CODE BASICS

Python Basics: List, If Condition, For Loop

TAKEAWAYS

codebasics.io

Lists

1. Lists allow you to store sequential data.
2. Lists are ordered, meaning each item has a fixed position unless explicitly changed.
3. Python lists can hold different data types in a single list, including numbers, strings, and other lists. This means they are heterogeneous. For example **my_list = ["car", 4.5, True]**
4. Lists are mutable, allowing for elements to be added, removed, or changed within the same list.
5. List slicing lets you access a specific range of elements quickly, using the syntax **list[start : end : step]**

If Condition

1. If statements execute a block of code only if the condition is true, enabling conditional logic in programs.
2. Use elif to specify additional conditions if the initial if condition fails, allowing for multiple sequential checks.
3. else provides a fallback block of code when all preceding if and elif conditions are false.
4. Combine logical operators like and, or, and not within if statements to handle complex conditional expressions.
5. Nested if statements allow for checking multiple layers of conditions, enabling detailed decision-making processes in code.

For Loop

1. **For** loops iterate over a sequence, such as a list, tuple, or string, executing a block of code for each item.
2. The **range()** function is often used with **for** loops to generate a sequence of numbers, facilitating iteration over a set number of steps.
3. Loop control statements like **break** and **continue** can alter the flow of a loop, with **break** exiting the loop and **continue** skipping to the next iteration.
4. **enumerate()** will allow you to access both the index and the element from a list inside the for loop