Conditional Statements in Python

UNDERSTANDING HOW TO CONTROL THE FLOW OF YOUR PROGRAMS

Introduction to Conditional Statements

What Are Conditional Statements?

- Conditional statements allow a program to act based on different conditions.
- They are used to control the flow of a program.

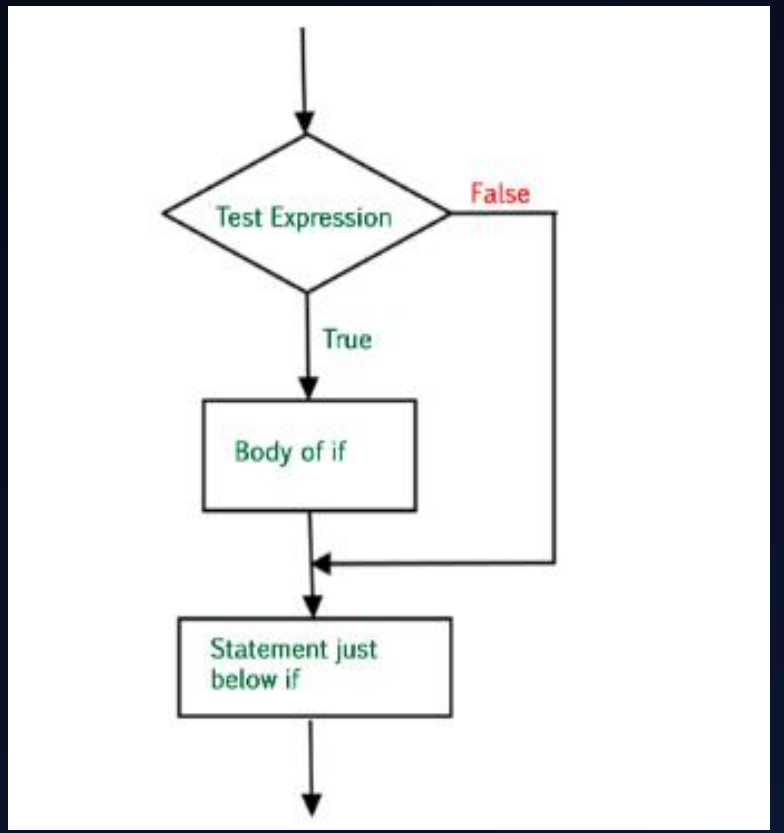
The if Statement

- Syntax:

```python
if condition:
    # code to execute if condition is true
```

- Example:

```python
age = 18
if age >= 18:
    print("You are an adult.")
```



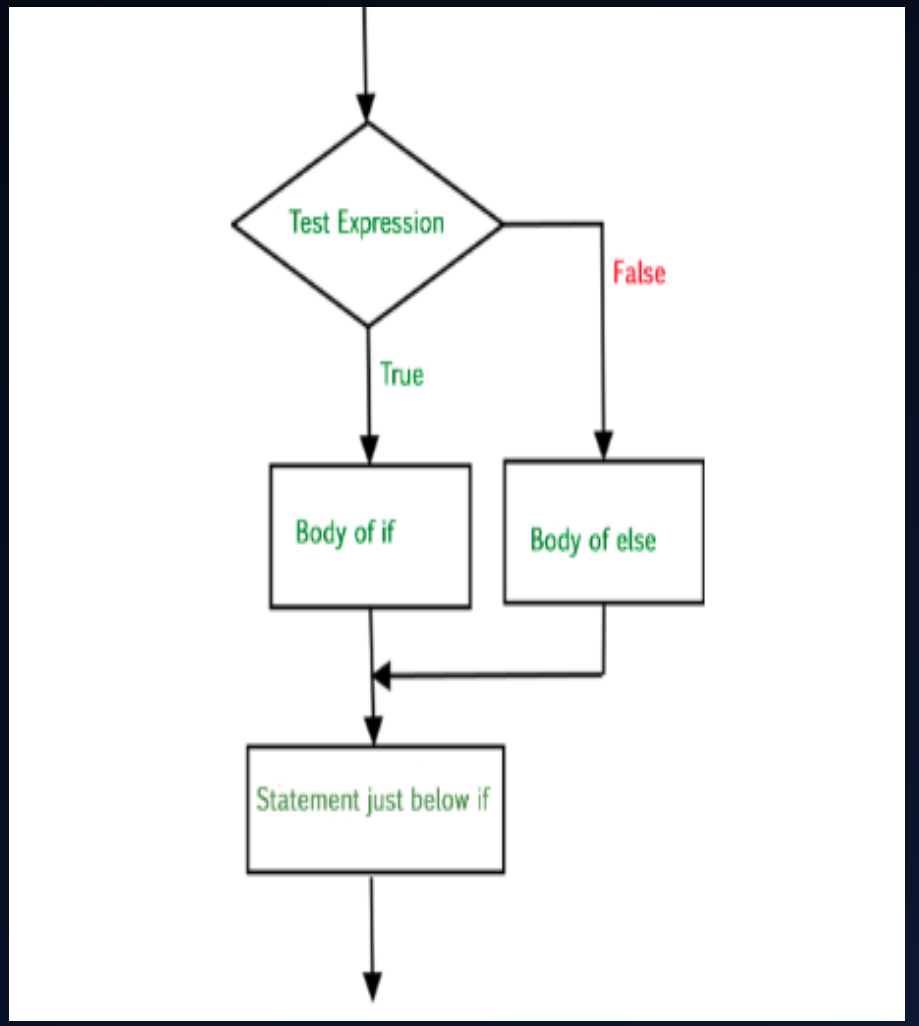
The else Statement

- Syntax:

```python
if condition:
    # code if condition is true
else:
    # code if condition is false
```

- Example:

```python
age = 16
if age >= 18:
    print("You are an adult.")
else:
    print("You are not an adult.")
```



The elif Statement

- Syntax:

```python
if condition1:
    # code if condition1 is true
elif condition2:
    # code if condition2 is true
else:
    # code if none of the above conditions are true
```

- **Example**:

```python
marks = 85
if marks >= 90:
    print("Grade: A")
elif marks >= 80:
    print("Grade: B")
else:
    print("Grade: C")
```

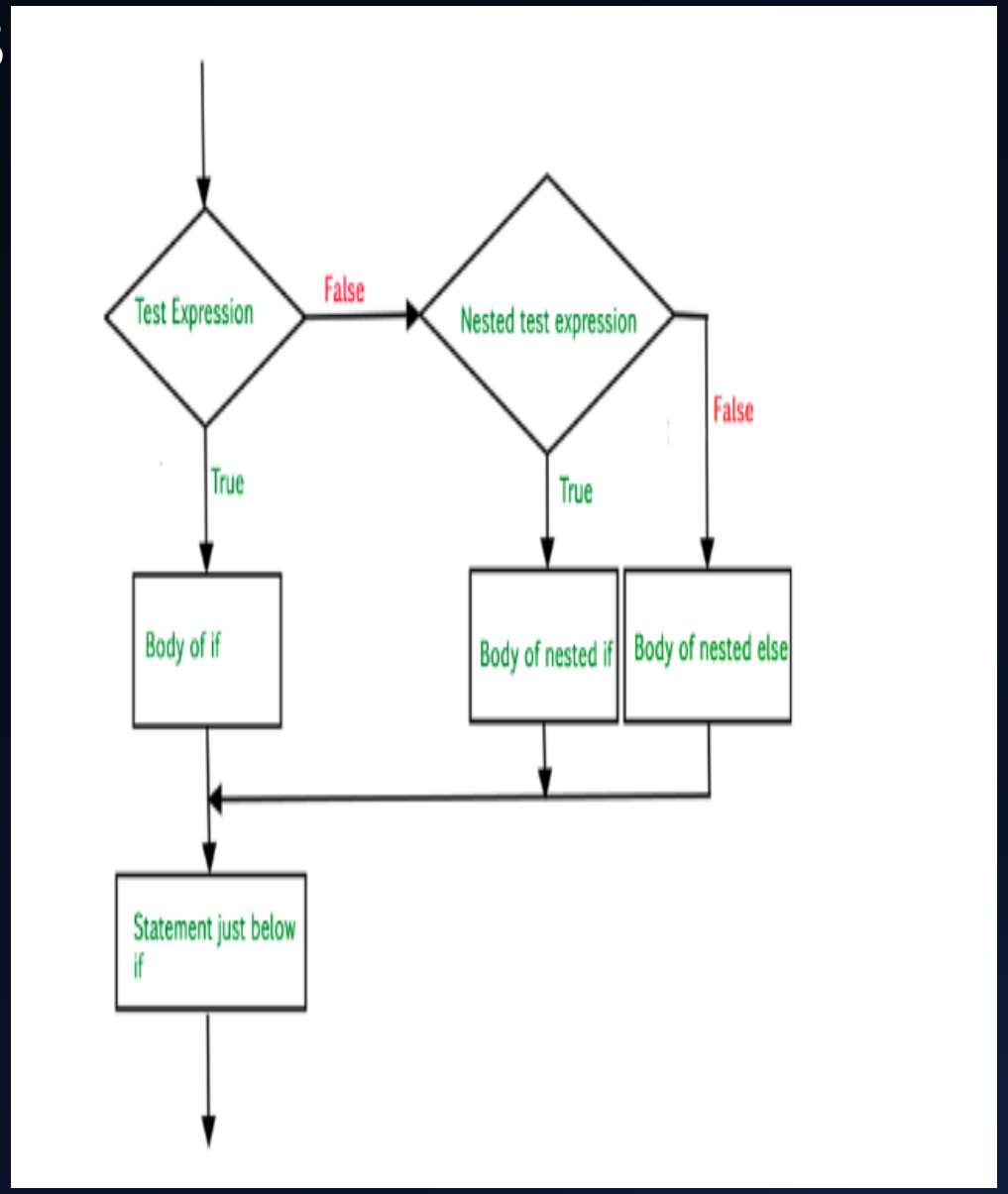
Nested Conditional Statements

WHAT IS NESTING?

- Nesting refers to placing one or more if, else, or elif statements inside another if or else statement.

Example:

```python
num = 10
if num > 0:
    if num % 2 == 0:
        print("Positive and Even")
    else:
        print("Positive and Odd")
else:
    print("Negative Number")
```



What is the Difference Between if-else and Nested If Statements in Python?

- ***if-else****: Refers to a simple conditional structure where if a condition is true, the if block executes; otherwise, the else block executes. There is only one alternative path.*
- ***Nested if statements:*** *Involve placing one or more if or if-else structures inside another if or else block. This setup is used to test for further conditions after a previous condition has been found to be true.*

Conditional Expressions (Ternary Operator)

- The ternary operator in Python is a concise way to perform conditional expressions. It allows you to evaluate a condition in a single line of code, returning one value if the condition is true and another if it is false.
- This form of conditional expression is often used to simplify code, making it more readable and concise, especially when the logic is straightforward.
- Syntax:

```python
value_if_true if condition else value_if_false
```

- Example:

```python
result = "Even" if num % 2 == 0 else "Odd"
print(result)
```

Common Mistakes

- **Indentation Errors**: Python uses indentation to define blocks of code.
- **Using = Instead of ==: = is for assignment; == is for comparison.**

Practice Problems

- Write an if-else statement to check if a number is positive, negative, or zero.
- Write a program that assigns grades based on a given score.
- Use nested if statements to categorize a person's age group (child, teenager, adult, senior).

Business use cases of conditional statements

- **1. Customer Segmentation and Personalization**
- **Use Case**: E-commerce websites often personalize product recommendations based on customer behavior.

```python
if customer_age < 18:
    show_products("youth_trending")
elif 18 <= customer_age < 35:
    show_products("young_adults_favorites")
else:
    show_products("classic_collections")
```

- **Benefit**: Increases customer engagement and sales by targeting the right products to the right audience.

- **Approval Workflows**
- **Use Case:** Automating HR, finance, or procurement approval processes based on set rules.

```python
if expense_amount > 1000:
    send_for_approval("manager")
elif expense_amount > 500:
    send_for_approval("supervisor")
else:
    auto_approve()
```

Conclusion

- Conditional statements control the flow of your program.
- Use if, Elif, and else to handle multiple conditions.
- Practice writing and nesting conditional statements.

Thank You

HAPPY LEARNING!