wird und dies nicht gültig ist für diesen Input.
Nach dem Input wird hier geprüft, ob die Anzahl der angegebenen Groß- und Kleinbuchstaben oder Sonderzeichen mit der angegeben Länge übereinstimmen.
Daraufhin wird dann ein zufälliger Buchstabe oder Zeichen aus der Variablen ausgewählt und ein Passwort erstellt.
Dies geschieht mit der Anforderung(secrets.choice).
Hierzu erstellten wir ein Import Secrets, damit man auf secrets.choice zugreifen kann.
Wenn der Nutzer den Buchstaben e eingibt, kommt die Frage mit dem Input "Bitte geben Sie ihr Passwort ein:")
Die Passwörter werden dann durch pyperclip.copy in der Zwischenlage kopiert und der Nutzer bekommt auch dementsprechend ein Hinweis mit "Dein erstelltes Passwort wurde im Clipboard gespeichert.")

Wir definierten dann zusätzlich noch eine Methode namens def fe(): .
Hier wurde unser file password.txt wieder aufgerufen, nur diesmal mit ein 'a' , was zu bedeuten hat, dass Informationen in die File hinzugefügt wird.
In diesem Falle werden neue Daten hinzugefügt, wie das neu erstellte Passwort und die dazugehörigen Benutzernamen und der Webseite.
Dies geschieht auch mit dem Befehl .write({} {} {}). In die geschweiften Klammern kommen die deklarierten Variablen ws, bn und pw.
ws steht bei uns für die Eingabe der Webseite, bn für die Eingabe des Benuternamen und pw für das erstellte Passwort.
Mit dieser Methode wurden die neu angegeben Daten in der File Datei gespeichert.
Und zuletzt wird die def fe(): Methode mit dem Befehl fe() aufgerufen.

Als nächste Option bieten wir dem Benutzer die Möglichkeit sein Passwort löschen zu können.