**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**Topic 11 - Else and Elif Statements**  
**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**Introduction**

In Python, if statements let us execute code based on conditions. However, sometimes we want alternative actions if the main condition fails. This is where else and elif statements come in. These statements allow us to handle multiple conditions in a single block, making code cleaner and more efficient.

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_What are Else and Elif Statements?**

* **Else Statements**: Executes a block of code when the if condition fails.
* **Elif Statements**: Stands for "else if"; used to add multiple conditions that Python will test in order until it finds a match.

**Examples of Else and Elif:**

*if species == "cat":*

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

*else:*

*print("Nope, not a cat.")*

# Multiple conditions using elif

*if donut\_condition == "fresh":*

*buy\_score = 10*

*elif donut\_price == "low":*

*buy\_score = 5*

*else:*

*buy\_score = 0*

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**Why Use Else and Elif Statements?**

* **Efficiency**: Simplifies code by consolidating multiple conditions in one block.
* **Readability**: Makes the logic more intuitive and easier to follow.
* **Multiple Outcomes**: Enables handling of more than two possible outcomes based on a condition, improving code flow and reducing redundancy.

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**How to Use Else and Elif in Python**

1. **Else Statement**  
   The else statement provides an alternative action if the if condition fails:

*if species == "cat":*

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

*else:*

*print("Nope, not a cat.")*

* + **Format**: else is on a new line after the if, followed by a colon.
  + **Indentation**: Statements inside else are indented just like those in if.

1. **Elif Statement**  
   Use elif for additional conditions when the initial if test fails. You can use as many elif statements as needed:

*if donut\_condition == "fresh":*

*buy\_score = 10*

*elif donut\_price == "low":*

*buy\_score = 5*

*else:*

*buy\_score = 0*

* + **Format**: elif is followed by a condition and a colon, just like if.
  + **Execution Flow**: Python evaluates each elif in order, and the first one that passes will run, skipping the rest.

1. **Using Multiple Independent If Statements**  
   If you want multiple conditions to be checked independently (rather than stopping after the first match), use separate if statements:

*buy\_score = 0*

*if donut\_condition == "fresh":*

*buy\_score += 10*

*if donut\_filling == "chocolate":*

*buy\_score += 5*

*if donut\_price == "reasonable":*

*buy\_score += 7*

* + Here, each condition adds to buy\_score if it passes, allowing multiple conditions to influence the outcome.

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**Conclusion**

The else and elif statements expand the functionality of if statements, enabling multiple outcomes and a more logical flow. By understanding how to use else, elif, and independent if statements, Python developers can create code that is both efficient and adaptable, catering to various conditions seamlessly.