Part 2 - Writing Better ChatGPT Scripts and Creating Simple Python Programs

Yahya Nazer

2026.06.18

Table of contents

Part 2 - Writing Better ChatGPT Scripts and Creating Simple Python Programs

Purpose of This Part

In Part 1, you learned the overall workflow:

```
Objective
    ↓
ChatGPT Script
    ↓
Generated Python Code
    ↓
```

```
Run in VS Code
    ↓
Test
    ↓
Improve Script
    ↓
Improve Python Code
```

In Part 2, you will learn how to write better ChatGPT scripts and create simple Python programs.

The most important idea in this part is that Python code should not appear by magic.

You first write a clear script for ChatGPT.

Then ChatGPT generates Python code.

Then you review, test, and improve the code.

Learning Objectives

After completing this part, you will be able to:

- ☐ Write clearer ChatGPT scripts
- ☐ Ask ChatGPT to generate beginner-friendly Python code
- ☐ Ask ChatGPT to include a program header
- ☐ Ask ChatGPT to divide code into steps
- ☐ Use variables
- ☐ Use `input()`
- ☐ Use `print()`
- ☐ Work with text
- ☐ Work with numbers
- ☐ Perform simple calculations
- ☐ Improve prompts after testing
- ☐ Debug simple Python errors with ChatGPT

Course Coding Standard

Starting in this part, every Python program should include a clear header at the top.

The header should include:

- Program name
- Purpose
- User ChatGPT script
- Expected output
- Version
- Notes for the student

The code should also be divided into steps.

Example step labels:

```
# ============================================================
# STEP 1 - Ask for User Input
# ============================================================
```

This makes the program easier to understand, test, and improve.

Standard ChatGPT Script Requirement

Whenever you ask ChatGPT to write Python code, include this section:

```
Coding Standards:

1. Add a detailed program header at the top of the Python file.
2. The header must include:
   - Program Name
   - Purpose
   - User ChatGPT Script
   - Expected Output
   - Version
3. Divide the Python code into clearly labeled steps.
4. Add comments explaining every important line.
5. Make the code suitable for beginners.
6. Explain how to run the code in VS Code.
```

Chapter 1 - Writing Better ChatGPT Scripts

Objective

Learn how to write prompts that generate useful Python code.

A vague prompt gives weak code.

A clear ChatGPT script gives better code.

Folder Structure

Use this folder structure:

```
ChatGPT-Python-Course
   A-Data
   B-Engine
   C-Results
   D-Documentation
```

Start VS Code

1. Open VS Code.
2. Select `File → Open Folder`.
3. Open your main course folder.
4. Open the `B-Engine` folder.
5. Create Python files in `B-Engine`.

Poor ChatGPT Script

```
Write a Python program.
```

This is too vague.

It does not tell ChatGPT:

- What the program should do
- What files or folders are used
- What output is expected
- Whether comments are needed
- Whether the code should be beginner friendly

Better ChatGPT Script

Copy this into ChatGPT:

```
Write a Python program.

Project folder structure:
Top Folder
+- A-Data
+- B-Engine
+- C-Results
+- D-Documentation

Program purpose:
Ask the user for their name and print a welcome message.

Requirements:
1. Use input().
2. Use print().
3. Store the user's name in a variable.
4. Display a friendly welcome message.

Coding Standards:
1. Add a detailed program header at the top of the Python file.
2. The header must include:
   - Program Name
   - Purpose
   - User ChatGPT Script
   - Expected Output
   - Version
3. Divide the Python code into clearly labeled steps.
4. Add comments explaining every important line.
5. Make the code suitable for beginners.
6. Explain how to run the code in VS Code.

Expected output:
The program should display a welcome message using the user's name.
```

ChatGPT Generated Python Code

ChatGPT should generate code similar to this:

```
# ============================================================
# Program Name:
# Welcome-Message.py
#
# Purpose:
# Ask the user for their name and print a welcome message.
#
# User ChatGPT Script:
# Write a Python program.
#
# Program purpose:
# Ask the user for their name and print a welcome message.
#
# Requirements:
# 1. Use input().
# 2. Use print().
# 3. Store the user's name in a variable.
# 4. Display a friendly welcome message.
#
# Expected Output:
# The program displays a welcome message using the user's name.
#
# Version:
# 1.0
# ============================================================

# ============================================================
# STEP 1 - Ask for User Input
# ============================================================

# Ask the user to enter their name.
name = input("Enter your name: ")

# ============================================================
# STEP 2 - Display the Welcome Message
# ============================================================

# Print a blank line to make the output easier to read.
print()

# Display a friendly welcome message.
print(f"Hello {name}. Welcome to Python.")

# ============================================================
# STEP 3 - End Program
# ============================================================

# Print a final message so the user knows the program is complete.
print("Program finished.")
```

Run the Program

Save the program as:

```
B-Engine/Welcome-Message.py
```

In VS Code Terminal, run:

```
python Welcome-Message.py
```

On some Mac computers:

```
python3 Welcome-Message.py
```

Improve the Script

Ask ChatGPT:

```
Please improve the program.

New requirements:
1. Ask for the user's city.
2. Add the city to the welcome message.
3. Keep the same program header format.
4. Keep the step-by-step comments.
```

Chapter Checklist

- ☐ Understand why vague prompts are weak
- ☐ Understand why clear scripts are better
- ☐ Used the coding standard requirement
- ☐ Generated Python code with a program header
- ☐ Generated Python code divided into steps

What You Learned

You learned that a good ChatGPT script should include purpose, requirements, coding standards, and expected output.

Chapter 2 - Variables and User Input

Objective

Create a Python program that asks the user for:

- Name
- Age
- City

and displays the answers in a simple report.

Folder Structure

```
ChatGPT-Python-Course
   A-Data
   B-Engine
   C-Results
   D-Documentation
```

The Python file will be saved as:

```
B-Engine/User-Information.py
```

Start VS Code

1. Open VS Code.
2. Open the main project folder.
3. Open the `B-Engine` folder.
4. Create a file named `User-Information.py`.

ChatGPT Script

Copy this into ChatGPT:

```
Write a Python program.

Project folder structure:
Top Folder
+- A-Data
+- B-Engine
+- C-Results
+- D-Documentation

Program Name:
User-Information.py

Program purpose:
Ask the user for:
1. Name
2. Age
3. City

Requirements:
1. Use input().
2. Store answers in variables.
3. Display all answers in a simple report.
4. Add a blank line before the report.
5. Make the output easy to read.

Coding Standards:
1. Add a detailed program header at the top of the Python file.
2. The header must include:
   - Program Name
```

```
   - Purpose
   - User ChatGPT Script
   - Expected Output
   - Version
3. Divide the Python code into clearly labeled steps.
4. Add comments explaining every important line.
5. Make the code suitable for beginners.
6. Explain how to run the code in VS Code.

Expected output:
A simple report showing the user's name, age, and city.
```

ChatGPT Generated Python Code

```
# ============================================================
# Program Name:
# User-Information.py
#
# Purpose:
# Ask the user for their name, age, and city,
# then display the information in a simple report.
#
# User ChatGPT Script:
# Write a Python program.
#
# Program purpose:
# Ask the user for:
# 1. Name
# 2. Age
# 3. City
#
# Requirements:
# 1. Use input().
# 2. Store answers in variables.
# 3. Display all answers in a simple report.
# 4. Add a blank line before the report.
# 5. Make the output easy to read.
#
# Expected Output:
# A simple report showing the user's name, age, and city.
#
# Version:
# 1.0
# ============================================================

# ============================================================
# STEP 1 - Ask for User Information
# ============================================================

# Ask the user to enter their name.
name = input("Enter your name: ")
```

```
# Ask the user to enter their age.
age = input("Enter your age: ")

# Ask the user to enter their city.
city = input("Enter your city: ")

# ============================================================
# STEP 2 - Display the Report
# ============================================================

# Print a blank line to separate input from output.
print()

# Print the report title.
print("User Information Report")

# Print a line under the title.
print("-----------------------")

# Display the user's name.
print("Name:", name)

# Display the user's age.
print("Age :", age)

# Display the user's city.
print("City:", city)

# ============================================================
# STEP 3 - End Program
# ============================================================

# Print a blank line.
print()

# Let the user know the program is finished.
print("Program finished.")
```

Run the Program

In VS Code Terminal:

```
python User-Information.py
```

On some Mac computers:

```
python3 User-Information.py
```

Example:

```
Enter your name: Yahya
Enter your age: 74
Enter your city: Toronto
```

```
User Information Report
-----------------------
Name: Yahya
Age : 74
City: Toronto

Program finished.
```

Improve the Script

Ask ChatGPT:

```
Please improve the program.

New requirements:
1. Ask for favorite food.
2. Ask for favorite color.
3. Add both values to the report.
4. Keep the program header.
5. Keep the code divided into steps.
6. Add comments to every important line.
```

Exercise

Write a ChatGPT script for a program named:

```
Student-Information.py
```

The program should ask for:

- Student name
- School
- Grade
- Favorite subject

Chapter Checklist

- ☐ Used input()
- ☐ Stored values in variables
- ☐ Displayed values with print()
- ☐ Used the program header standard
- ☐ Used step-by-step code comments
- ☐ Improved the script

What You Learned

You learned how to ask ChatGPT to create a beginner-friendly program using variables and user input.

Chapter 3 - Working with Text

Objective

Create a program that asks for first name and last name, then combines them into a full name.

Python File

Create:

```
B-Engine/Full-Name.py
```

ChatGPT Script

```
Write a Python program.

Project folder structure:
Top Folder
+- A-Data
+- B-Engine
+- C-Results
+- D-Documentation

Program Name:
Full-Name.py

Program purpose:
Ask the user for first name and last name.
Combine them into a full name.
Display the full name.

Requirements:
1. Use input().
2. Store first name in a variable.
3. Store last name in a variable.
4. Combine first name and last name.
5. Display the full name.
6. Use comments for beginners.

Coding Standards:
1. Add a detailed program header at the top of the Python file.
2. The header must include:
   - Program Name
   - Purpose
   - User ChatGPT Script
   - Expected Output
   - Version
3. Divide the Python code into clearly labeled steps.
4. Add comments explaining every important line.
5. Make the code suitable for beginners.
```

```
Expected output:
Display the user's full name.
```

ChatGPT Generated Python Code

```
# ============================================================
# Program Name:
# Full-Name.py
#
# Purpose:
# Ask the user for first name and last name,
# combine them, and display the full name.
#
# User ChatGPT Script:
# Write a Python program.
#
# Program purpose:
# Ask the user for first name and last name.
# Combine them into a full name.
# Display the full name.
#
# Expected Output:
# Display the user's full name.
#
# Version:
# 1.0
# ============================================================

# ============================================================
# STEP 1 - Ask for Text Input
# ============================================================

# Ask the user to enter their first name.
first_name = input("Enter your first name: ")

# Ask the user to enter their last name.
last_name = input("Enter your last name: ")

# ============================================================
# STEP 2 - Combine Text Values
# ============================================================

# Combine the first name, a space, and the last name.
full_name = first_name + " " + last_name

# ============================================================
# STEP 3 - Display the Result
# ============================================================

# Print a blank line for readability.
print()
```

```
# Display the full name.
print("Full Name:", full_name)

# ============================================================
# STEP 4 - End Program
# ============================================================

# Print completion message.
print("Program finished.")
```

Improve the Script

Ask ChatGPT:

```
Please improve the program.

New requirements:
1. Ask for country.
2. Ask for favorite language.
3. Display a friendly sentence using all values.
4. Keep the header and step comments.
```

What You Learned

You learned how text values can be stored, combined, and displayed.

Chapter 4 - Working with Numbers

Objective

Create a program that asks for two numbers and displays them.

This chapter introduces numeric input.

Python File

Create:

```
B-Engine/Two-Numbers.py
```

ChatGPT Script

```
Write a Python program.

Program Name:
Two-Numbers.py
```

```
Program purpose:
Ask the user for two numbers and display them.

Requirements:
1. Ask for first number.
2. Ask for second number.
3. Convert both inputs to numbers using float().
4. Display both numbers.
5. Add comments for beginners.

Coding Standards:
1. Add a detailed program header at the top of the Python file.
2. The header must include Program Name, Purpose, User ChatGPT Script, Expected Output, and Version.
3. Divide the code into clearly labeled steps.
4. Add comments explaining every important line.
5. Make the code suitable for beginners.

Expected output:
Display the two numbers entered by the user.
```

ChatGPT Generated Python Code

```
# ============================================================
# Program Name:
# Two-Numbers.py
#
# Purpose:
# Ask the user for two numbers and display them.
#
# User ChatGPT Script:
# Write a Python program that asks the user for two numbers,
# converts them using float(), and displays both numbers.
#
# Expected Output:
# Display the two numbers entered by the user.
#
# Version:
# 1.0
# ============================================================

# ============================================================
# STEP 1 - Ask for Numeric Input
# ============================================================

# Ask for the first number and convert it to a decimal number.
number1 = float(input("Enter first number: "))

# Ask for the second number and convert it to a decimal number.
number2 = float(input("Enter second number: "))

# ============================================================
```

```
# STEP 2 - Display the Numbers
# ============================================================

# Print a blank line.
print()

# Display the first number.
print("Number 1 =", number1)

# Display the second number.
print("Number 2 =", number2)

# ============================================================
# STEP 3 - End Program
# ============================================================

# Print completion message.
print("Program finished.")
```

Important Note

The function `input()` always returns text.

To do math, convert the text to a number.

Use:

```
float(input("Enter number: "))
```

Improve the Script

Ask ChatGPT:

```
Please improve the program.

New requirements:
1. Add the two numbers.
2. Subtract the second number from the first number.
3. Multiply the two numbers.
4. Divide the first number by the second number.
5. Add a warning if the second number is zero.
6. Keep the program header and step comments.
```

What You Learned

You learned how to ask ChatGPT to generate code that uses numeric input.

Chapter 5 - Simple Calculations

Objective

Create a program that calculates the area and perimeter of a rectangle.

Python File

Create:

```
B-Engine/Rectangle-Calculator.py
```

ChatGPT Script

```
Write a Python program.

Program Name:
Rectangle-Calculator.py

Program purpose:
Calculate the area and perimeter of a rectangle.

Requirements:
1. Ask for length.
2. Ask for width.
3. Convert both inputs to numbers using float().
4. Calculate area.
5. Calculate perimeter.
6. Display results clearly.
7. Add comments for beginners.

Coding Standards:
1. Add a detailed program header at the top of the Python file.
2. The header must include Program Name, Purpose, User ChatGPT Script, Expected Output, and Version.
3. Divide the code into clearly labeled steps.
4. Add comments explaining every important line.
5. Make the code suitable for beginners.

Expected output:
Display rectangle length, width, area, and perimeter.
```

ChatGPT Generated Python Code

```
# ============================================================
# Program Name:
# Rectangle-Calculator.py
#
# Purpose:
# Calculate the area and perimeter of a rectangle.
```

```
#
# User ChatGPT Script:
# Write a Python program that asks for length and width,
# calculates area and perimeter, and displays the results.
#
# Expected Output:
# Display rectangle length, width, area, and perimeter.
#
# Version:
# 1.0
# ============================================================

# ============================================================
# STEP 1 - Ask for Rectangle Dimensions
# ============================================================

# Ask the user for the rectangle length.
length = float(input("Enter rectangle length: "))

# Ask the user for the rectangle width.
width = float(input("Enter rectangle width: "))

# ============================================================
# STEP 2 - Perform Calculations
# ============================================================

# Calculate rectangle area.
area = length * width

# Calculate rectangle perimeter.
perimeter = 2 * (length + width)

# ============================================================
# STEP 3 - Display Results
# ============================================================

# Print a blank line.
print()

# Print report title.
print("Rectangle Report")
print("----------------")

# Display input values.
print("Length   :", length)
print("Width    :", width)

# Display calculated values.
print("Area     :", area)
print("Perimeter:", perimeter)

# ============================================================
# STEP 4 - End Program
```

```
# ============================================================

# Print completion message.
print("Program finished.")
```

Run the Program

```
python Rectangle-Calculator.py
```

or:

```
python3 Rectangle-Calculator.py
```

Improve the Script

Ask ChatGPT:

```
Please improve the program.

New requirements:
1. Ask for price per square foot.
2. Calculate total cost.
3. Display area, perimeter, and total cost.
4. Format the cost with two decimal places.
5. Keep the program header and step comments.
```

Exercise

Write a ChatGPT script for a program that calculates the area of a circle.

Requirements:

- Ask for radius
- Calculate area
- Use 3.14159 for pi
- Display result
- Include header and step comments

What You Learned

You learned how to create simple calculation programs with ChatGPT.

Chapter 6 - Improving ChatGPT Scripts

Objective

Learn how to improve your ChatGPT script after testing the program.

Weak Script

```
Write a calculator.
```

This is not clear enough.

Strong Script

```
Write a Python program.

Program Name:
Basic-Calculator.py

Program purpose:
Ask the user for two numbers and perform addition, subtraction, multiplication, and division.

Requirements:
1. Ask for first number.
2. Ask for second number.
3. Convert both values using float().
4. Add the numbers.
5. Subtract the numbers.
6. Multiply the numbers.
7. Divide the numbers.
8. If the second number is zero, do not divide.
9. Display all results clearly.

Coding Standards:
1. Add a detailed program header.
2. Include the user ChatGPT script in the header.
3. Divide the code into labeled steps.
4. Add comments for beginners.
```

Key Lesson

A stronger script includes:

- Program name
- Purpose
- Requirements
- Error handling
- Expected output
- Coding standards

Exercise

Rewrite this weak script:

```
Make a program for age.
```

into a strong ChatGPT script.

What You Learned

You learned how better scripts create better Python code.

Chapter 7 - Debugging with ChatGPT

Objective

Learn how to ask ChatGPT to help when your Python program has an error.

Debugging Script

Use this script when your program fails:

```
My Python program has an error.

Please help me debug it.

Explain:
1. What the error means.
2. Which line caused it.
3. How to fix it.
4. How to avoid it next time.

Here is my code:

[paste code here]

Here is the full error message:

[paste error message here]
```

Example Error Program

```
# ============================================================
# Program Name:
# Error-Demo.py
#
# Purpose:
# Demonstrate a simple variable name error.
#
# User ChatGPT Script:
# Create a short program that asks for a name and prints it.
#
```

```
# Expected Output:
# Display the entered name.
#
# Version:
# 1.0
# ============================================================

# ============================================================
# STEP 1 - Ask for Name
# ============================================================

# Ask the user for their name.
name = input("Enter your name: ")

# ============================================================
# STEP 2 - Display Name
# ============================================================

# This line has an intentional mistake.
# The variable should be name, not nam.
print(nam)
```

Corrected Code

```
# Ask the user for their name.
name = input("Enter your name: ")

# Correct variable name.
print(name)
```

Exercise

Create a small mistake and ask ChatGPT to fix it.

Examples:

- Misspell a variable
- Remove a quote
- Remove a parenthesis
- Forget to convert input to float()

What You Learned

You learned how to use ChatGPT as a debugging assistant.

Chapter 8 - Part 2 Review

Main Workflow

```
Objective
    ↓
ChatGPT Script
    ↓
Python Header
    ↓
Step-by-Step Python Code
    ↓
Run in VS Code
    ↓
Test
    ↓
Improve Script
```

Part 2 Skills Checklist

By the end of Part 2, you should be able to:

- ☐ Write better ChatGPT scripts
- ☐ Ask for a program header
- ☐ Ask for step-by-step code sections
- ☐ Use variables
- ☐ Use input()
- ☐ Use print()
- ☐ Work with text
- ☐ Work with numbers
- ☐ Perform simple calculations
- ☐ Improve prompts
- ☐ Debug programs with ChatGPT

Summary

In Part 2, you learned that the quality of your ChatGPT script directly affects the quality of the generated Python code.

A good script includes:

- Objective
- Folder structure
- Program name
- Requirements
- Coding standards
- Expected output

The new standard for this course is:

```
ChatGPT Script
        ↓
Program Header
```

```
        ↓
Step-by-Step Python Code
        ↓
Testing
        ↓
Improvement
```

Looking Ahead

In Part 3, you will use the same method to create Python programs that work with files and folders.

You will learn how to ask ChatGPT to generate programs that can:

- Read text files
- Write text files
- Copy files
- Scan folders
- Save logs