Programming Problem-Solving seatwork 1

April 4, 2025

1 Programming Problem-Solving seatwork 1

Name: Fritz Andrew E. Flores Section: BSCpE-2A Date: April 4, 2025

1.1 Problem 1: Number Statistics Analyzer (Loops + Input Validation + Conditional Logic + Aggregation) **Instructions:** Write a program that: 1. Asks the user to input numbers continuously until they enter "done". 2. For all valid numeric entries: > - Count how many are even and how many are odd > - Find the maximum and minimum > - Compute the average 3. Display the statistics after termination.

[1]:
```
def number_statistics():
    numbers = []

    while True:
        num = input("Enter a number (or done to finish): ")
        if num.lower() == "done":
            break

        try:
            num = int(num)
            numbers += [num]
        except ValueError:
            print("Invalid input. Try again.")

    if numbers:
        evens = sum(1 for n in numbers if n % 2 == 0)
        odds = len(numbers) - evens
        print(f"Even: {evens}")
        print(f"Odd: {odds}")
        print(f"Max: {max(numbers)}")
        print(f"Min: {min(numbers)}")
        print(f"Average: {sum(numbers) / len(numbers):.1f}")
    else:
        print("No numbers entered.")

number_statistics()
```

```
Enter a number (or done to finish):  34
Enter a number (or done to finish):  22
```

```
Enter a number (or done to finish):  11
Enter a number (or done to finish):  16
Enter a number (or done to finish):  12
Enter a number (or done to finish):  8
Enter a number (or done to finish):  done

Even: 5
Odd: 1
Max: 34
Min: 8
Average: 17.2
```

1.2 Problem 2: Password Validator and Retry System (Loops + Input Validation + Control Flow + String Operations) **Instructions:** Create a password validator program that: > 1. Asks the user to enter a password. > 2. Validates that the password: > • Is at least 8 characters long > • Contains at least one uppercase letter > • Contains at least one lowercase letter > • Contains at least one digit > • Contains at least one special character (!@#$%^&*) > 3. If the password is invalid, show all validation errors. > 4. Allow the user to retry up to 3 times. > 5. If a valid password is entered, print "Password accepted". > 6. If 3 invalid attempts are made, print "Too many failed attempts. Access denied."

[1]:
```python
password = input("Enter password: ")
attempts = 3

while attempts > 0:
    errors = ""
    if len(password) < 8:
        errors += "Must be at least 8 characters\n"
    if not any(char.isupper() for char in password):
        errors += "Must include an uppercase letter\n"
    if not any(letter.islower() for letter in password):
        errors += "Must include a lowercase letter\n"
    if not any(digit.isdigit() for digit in password):
        errors += "Must include a digit\n"
    if not any(symbol in '!@#$%^&*' for symbol in password):
        errors += "Must include a special character\n"
    if errors:
        print(errors, end="")
        attempts -= 1
        print(f"Attempts left: {attempts}")
        if attempts > 0:
            password = input("Enter password: ")
    else:
        print("Password accepted")
        break
if attempts == 0:
    print("Too many failed attempts. Access denied.")
```

```
Enter password:  hello
```

```
Must be at least 8 characters
Must include an uppercase letter
Must include a digit
Must include a special character
Attempts left: 2

Enter password:  Hello123

Must include a special character
Attempts left: 1

Enter password:  Hello@123

Password accepted
```

1.3 Problem 3: Secure PIN Setup and Verification System (Loops + Conditional Logic + Input Masking + Retry Attempts) **Instructions:** Write a Python program that: > 1. Asks the user to set up a 4-digit PIN code. 2. Verifies that the PIN contains only digits and is exactly 4 characters long. 3. After a valid PIN is set, allow the user to attempt login. 4. The user has 3 attempts to input the correct PIN. 5. On success, print "fi Access granted". 6. On failure after 3 tries, print " Account locked".

[1]:
```python
while True:
    pin = input("Set your 4-digit PIN: ")
    if len(pin) == 4 and pin.isdigit():
        break
    else:
        print("Invalid PIN. Must be exactly 4 digits.")

print("PIN set successfully.")

attempts = 3
while attempts > 0:
    entered_pin = input("Enter PIN to login: ")
    if entered_pin == pin:
        print("Access granted")
        break
    else:
        attempts -= 1
        print(f"Incorrect PIN. Attempts left: {attempts}")

if attempts == 0:
    print("Account locked")
```

```
Set your 4-digit PIN:  abc

Invalid PIN. Must be exactly 4 digits.

Set your 4-digit PIN:  12345

Invalid PIN. Must be exactly 4 digits.

Set your 4-digit PIN:  abcd
```

```
Invalid PIN. Must be exactly 4 digits.
Set your 4-digit PIN:  2115
PIN set successfully.
Enter PIN to login:  1234
Incorrect PIN. Attempts left: 2
Enter PIN to login:  2345
Incorrect PIN. Attempts left: 1
Enter PIN to login:  2115
Access granted
```