Task - 03: Password Complexity Checker

This project implements a Password Complexity Checker in Python. The tool evaluates the strength of a password based on several criteria such as length, use of uppercase and lowercase letters, numbers, and special characters. It provides real-time feedback on whether the password is Strong, Moderate, or Weak.

Algorithm:

1. Step 1: Input a password from the user.
2. Step 2: Check the password length (>= 8 characters).
3. Step 3: Check for at least one uppercase letter.
4. Step 4: Check for at least one lowercase letter.
5. Step 5: Check for at least one number.
6. Step 6: Check for at least one special character (e.g., @, #, $, %).
7. Step 7: Assign strength points for each satisfied condition.
8. Step 8: Classify the password as Strong, Moderate, or Weak.
9. Step 9: Display feedback and suggestions to the user.

Example Run:

Input: abc
Output: Weak password ■
- Password should be at least 8 characters long.
- Add at least one uppercase letter.
- Add at least one number.
- Add at least one special character (e.g. @, #, $, %).

Input: Abc1234
Output: Moderate password ■■
- Password should be at least 8 characters long.
- Add at least one special character (e.g. @, #, $, %).

Input: Abc1234@
Output: Strong password ■

This project demonstrates how to use Python's string handling and regular expressions to implement a real-world utility program. The Password Complexity Checker ensures better security by guiding users to create stronger and more reliable passwords.