

Chapter 2 Study Guide

Input, Processing, and Output

1. Program Development Cycle (5 phases)

The program development cycle typically has five phases: **Design the program**, **Write the code**, **Correct syntax errors**, **Test the program**, and **Correct logic errors**. Good design prevents costly rework later.

Fill in: The five phases include Design, Write, Correct _____ errors, Test, and Correct _____ errors. / .

2. Design: Algorithms, Pseudocode, and Flowcharts

An **algorithm** is a set of well-defined steps to solve a task. Programmers often write **pseudocode** (logic) or draw **flowcharts** (diagram symbols: ovals, parallelograms, rectangles) before coding.

Fill in: Pseudocode is not executed; it helps focus on the program's .

3. Input – Processing – Output

Most programs follow this three-step model: obtain *input*, perform *processing* (calculations, conversions), and produce *output* (display or store results).

Fill in: The three parts of a program's basic model are , , and .

4. Displaying Output — print()

Use Python's `print()` function to display text and values. Strings are enclosed in quotes. You can pass multiple items to `print` (they are separated by spaces by default).

Fill in: To display values separated by spaces, pass them to `print` separated by .

5. Comments

Comments begin with `#` and are ignored by Python. They explain code for human readers and should be used liberally.

Fill in: In Python, a single-line comment begins with the character .

6. Variables and Assignment

A **variable** is a name that references a value stored in memory. Use assignment (e.g., `hours = 40`) to create/update variables. Variable names follow specific rules (start with letter or underscore, no spaces, case-sensitive).

Fill in: The operator used to assign a value to a variable is the operator.

7. Reading Input — input()

`input(prompt)` reads text from the keyboard and always returns a `str`. We can convert the data returned from the input function to numbers with `int()` or `float()` when needed e.g., `age = int(input('Age: '))`. Notice the TWO parenthesis at the end.

Fill in: `input()` returns a value of type .

8. Performing Calculations

Python supports standard arithmetic operators: +, -, *, / (float division), // (integer division), % (remainder), and ** (exponent).

Fill in: Use for floating-point division and for integer division.

9. String Concatenation (important)

Concatenation appends one string to another. In Python use the + operator for strings: 'Hello ' + 'world' -> 'Hello world'. Implicit concatenation occurs when adjacent string literals are written next to each other across lines.

Example: building a full name — `full_name = first_name + ' ' + last_name`.

Fill in: The operator used to concatenate strings is .

10. F-strings and Formatted Output

F-strings (prefix with f) let you embed expressions inside strings: `f'Hello {name}'`. Placeholders can include expressions and format specifiers for rounding and alignment.

Fill in: An f-string is prefixed with the letter .

11. Named Constants & Types

Use named constants (by convention UPPER_CASE) for values that don't change. Know basic types: `int`, `float`, and `str`. Mixed-type expressions convert ints to floats automatically when needed.

Fill in: Integers are type and decimals are .

Quick Review — Fill in the Blanks

1. A byte contains bits.

2. The Program Development Cycle stage after writing code is correcting _____ errors. **Syntax**
3. `input()` returns a **String** value — convert with **Int** or **Float**.
4. The concatenation operator for strings is **+**.
5. Use **f** before a string to create an f-string.
6. The remainder operator is **%**.
7. To perform exponentiation use ******.
8. To join multiple items in one print statement, separate them with **Commas**.
9. In a flowchart, a rectangle denotes **Processing**.
10. Python variable names are case-**Sensitive**.
11. A **Logic** error does not prevent the program from running, but causes it to produce incorrect results.
12. A **Software requirement** is a single function that the program must perform in order to satisfy the customer.
13. A(n) **Algorithm** is a set of well-defined logical steps that must be taken to perform a task.
14. An informal language that has no syntax rules and is not meant to be compiled or executed is called **Pseudocode**.
15. A **Flowchart** is a diagram that graphically depicts the steps that take place in a program.
16. A **String** is a sequence of characters.
17. A **Variable** is a name that references a value in the computer's memory.
18. A **User** is any hypothetical person using a program and providing input for it.
19. A string literal in Python must be enclosed in either single-quotes or **Double-quotes**.
20. Short notes placed in different parts of a program explaining how those parts of the program work are called **Comments**.
21. A(n) **Variable** makes a variable reference a value in the computer's memory.
22. This symbol marks the beginning of a comment in Python.

#

23. Which of the following statements will cause an error?

17 = x

24. In the expression 12 + 7, the values on the right and left of the + symbol are called **Operands**.

25. This operator performs integer division. //

26. This is an operator that raises a number to a power.

**

27. This operator performs division, but instead of returning the quotient it returns the remainder. %

28. Suppose the following statement is in a program: price = 99.0. After this statement executes, the price variable will reference a value of which data type? **float**

29. Which built-in function can be used to read input that has been typed on the keyboard? **Input()**

30. Which built-in function can be used to convert an int value to a float? **float()**

31. A magic number is an **Single digit** value that appears in a program's code.

32. A **Named Constant** is a name that represents a value that does not change during the program's execution.