## Scenario-Based Practice Questions for Go Fundamentals

### 1. Variables & Data Types

**Scenario:**  
You’re building a simple inventory app for a bookstore. Each book has a title (string), number of copies in stock (int), price (float64), and availability status (bool).  
Write code to declare variables for these properties, assign values, and print a formatted summary.

### 2. Conditional Statements

**Scenario:**  
You’re writing a grading program. The user inputs a percentage score (0-100).

* If score ≥ 90, print “Grade: A”
* If score ≥ 80, print “Grade: B”
* If score ≥ 70, print “Grade: C”
* Else print “Grade: F”  
  Also, print “Pass” if score ≥ 70, otherwise “Fail”.

### 3. Loops

**Scenario:**  
You need to generate a multiplication table for a given number input by the user (1-10). Use loops to print the table in the format:  
5 x 1 = 5  
5 x 2 = 10  
… and so on.

### 4. Functions

**Scenario:**  
Create a calculator program that supports addition, subtraction, multiplication, and division.

* Implement separate functions for each operation.
* Take user input for two numbers and the operation to perform.
* Call the right function and display the result.
* Handle division by zero error.

### 5. Structs

**Scenario:**  
You’re designing a system to manage employee records.  
Create an Employee struct with fields: ID (int), Name (string), Position (string), and Salary (float64).  
Write a function to give a 10% raise to an employee’s salary.  
Create an employee, apply the raise, and print updated details.

### 6. Interfaces

**Scenario:**  
You’re developing a drawing app with different shapes: rectangles, circles, and triangles.

* Define an interface Shape with methods Area() and Perimeter().
* Implement these for each shape struct with user-provided dimensions.
* Write a function that takes any Shape and prints its details.

### 7. Slices

**Scenario:**  
You’re maintaining a dynamic to-do list application.

* Create a slice of strings representing tasks.
* Add new tasks to the list (simulate user input).
* Remove completed tasks by index.
* Print the current list after every change.

### 8. Maps

**Scenario:**  
Build a program that keeps track of country populations.

* Use a map with country names as keys and populations as values (int).
* Add at least 5 countries with data.
* Allow the user to input a country name and display its population or an error if not found.
* Update the population of a country and show the new data.

### 9. Error Handling

**Scenario:**  
Create a function to parse user input string into an integer.

* Return an error if the input is not a valid integer.
* In main(), prompt the user to enter a number, parse it, and print the number or error message.