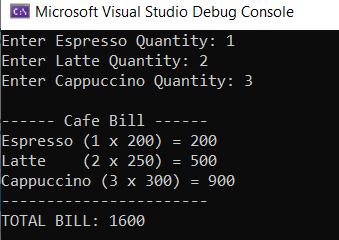
## **Problem Statement: “Pointer Café – Coffee Billing System”**

You are working at a small café that sells 3 types of coffee: **Espresso**, **Latte**, and **Cappuccino**.You need to write a C++ program that calculates the total bill but with a twist: **you must use pointers to access and modify variables.**

### Requirements

1. **Use Pointers for All Variables**
   * Create three float variables for the price of Espresso, Latte, and Cappuccino.
   * Create three int variables for the quantity ordered.
   * Create pointers for each of these variables and use the pointers to:
     + Take input from the user for quantity.
     + Calculate cost for each coffee type.
2. **Calculate Total**
   * Use a separate pointer for totalBill and store the final result using only pointers.
3. **Display Output**
   * Print a small bill showing quantity, price, and subtotal for each item — **accessing variables via pointers only**.



Problem Statement: **“Secret Code Machine”**

### Scenario

You are building a **Secret Code Machine** for a spy agency.  
The machine takes an input number from the agent and must **generate a secret code**. The agency says:

“***The code must be computed by multiplying all numbers from the input number down to 1.”***

They want the program to be flexible, so instead of calling the function directly, the machine must store the algorithm in a **function pointer** and call it dynamically.

### Requirements

1. **Write a Function**  
   Create a function (e.g., generateCode(int n)) that:
   * Multiplies numbers from n down to 1.
   * Returns the final product as the **secret code**.
2. **Function Pointer**
   * Create a function pointer:
   * int (\*codePtr)(int);
   * Assign it to generateCode.
3. **User Input & Call via Pointer**
   * Ask the agent to enter a number.
   * Call the function **only through the pointer**, not directly.
   * Print the result as:  
     Your secret code is: <value>

### Bonus Twist

* If the user enters 0, print Your secret code is: 1 (force them to handle base case properly).

Problem Statement

### **Scenario: “Custom Score Tracker”**

You are developing a **sports score tracker** for a local tournament.  
The program should:

1. Ask the user how many players are participating.
2. Allocate memory dynamically to store each player’s score.
3. Take input for all scores.
4. Display the scores back to the user.
5. At the end, release the memory back to the system.

**Hint to Students:**  
You do **not** know how many players will participate until runtime, so you must allocate memory **after asking the user**.

Problem Statement

### **Scenario: “Clean Attendance Register”**

You are designing an **attendance register** for a workshop.  
The program should:

1. Ask the user for the number of participants.
2. Dynamically allocate memory for their attendance status (1 = present, 0 = absent).
3. Initially, all participants should be marked **absent by default**.
4. Allow the user to mark selected participants as present.
5. Display the final attendance list at the end.

**Hint to Students:**  
The attendance must be **clean (all zeros)** before marking anyone present.