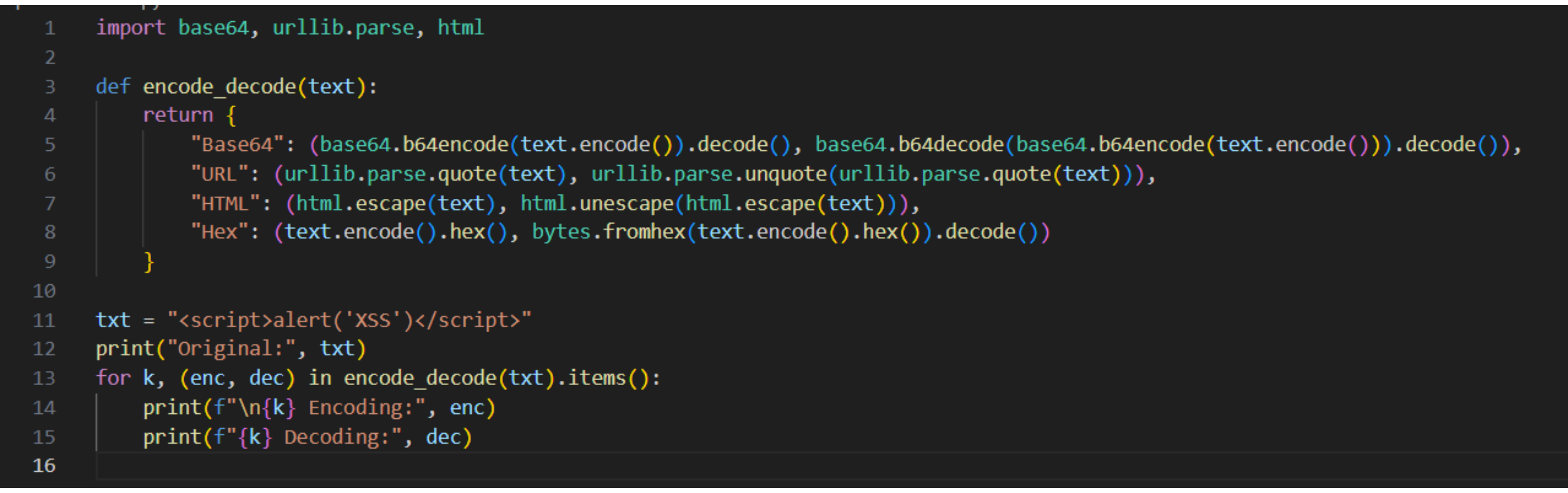
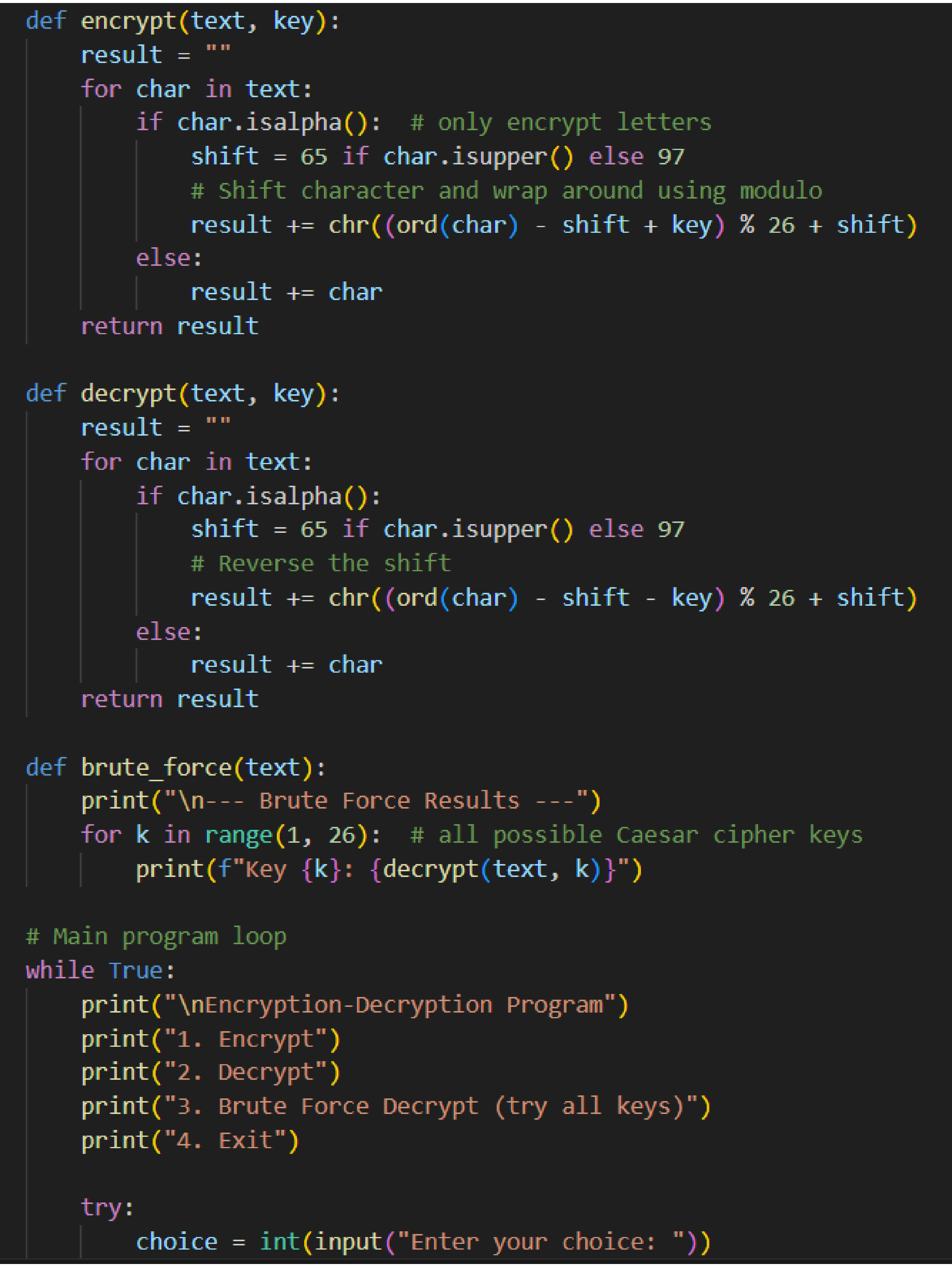
1.py

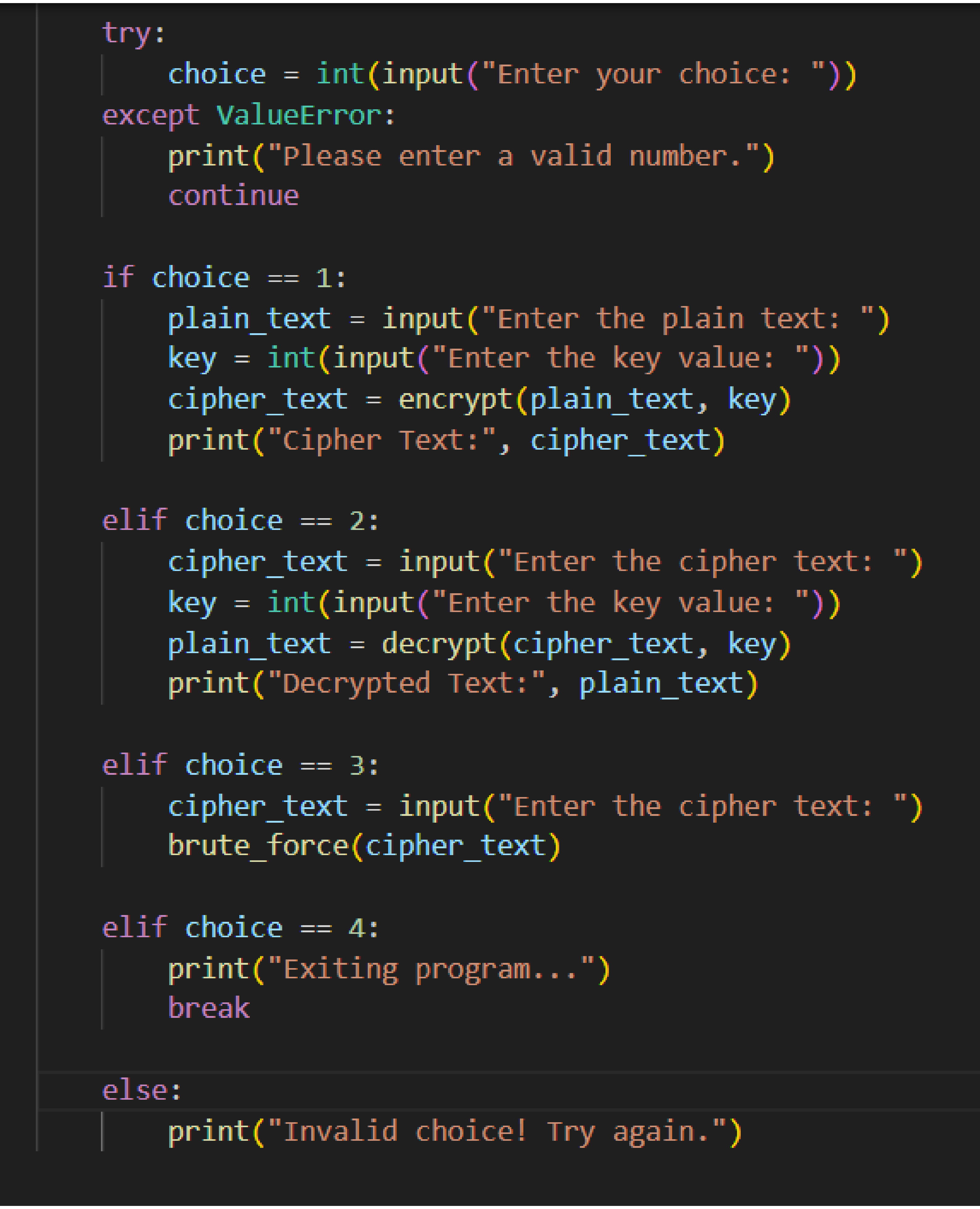


```python
import base64, urllib.parse, html

def encode_decode(text):
    return {
        "Base64": (base64.b64encode(text.encode()).decode(), base64.b64decode(base64.b64encode(text.encode())).decode()),
        "URL": (urllib.parse.quote(text), urllib.parse.unquote(urllib.parse.quote(text))),
        "HTML": (html.escape(text), html.unescape(html.escape(text))),
        "Hex": (text.encode().hex(), bytes.fromhex(text.encode().hex()).decode())
    }

txt = "<script>alert('XSS')</script>"
print("Original:", txt)
for k, (enc, dec) in encode_decode(txt).items():
    print(f"\n{k} Encoding:", enc)
    print(f"{k} Decoding:", dec)
```

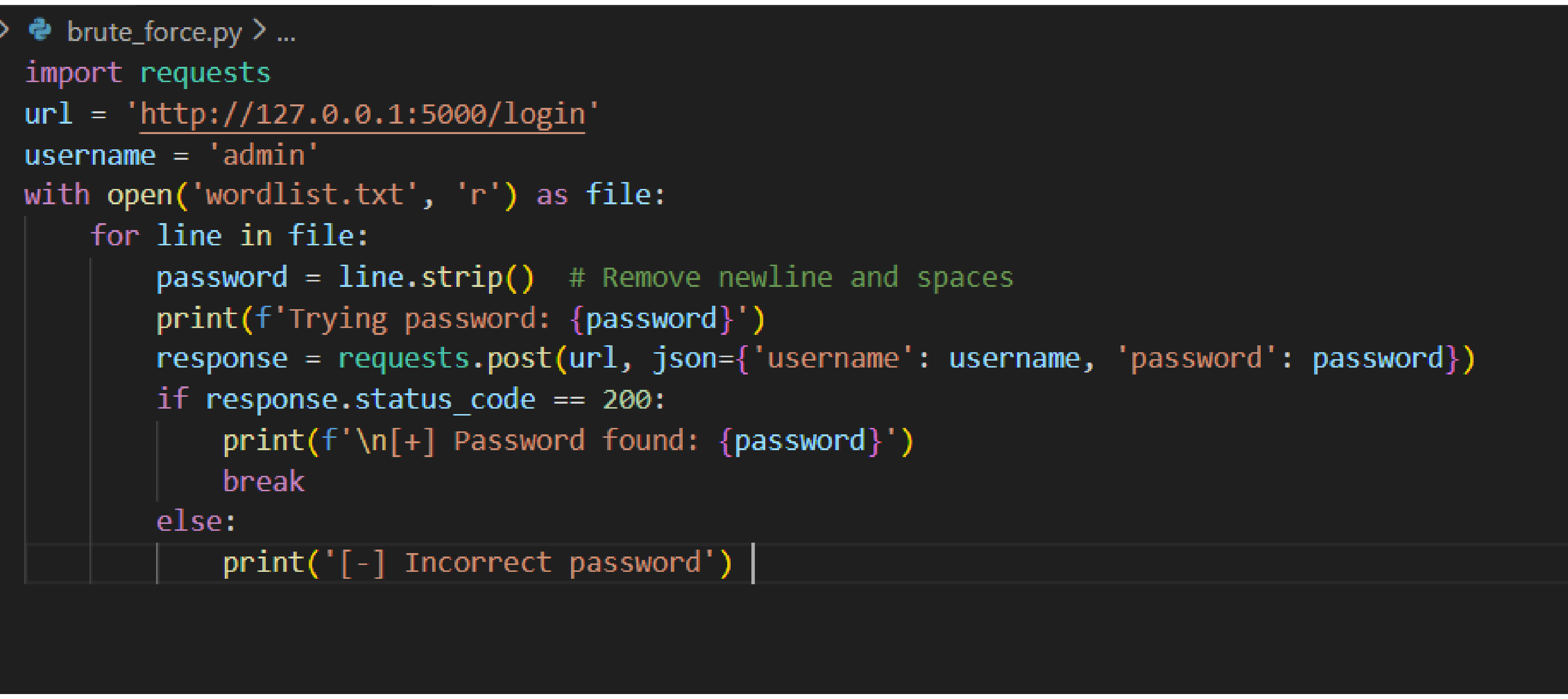
enc, dec



```python
def encrypt(text, key):
    result = ""
    for char in text:
        if char.isalpha():  # only encrypt letters
            shift = 65 if char.isupper() else 97
            # Shift character and wrap around using modulo
            result += chr((ord(char) - shift + key) % 26 + shift)
        else:
            result += char
    return result

def decrypt(text, key):
    result = ""
    for char in text:
        if char.isalpha():
            shift = 65 if char.isupper() else 97
            # Reverse the shift
            result += chr((ord(char) - shift - key) % 26 + shift)
        else:
            result += char
    return result

def brute_force(text):
    print("\n--- Brute Force Results ---")
    for k in range(1, 26):  # all possible Caesar cipher keys
        print(f"Key {k}: {decrypt(text, k)}")

# Main program loop
while True:
    print("\nEncryption-Decryption Program")
    print("1. Encrypt")
    print("2. Decrypt")
    print("3. Brute Force Decrypt (try all keys)")
    print("4. Exit")

    try:
        choice = int(input("Enter your choice: "))
```

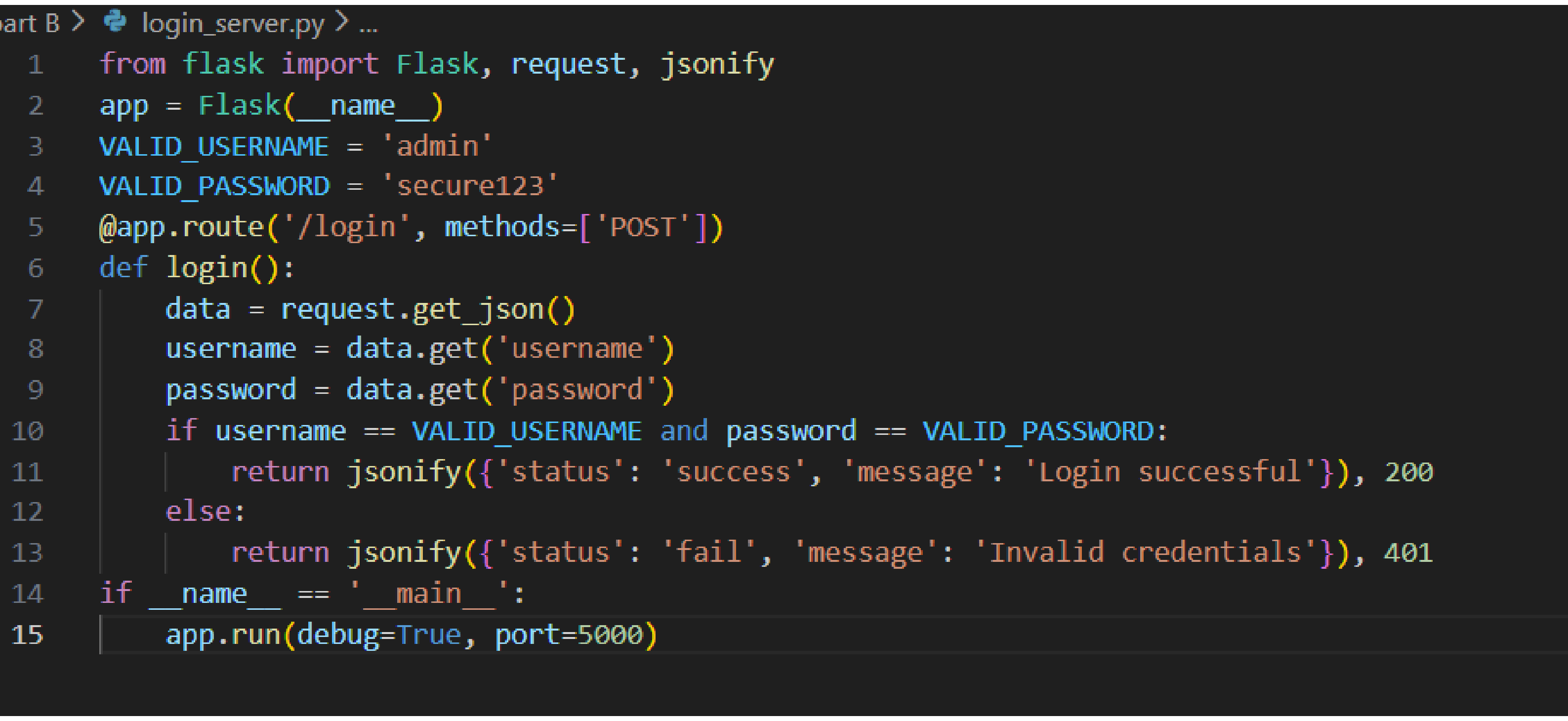


```python
    try:
        choice = int(input("Enter your choice: "))
    except ValueError:
        print("Please enter a valid number.")
        continue

    if choice == 1:
        plain_text = input("Enter the plain text: ")
        key = int(input("Enter the key value: "))
        cipher_text = encrypt(plain_text, key)
        print("Cipher Text:", cipher_text)

    elif choice == 2:
        cipher_text = input("Enter the cipher text: ")
        key = int(input("Enter the key value: "))
        plain_text = decrypt(cipher_text, key)
        print("Decrypted Text:", plain_text)

    elif choice == 3:
        cipher_text = input("Enter the cipher text: ")
        brute_force(cipher_text)

    elif choice == 4:
        print("Exiting program...")
        break

    else:
        print("Invalid choice! Try again.")
```

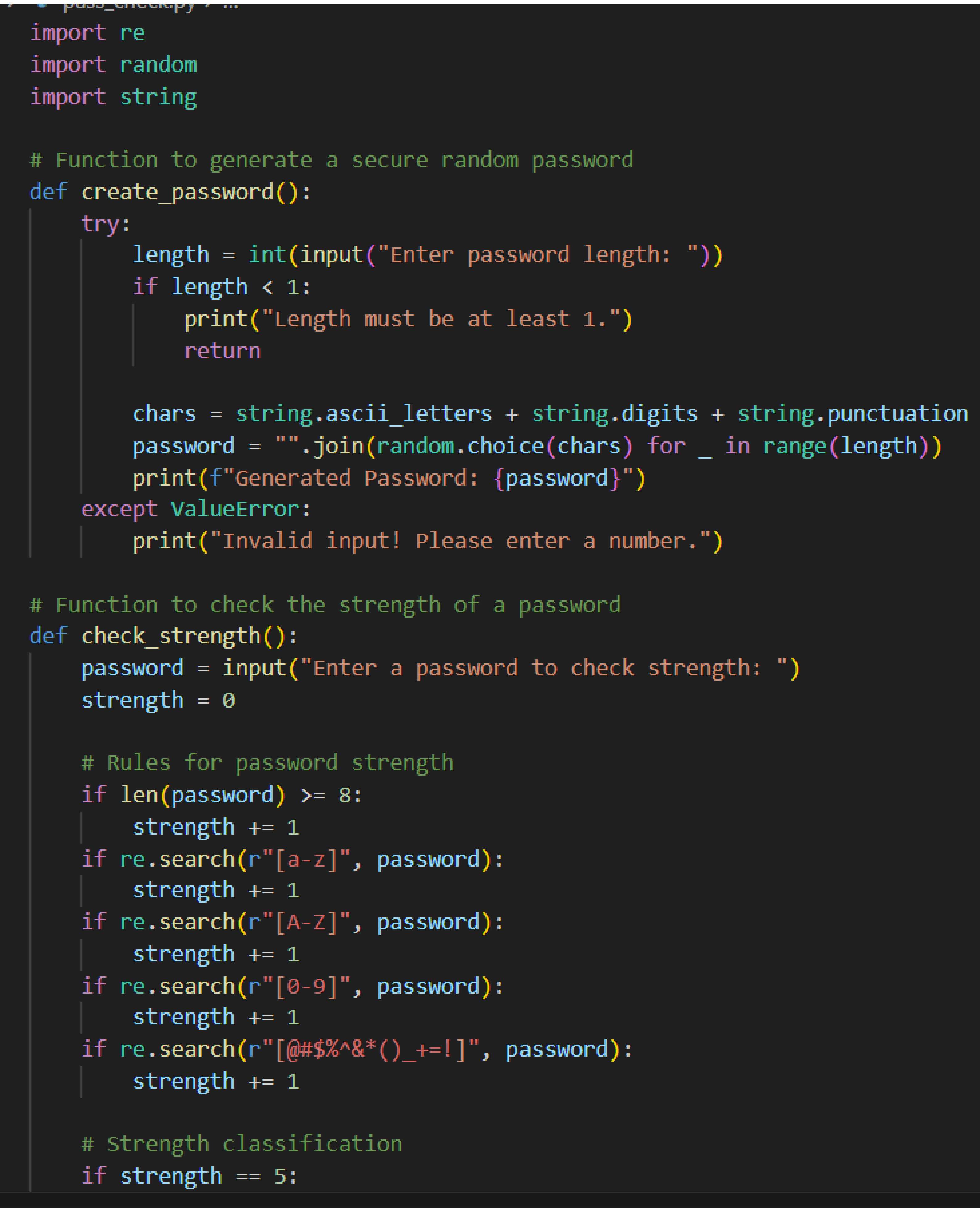


brute_force.py > ...

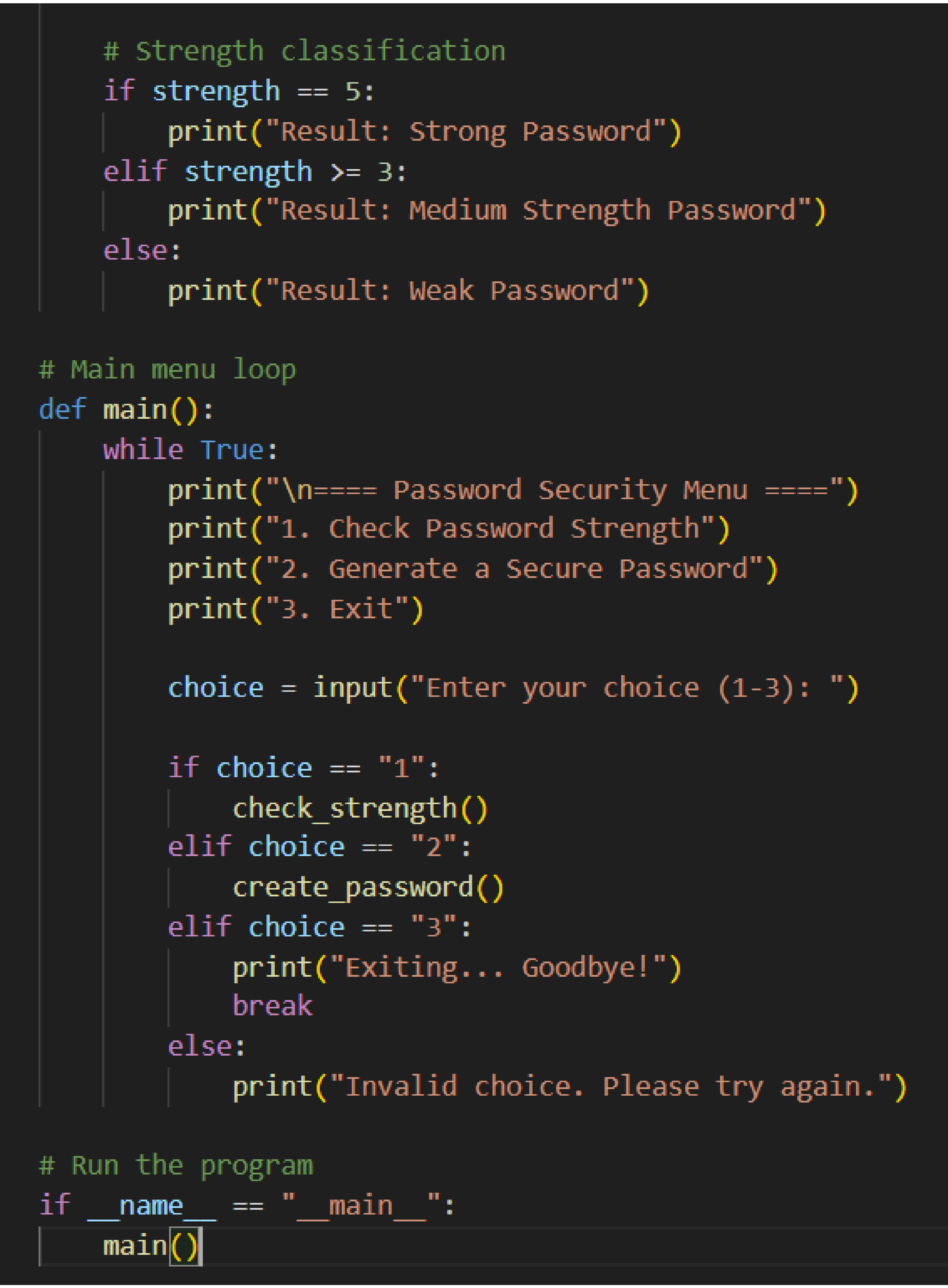
```python
import requests
url = 'http://127.0.0.1:5000/login'
username = 'admin'
with open('wordlist.txt', 'r') as file:
    for line in file:
        password = line.strip()  # Remove newline and spaces
        print(f'Trying password: {password}')
        response = requests.post(url, json={'username': username, 'password': password})
        if response.status_code == 200:
            print(f'\n[+] Password found: {password}')
            break
        else:
            print('[-] Incorrect password')
```


art B > login_server.py > ...

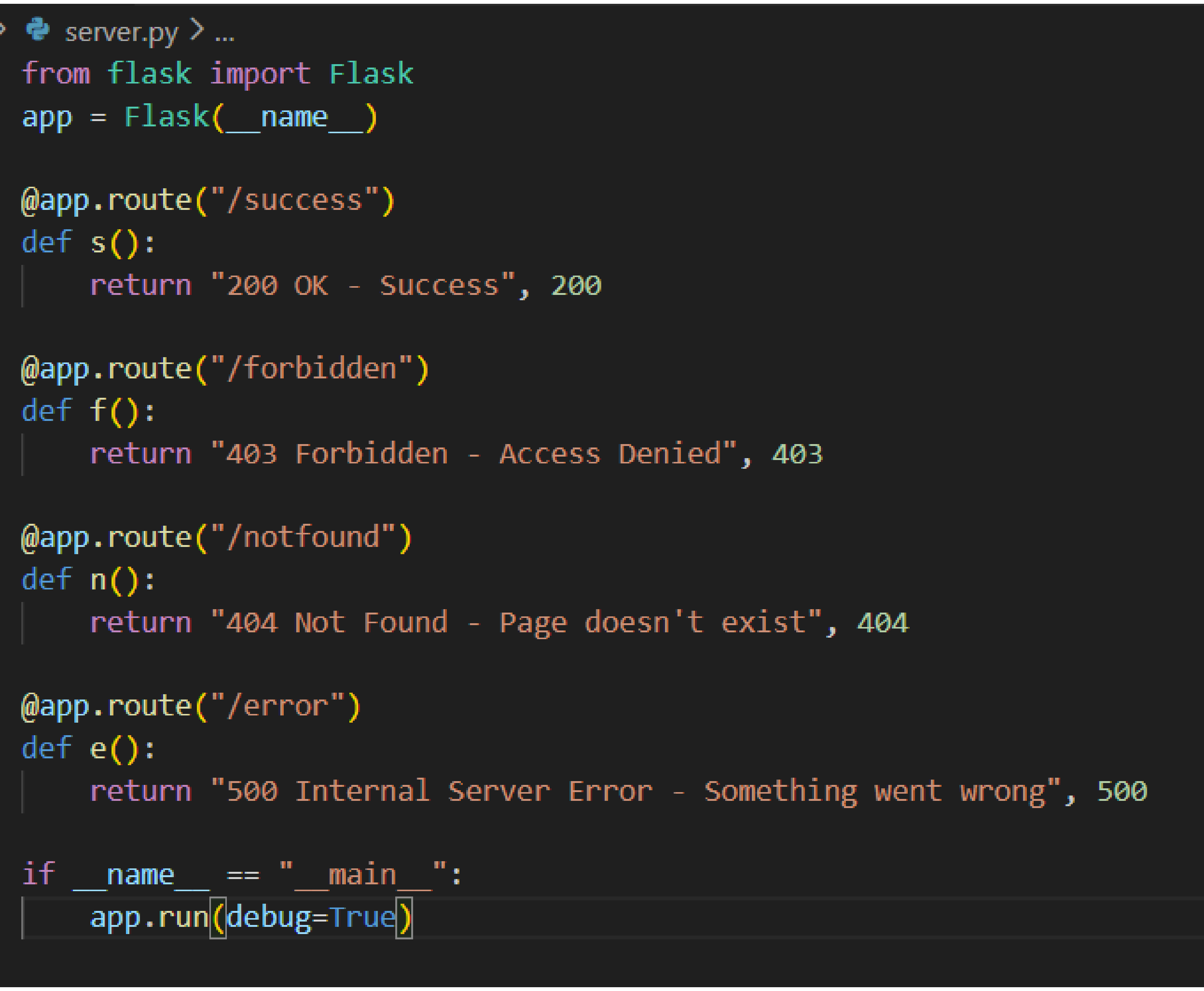
```python
from flask import Flask, request, jsonify
app = Flask(__name__)
VALID_USERNAME = 'admin'
VALID_PASSWORD = 'secure123'
@app.route('/login', methods=['POST'])
def login():
    data = request.get_json()
    username = data.get('username')
    password = data.get('password')
    if username == VALID_USERNAME and password == VALID_PASSWORD:
        return jsonify({'status': 'success', 'message': 'Login successful'}), 200
    else:
        return jsonify({'status': 'fail', 'message': 'Invalid credentials'}), 401
if __name__ == '__main__':
    app.run(debug=True, port=5000)
```



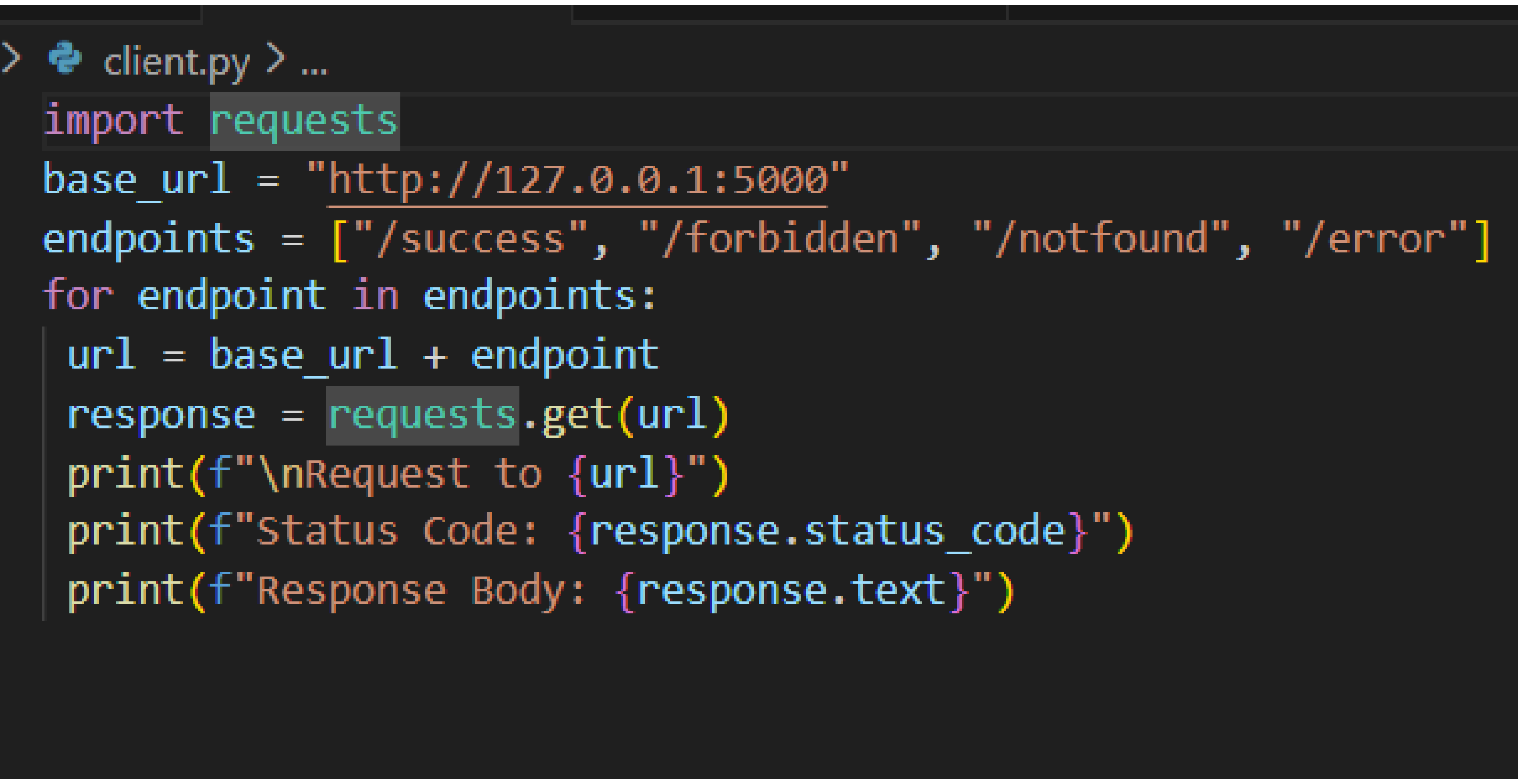
```python
import re
import random
import string

# Function to generate a secure random password
def create_password():
    try:
        length = int(input("Enter password length: "))
        if length < 1:
            print("Length must be at least 1.")
            return

        chars = string.ascii_letters + string.digits + string.punctuation
        password = "".join(random.choice(chars) for _ in range(length))
        print(f"Generated Password: {password}")
    except ValueError:
        print("Invalid input! Please enter a number.")

# Function to check the strength of a password
def check_strength():
    password = input("Enter a password to check strength: ")
    strength = 0

    # Rules for password strength
    if len(password) >= 8:
        strength += 1
    if re.search(r"[a-z]", password):
        strength += 1
    if re.search(r"[A-Z]", password):
        strength += 1
    if re.search(r"[0-9]", password):
        strength += 1
    if re.search(r"[@#$%^&*()_+=!]", password):
        strength += 1

    # Strength classification
    if strength == 5:
```



```python
    # Strength classification
    if strength == 5:
        print("Result: Strong Password")
    elif strength >= 3:
        print("Result: Medium Strength Password")
    else:
        print("Result: Weak Password")

# Main menu loop
def main():
    while True:
        print("\n==== Password Security Menu ====")
        print("1. Check Password Strength")
        print("2. Generate a Secure Password")
        print("3. Exit")

        choice = input("Enter your choice (1-3): ")

        if choice == "1":
            check_strength()
        elif choice == "2":
            create_password()
        elif choice == "3":
            print("Exiting... Goodbye!")
            break
        else:
            print("Invalid choice. Please try again.")

# Run the program
if __name__ == "__main__":
    main()
```



```python
from flask import Flask
app = Flask(__name__)

@app.route("/success")
def s():
    return "200 OK - Success", 200

@app.route("/forbidden")
def f():
    return "403 Forbidden - Access Denied", 403

@app.route("/notfound")
def n():
    return "404 Not Found - Page doesn't exist", 404

@app.route("/error")
def e():
    return "500 Internal Server Error - Something went wrong", 500

if __name__ == "__main__":
    app.run(debug=True)
```



```python
import requests
base_url = "http://127.0.0.1:5000"
endpoints = ["/success", "/forbidden", "/notfound", "/error"]
for endpoint in endpoints:
  url = base_url + endpoint
  response = requests.get(url)
  print(f"\nRequest to {url}")
  print(f"Status Code: {response.status_code}")
  print(f"Response Body: {response.text}")
```