**1. Introduction to Functions (2 minutes)**

**Talking Points**:

• “Functions allow us to group a set of instructions under a name. We can call that name anytime to run the instructions without repeating code.”

• “Using functions makes code modular and reusable, meaning we can organize our code better, especially in bigger projects.”

**Key Concepts**:

• **Defining a Function**: Explain how to define a function using def.

• **Calling a Function**: Show how to execute or “call” the function once it’s defined.

**Example**:

A black background with green text

Description automatically generated**2. Demonstrating Custom Functions (10 minutes)**

**Step 1: Functions with Arguments (4 minutes)**

**Talking Points**:

• “Functions can also accept inputs called arguments. These allow us to pass in values and customize what the function does.”

• “Arguments make functions flexible because they can do different things depending on the values passed to them.”

**Example**:

A black background with green text

Description automatically generated

**Explanation**:

• "name" is an argument. When you call the function, you can pass any name to it.

• Call the function with an argument:

• Open the floor for questions: “Do you have any questions about how to use functions?”

• Encourage students to share how they would use functions in their projects.

• “Where do you think using a function would help make your code cleaner or more efficient?”

• Discuss real-world applications where functions play a key role, like calculators, web apps, and games.