50 Python Problem’s (Basic to Advanced):

Basic Python Syntax (1-10)

1. Write a Python program to print "Learning Python for Data Engineering!".
2. Create variables of type `int`, `float`, and `string`, assign them values, and print their types using `type ()`.
3. Write a program to swap two numbers using a temporary variable.
4. Create a list of the first 10 even numbers and print it.
5. Write a program to calculate the sum of the first 50 integers.
6. Create a dictionary with keys as column names (`name`, `age`, `salary`) and values as lists. Print the dictionary.
7. Use a set to store unique column names from a dataset (`name`, `age`, `name`, `department`).
8. Write a Python program to calculate the square root of a number using `math.sqrt()`.
9. Create a variable to store a string and find its length using `len()`.
10. Write a program to reverse a string and print both the original and reversed string.

Variables, Data Types, and Operations (11-20)

11. Write a program to check if a given number is positive, negative, or zero.
12. Create a variable to store a list of names and print the names in uppercase using a loop.
13. Write a function to convert a temperature from Celsius to Fahrenheit.
14. Create a dictionary with student names as keys and their scores as values. Write a program to print the student with the highest score.
15. Write a Python program to merge two dictionaries into one.
16. Write a Python program to find the union and intersection of two sets.
17. Create a program to generate the first `n` Fibonacci numbers using a list.
18. Write a Python program to check if a string is a palindrome (e.g., `madam`).
19. Create a program that calculates the factorial of a number using a `for` loop.
20. Write a program to calculate the sum of all values in a dictionary.

Conditionals and Loops (21-30)

21. Write a program to check if a number is even or odd using an `if-else` statement.
22. Create a loop that prints all integers between 1 and 100 that are divisible by 7.

23. Write a program that prints "Data Engineering" for multiples of 3, "Python" for multiples of 5, and "Data Engineering Python" for multiples of both.
24. Write a Python program to find the largest and smallest numbers in a list using loops.
25. Create a nested `for` loop to print a multiplication table up to 10.
26. Write a program that counts the number of vowels and consonants in a string using loops and conditionals.
27. Implement a program to check if a given year is a leap year.
28. Write a program that sums all odd numbers between 1 and 100 using a `while` loop.
29. Write a Python program to iterate through a dictionary and print the key-value pairs.
30. Use a loop to print the first 10 powers of 2 (e.g., `2^1, 2^2, ..., 2^10`).

Intermediate Python for Data Engineering (31-40)

31. Create a program to find the second largest number in a list without using built-in functions.
32. Write a function that takes a list of numbers and returns a new list with only the even numbers.
33. Write a program to flatten a nested list (e.g., `[[1, 2], [3, 4]]` to `[1, 2, 3, 4]`).
34. Create a dictionary from two lists: one containing column names and another containing data values.
35. Write a function to check if two strings are anagrams (e.g., `listen` and `silent`).
36. Use a `for` loop to generate the transpose of a 2D list (matrix).
37. Write a program to sort a dictionary by its values in ascending order.
38. Create a program to find the most frequent element in a list.
39. Write a Python function to count the occurrences of each word in a string.
40. Write a program to generate a list of prime numbers between 1 and 100 using loops and conditionals.

Advanced Python for Data Engineering (41-50)

41. Write a Python function to read a dataset (CSV file) and calculate the mean of a specific column.
42. Use list comprehensions to create a list of squares of all numbers from 1 to 50.
43. Write a Python program to find duplicate rows in a dataset loaded as a list of dictionaries.
44. Create a function that calculates the correlation coefficient between two lists of numbers.
45. Use Python to parse a JSON file and extract specific fields (e.g., `name`, `age`) into a list of dictionaries.
46. Write a Python program to group data by a specific column and calculate aggregate statistics (e.g., sum, average).
47. Write a Python program to find missing values in a list and replace them with the column mean.
48. Use `try-except` blocks to handle missing files or data parsing errors in a Python script.

49. Create a program to load two datasets as dictionaries and join them on a common key (like an SQL `JOIN` operation).
50. Write a Python program to implement a sliding window algorithm to calculate the moving average of a dataset.

These problems guide you from the foundational aspects of Python (syntax, data types, loops, conditionals) to **data engineering-focused applications**, preparing you for tasks like **data manipulation**, **ETL operations**, and **data processing** in Python.

Start solving these problems incrementally, and feel free to ask for hints or explanations!