**Imperative:** An imperative language is like giving step-by-step instructions to a computer. You tell the computer exactly what to do and how to do it. It's like writing a recipe where you list all the ingredients (data) and then give clear directions (commands) on how to cook (process) them.

Eg. Azure CLI and Azure PowerShell scripts.

Key Points:

* **Instructions:** You directly tell the computer what actions to take and in what order.
* **State Changes:** You can change the values of variables as the program runs.
* **Control Flow:** You use loops, conditionals, and function calls to control how the instructions are executed.
* **Example:** In a program to find the largest number in a list, you'd write steps like "start with the first number, compare it with the next, and if it's bigger, remember it as the largest."

Example (Finding the Largest Number):

Imagine you have a list of numbers: [3, 7, 1, 9, 4, 2, 6]. To find the largest number using an imperative approach, you'd:

1. Start with the first number (3).
2. Compare it with the next number (7). Since 7 is larger, you remember it.
3. Continue comparing each number with the remembered largest number, updating it whenever you find a larger one.

Why Use Imperative Languages?

* **Control:** You have precise control over how things happen.
* **Efficiency:** You can optimize tasks because you decide exactly how they're done.
* **Familiarity:** Many programming tasks fit well with this direct approach.

In Summary:

Imperative languages are like giving direct instructions to a computer step by step. You decide and control everything from start to finish, making it powerful for tasks that require clear, structured procedures.

**Declarative:** Declarative programming focuses on the end result. A declarative language is like making a request rather than giving step-by-step instructions. Instead of telling the computer exactly how to do something, you describe what you want to achieve, and the computer figures out how to do it.

Eg.: Azure Bicep and Terraform.

**Key Points:**

**Describing Goals:** You focus on stating what you want to happen, rather than the exact steps to make it happen.

**No Detailed Instructions:** You don't specify the order of operations or how data should be manipulated; you just define relationships and constraints.

* **Example:**
* Imagine ordering a pizza online. You choose the toppings and size, but you don't tell the kitchen how to assemble it— they follow a recipe (program) based on your choices.
* when you jump in a taxi, you declare to the driver where you want to go.

You don't tell him how to get there by providing turn-by-turn directions.