\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  
**Topic 9 - If Statements**  
\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Introduction**

In Python, if statements allow you to perform actions based on conditions. This lesson introduces the basics of using if statements to make decisions in code and explains the syntax and structure required for proper use.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**What is an If Statement?**

An if statement checks whether a condition is True or False. If the condition is True, the code under the if statement is executed. If it’s False, that code is skipped.

**Example of an If Statement:**

*species = "cat"*

*if species == "cat":*

*print("Yep, it's a cat.")*

In this example:

* Condition: species == "cat"
* Action: print("Yep, it's a cat.")

If species is "cat", the message is displayed; otherwise, nothing happens.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Why Use If Statements?**

If statements are essential for:

* **Making Decisions:** Allowing programs to respond differently based on conditions.
* **Controlling Flow:** Code can adapt based on data or user input, making it dynamic.
* **Error Checking:** Checking conditions helps prevent errors or invalid actions, especially when managing user input or uncertain data.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  
**How to Use If Statements in Python**

1. Basic If Statement Syntax  
   The syntax for an if statement includes:
   * The if keyword: Written in lowercase.
   * Double Equals **==** : Used for comparison (not = which is for assignment).
   * Colon **:** : Ending the if line.
   * Indentation: Indent actions to specify what should run if the condition is True.

*species = "cat"*

*if species == "cat":*

*print("Yep, it's a cat.")*

1. Multiple Actions for a Single Condition  
   You can add multiple lines of code under an if statement, all indented to indicate they run only if the condition is True:

*if species == "cat":*

*status = "ok"*

*kingdom = "animal"*

*print("Yep, it's a cat.")*

1. Using If Statements with Numbers  
   If statements can also be used to evaluate numbers and arithmetic expressions:

*if 2 + 2 == 4:*

*print("Everything makes sense.")*

1. Indentation Rules in If Statements
   * Dependent Actions: Lines indented after an if statement execute only if the condition is True.
   * Independent Actions: Code that is not indented runs independently of the if condition.

*number\_of\_husbands = 1*

*if number\_of\_husbands == 1:*

*print("So far so good.")*

*print("Congratulations.")*

*print("All done")*

In this example, "All done" prints regardless of the condition.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Conclusion**

Python’s if statements are fundamental for making decisions in code. By checking conditions and indenting dependent actions, we can control the flow and responses of our programs, making them more adaptable and dynamic. Indentation is essential for correct structure, as Python relies on it to define code blocks within conditions.