Password Strength Evaluation Report

Objective:

To understand what constitutes a strong password and evaluate its strength using online tools.

Passwords Created for Evaluation:

1. **Simple Password**: password123
2. **Moderate Complexity**: Summer2025!
3. **High Complexity**: G7r$2nV!lQ9zX8
4. **Passphrase**: BlueSky$Dances!UnderMoon
5. **Randomly Generated**: 9u$W2f!Kq8Zx@L

Evaluation Using Online Password Strength Checker:

Using the online tool PasswordMeter, the following evaluations were made:

Password	**Strength Score**	**Feedback**
password123	Weak	Common password; easily guessable.
Summer2025!	Moderate	Contains uppercase, numbers, and symbols; still predictable.
G7r$2nV!lQ9zX8	Strong	High entropy; includes uppercase, lowercase, numbers, and symbols.
BlueSky$Dances!UnderMoon	Very Strong	Long passphrase; highly unpredictable.
9u$W2f!Kq8Zx@L	Very Strong	Randomly generated; high complexity.

Analysis and Best Practices:

- **Length Over Complexity**: Longer passwords are generally more secure. Aim for at least 16 characters. cisa.gov+3the-sun.com+3it.ucsb.edu+3
- **Avoid Common Patterns**: Passwords like password123 are easily guessable. cu.edu

- **Use Passphrases**: Combining unrelated words can create memorable yet strong passwords. cyber.gc.ca
- **Randomness is Key**: Randomly generated passwords with a mix of characters are harder to crack. acaglobal.com
- **Unique Passwords for Each Account**: Reusing passwords increases vulnerability. cisa.gov+1acaglobal.com+1

Python Code to Check Password Strength:

Python

```
import re


def check_password_strength(password):
    if len(password) < 12:
        return "Weak: Password must be at least 12 characters long."
    if not re.search(r"[a-z]", password):
        return "Weak: Password must contain at least one lowercase letter."
    if not re.search(r"[A-Z]", password):
        return "Weak: Password must contain at least one uppercase letter."
    if not re.search(r"[0-9]", password):
        return "Weak: Password must contain at least one digit."
    if not re.search(r"[!@#$%^&*(),.?\":{}|<>]", password):
        return "Weak: Password must contain at least one special character."
    return "Strong: Password meets all criteria."


# Ask user to input a password
user_password = input("Enter a password to check its strength: ")
result = check_password_strength(user_password)
print(f"Password: {user_password} -> {result}")
```
