Python Practice Problems (July – Oct. 2025)

Below is a comprehensive list of Python practice problems designed to cover the same breadth and progression as the provided C programming practice file, while adapting to Python's syntax and features. This list is ideal for beginners through intermediate learners.

Section 1: Print & Read

This section covers the fundamentals of input/output in Python, variable operations, data types, assignments, and basic arithmetic.

Basic Input/Output and Variable Handling

1. Print "Hello world..."
2. Print an integer constant.
3. Read an integer from the user and print it.
4. Read two integers from the user and display both.
5. Read two integers, to perform:
    - a. Addition, subtraction, and multiplication without using a third variable
    - b. Addition, subtraction, and multiplication using a third variable
6. Swap two numbers using a third variable.
7. Swap two numbers without using a third variable.
8. Get and display the size of `int`, `float`, `complex`, and `str` using `sys.getsizeof()`.
9. Assign a character constant to a variable and print it.
10. Read a character from the user, assign to a variable, and print it.
11. Read the string "Hello World" using `input()` and print it. *(Explore why input() behaves differently from C's scanf.)*

Arithmetic and Expression Evaluation

12. Accept an integer and compute:
    - a. $ x^2 $

- b. $ x^2 + 2x $

13. Accept two integers x, y and compute:
    - a. $ x^3 + 3x^2 + 4x - y^3 $
    - b. $ 2x^2 + 4y^2 + x^3 + 10 $
    - c. $ 4x^2 + 8y^2 + x^3 + 5 / (2x^2) $
14. Find the roots of a quadratic equation given coefficients a, b, and c using the `cmath` module.
15. Check what happens if non-numeric data is entered when an integer or float is expected (input validation using `try-except`).

Section 2: Conditional Statements

This section provides practice programs focusing on conditional statements, using `if`, `elif`, `else`, and Python's ternary operator.

2.1 Simple If, If...Else, and Elif

1. Read an integer and determine whether it is positive, negative, or zero.
2. Read two integers and check if they are equal, or determine which is larger.
3. Read three integers and find the largest among them.
4. Check if a number is odd or even.
5. Determine if a given year is a leap year.
6. Check if a user-input character is an alphabet letter.
7. Check if a character is a vowel or consonant.
8. Check if a character is uppercase or lowercase.
9. Check if the input character is an alphabet, digit, or special character.

2.2 Nested If...Else and Logical Operators

10. Read marks for five subjects and assign a grade based on percentage ranges (A/B/C/D/E/F).
11. Compute profit or loss given cost price and selling price; display the result.

12. Read three angles of a triangle and check validity:
    - a. Each angle must be greater than 0°
    - b. The sum must be exactly 180°
13. Read three positive numbers (sides of a triangle). Check validity and categorize (equilateral, isosceles, scalene).
14. Input week number (1–7) and print the corresponding weekday name.
15. Input month number and display the number of days in the month (consider leap years for February).

2.3 Ladder and Multiple Conditions

16. Print a message based on these ranges:
    - Number <100: "small"
    - 100–200: "large"
    - 201–300: "bigger"
    - 301–400: "largest"
    - 400: "very large"

2.4 Conditional (Ternary) Operator

17. Input two numbers, find the maximum using a conditional expression.
18. Use a conditional expression to check if a number is even or odd.
19. Assign remarks ("Pass"/"Fail") using conditional expressions based on marks.
20. Use nested conditional expressions to classify a number as positive/negative/zero.

2.5 Switch Statements

Note: Python does not have a switch statement; use if-elif-else instead.

21. Read a character (`+`, `-`, `*`, `/`) and two integers, perform the corresponding operation.
22. Take a number (1–7) and print the day of the week using if-elif-else.
23. Input a number (1–12) and display the name of the month via if-elif-else.

24. Read a grade character and print remarks using if-elif-else.
25. Given a number, output "small" if <10, "large" if >10, "equal" if exactly 10.

Section 3: Loops and Iteration

This section builds mastery of loops and iterative constructs in Python.

3.1 While Loops

1. Print all numbers from 1 to n (user input)
2. Print all even numbers between 1 and n
3. Print the sum of all integers from 1 to n
4. Print the sum of all odd numbers from 1 to n
5. Calculate and print the sum of $ 1 + 1/2 + 1/3 + ... + 1/n $
6. Sum of squares of first n numbers
7. Print the multiplication table for a given number
8. Reverse a number entered by the user
9. Count and display the number of digits in a number
10. Calculate the factorial of a number
11. Print the first n terms of the Fibonacci series
12. Check whether a given number is prime or composite
13. Find the power ($ x^y $) without using `pow()`
14. Display all uppercase ASCII characters with their integer values
15. Print a square/triangle/star pattern based on user input (height)
16. Check if a number is a palindrome
17. Find the GCD (HCF) and LCM of two numbers
18. Implement all above programs using for loop

3.2 Do-While Loops

Note: Python does not have do-while; use while True and break.

1. Read numbers until the user enters 0; compute and print their sum.
2. Ask the user for a password until correct.
3. Menu-driven calculator: repeat operations until user chooses to exit.
4. Display digits of a number, one per line.

3.3 Nested Loops

1. Print multiplication tables from 2 to 11 using nested for loops.
2. Find and display all prime numbers between 1 and 1,000,000.
3. Generate and print the prime-factored form of numbers 1 to n.
4. Check and print all Armstrong numbers in a range using nested loops.
5. Determine whether two numbers are amicable.
6. Check whether a number is perfect.
7. Print various formatted patterns (pyramid, diamond, Pascal's triangle).

3.4 Use of Break and Continue

1. Skip printing odd numbers in a loop using continue.
2. Keep summing until the user enters a negative number, then break the loop.
3. Find the smallest divisor of a number greater than 1 using break.
4. Find the first number greater than n that is divisible by 7.
5. Print numbers from 1 to N, skipping numbers divisible by 3.
6. Allow password input at most 3 times. If correct, print "Access granted," else "Account locked."
7. Print numbers from 1 to N, skipping those ending in 5.
8. Print the smallest number ≤N divisible by both 4 and 6.