Week 6: Algorithm Design Guidance

Before writing any Python code, plan your approach to each task. Designing algorithms helps you think logically, avoid errors, and build structured solutions.

Step 1: Understand the Problem

Identify the goal of the program or function, what inputs it needs, and what outputs it should produce.

Step 2: Identify Key Data Structures

Choose appropriate Python data types such as lists, dictionaries, tuples, and strings. For example, a contact can be represented as a dictionary with keys like 'name', 'phone', and 'email'.

Step 3: Outline the Algorithm (Pseudocode)

Write each step in plain English before coding. Describe what happens, not how to code it.

Step 4: Consider Edge Cases

Think about what might go wrong—empty inputs, invalid data, duplicates—and plan to handle them.

Step 5: Write and Test Each Function

Translate your pseudocode into Python and test it using small, simple examples. Use print statements to verify results.

Step 6: Combine and Refine

Once individual functions work, combine them into a complete program. Organize your code with functions or a main() block.

Recommended Exercise Flow

Task A: Contact List — Focus: Lists & Dictionaries. Draw how contacts will be stored.
Task B: Survey Analyzer — Focus: Loops & Input. Write pseudocode for collecting and analyzing responses.
Task C: Module & Report — Focus: Imports & File Handling. Sketch how modules interact with main script.