## Question 1: AI as a Programming Partner (7 minutes)

### Part 1: Two Ways to Use AI in "Understand" and "Identify Input/Output"

**1. During the “Understand” phase:**

“I need to build a temperature converter that handles user errors. Can you help me understand the key features and logic I should consider before coding?”

Answer: Absolutely! Building a temperature converter that handles user input errors requires both functionality and robustness. Here are the key features and logic to consider:

◆ **1. User Input for Temperature Value**

- **Expected Input:** A numeric value (e.g., 100, -40, 36.6)
- **Possible Errors to Handle:**
  - User enters a word or symbol (e.g., "hot", "**")
  - User leaves the input blank
- **Solution:** Use a try block with float() conversion and a loop to re-prompt if input is invalid.

◆ **2. Input for Original Unit**

- **Expected Input:** 'C' for Celsius or 'F' for Fahrenheit
- **Possible Errors:**
  - User enters lowercase, wrong letters (e.g., celsius, Z, Cel)
- **Solution:** Use .strip().upper() to normalize input and validate against allowed values ('C' or 'F')

◆ **3. Input for Target Unit**

- **Expected Input:** Same as above ('C' or 'F')
- **Logic Needed:**
  - Prevent conversion from 'C' to 'C' (same-unit)
  - Alert user and ask for a different target

◆ **4. Conversion Formulas**

- **F = C × 9/5 + 32**

- **C = (F − 32) × 5/9**

◆ **5. Output Formatting**

- Display result clearly (e.g., 100°C = 212°F)
- Round to 2 decimal places
- Include units in output

◆ **6. Edge Cases**

- Large numbers (e.g., 1e6)
- Negative temperatures (valid, especially for Celsius)
- Same unit entered twice

## 2. During the "Identify Input/Output" phase:

"For a quiz program that asks multiple-choice questions, what should the input and output look like from a user and code perspective?"

Answer: **AI Response (what a good assistant would say):**

Let's break this down into **user-level** and **code-level** inputs and outputs:

USER PERSPECTIVE

◆ **User Inputs:**

1. Select an answer choice for each question (e.g., A, B, C, or D)
2. Possibly input their name or start the quiz manually

◆ **User Outputs:**

1. See each question with options
2. Immediate feedback (optional): "Correct!" / "Wrong, the right answer is B."
3. Final score: percentage (e.g., You got 3 out of 4 correct. Score: 75%)
4. Optional: Retry prompt or review correct answers

## CODE PERSPECTIVE

- **Code Inputs:**
  - A **data structure** containing questions, options, and correct answers
    Example:

```
questions = [
  {
     "question": "What data type is '42' in Python?",
     "options": {"A": "int", "B": "str", "C": "float", "D": "bool"},
     "answer": "B"
  },
  ...
]
```

  - **User response** for each question (validated: only A, B, C, or D allowed)
- **Code Outputs:**
  - Printed/displayed question and answer options
  - Track whether user answer was correct
  - Tally total score
  - Output final result, such as:

```
Question 1: Correct!
Question 2: Incorrect. Correct answer: C
Final Score: 2/3 (66.67%)
```

**Error Handling to Consider:**

- What if the user types e, 1, or a blank input?
- Solution: Re-prompt user until valid option (A-D) is entered

## Question:  Why Use AI Early On & Potential Pitfalls

1. **Why It's Beneficial:**

- Clarifies problem requirements
- Helps uncover edge cases early
- Saves time by avoiding confusion later
- Encourages planning before coding

2. **Potential Pitfalls:**

- Accepting AI answers without understanding
- Over-relying on AI to "think" for you
- Getting generic advice if prompts are vague

# Question2

**Part1:**

**a.) This function works fine unless the input list is empty. The problem arises here:  average = total / len(numbers)**

**Reason: If numbers is an empty list ([]), then len(numbers) is 0, and you'll get a ZeroDivisionError when the code tries to divide by zero.**

**b.) What Will Happen When It Runs:**

- **The first list [10, 20, 30, 40, 50] will print an average correctly.**
- **The second list [] will cause the program to crash with:**

**ZeroDivisionError: division by zero**

- **Because of the crash, the remaining test cases ([0] and [-5, 5, -10, 10]) will not be executed.**

Part2: **AI Debugging Prompt**

I have a Python function that calculates the average of a list of numbers. It works for non-empty lists, but crashes when the list is empty. Can you help me understand why it crashes and how I can improve the code to handle empty lists carefully? Also, is there a way to return something meaningful like None or a message if the list has no values?

Part3: Fixed code

```
def calculate_average(numbers):
    if not numbers:
        return "Cannot calculate average of an empty list."
    total = 0
    for number in numbers:
        total += number
    average = total / len(numbers)
    return average


# Test cases
test_data = [
    [10, 20, 30, 40, 50],
    [],  # Empty list
    [0],
    [-5, 5, -10, 10]
]

for data in test_data:
    result = calculate_average(data)
    print(f"Average of {data}: {result}")
```