# Learn Python Programming from Scratch

## *Topic: If-Else Statements in Python*

### 1. What are If-Else Statements?

If-else statements are fundamental control flow structures that allow your program to make decisions based on conditions. Think of them as the decision-making brain of your program - they help your code choose different paths based on whether certain conditions are true or false.

### 2. Basic If Statement

The simplest form of conditional statement is the `if` statement. It executes code only when a condition is True.

In [1]:

```python
# Basic if statement syntax
age = 18

if age >= 18:
    print("You are eligible to vote!")

print("This line always executes")
```

```
You are eligible to vote!
This line always executes
```

### 3. If-Else Statement

The `else` clause provides an alternative path when the if condition is False.

In [2]:

```python
# If-else statement syntax
temperature = 25

if temperature > 30:
    print("It's hot outside!")
else:
    print("It's pleasant weather!")

print(f"Current temperature: {temperature}°C")
```

```
It's pleasant weather!
Current temperature: 25°C
```

### 4. If-Elif-Else Chain

Use `elif` (else if) to check multiple conditions in sequence. Python checks each condition in order and executes the first True condition it finds.

In [3]:

```python
# If-elif-else chain
marks = 87
```

```python
if marks >= 90:
    grade = 'A'
elif marks >= 80:
    grade = 'B'
elif marks >= 70:
    grade = 'C'
elif marks >= 60:
    grade = 'D'
else:
    grade = 'F'

print(f"Marks: {marks}, Grade: {grade}")
```

```
Marks: 87, Grade: B
```

## 5. Comparison Operators

Python provides several comparison operators for conditions:

- `==` - Equal to
- `!=` - Not equal to
- `>` - Greater than
- `<` - Less than
- `>=` - Greater than or equal to
- `<=` - Less than or equal to

In [4]:

```python
# Using comparison operators
x = 10
y = 20

print(f"x == y: {x == y}")  # False
print(f"x != y: {x != y}")  # True
print(f"x < y: {x < y}")    # True
```

```
x == y: False
x != y: True
x < y: True
```

## 6. Logical Operators

Combine multiple conditions using logical operators:

- `and` - Both conditions must be True
- `or` - At least one condition must be True
- `not` - Reverses the boolean value

In [5]:

```python
# Using logical operators
age = 25
income = 50000

# Check eligibility for a loan
if age >= 18 and income >= 30000:
    print("Eligible for loan")
else:
    print("Not eligible for loan")
```

```
Eligible for loan
```

## Exercises

1. Write a program to check if a number is positive, negative, or zero.
2. Create a simple grade calculator using if-elif-else statements.
3. Build a program that checks if a year is a leap year.
4. Write a program to find the largest of three numbers.
5. Create a simple login system that checks username and password.

---

# Practical Examples

Let's explore some practical examples of working with if-else statements in Python. These examples demonstrate real-world applications of conditional logic.

## Age Verification System

Here's a practical example of using if-else statements to create an age verification system with different access levels.

In [6]:

```python
# Age verification system with multiple conditions

# User's age
user_age = 25

# Check different age categories and permissions
if user_age < 13:
    category = "Child"
    can_vote = False
    can_drive = False
    can_drink = False
elif user_age < 18:
    category = "Teenager"
    can_vote = False
    can_drive = True  # In some places, with permit
    can_drink = False
elif user_age < 21:
    category = "Young Adult"
    can_vote = True
    can_drive = True
    can_drink = False  # In US
else:
    category = "Adult"
    can_vote = True
    can_drive = True
    can_drink = True

# Display the results
print(f"Age: {user_age}")
print(f"Category: {category}")
print(f"Can vote: {can_vote}")
print(f"Can drive: {can_drive}")
print(f"Can drink: {can_drink}")
```

```
Age: 25
Category: Adult
Can vote: True
Can drive: True
Can drink: True
```

## Grade Calculator with Detailed Feedback

This example shows how to use if-elif-else chains to create a comprehensive grade calculator that provides detailed feedback based on student performance.

In [7]:

```python
# Comprehensive grade calculator with detailed feedback

# Student information
student_name = "Alice"
math_score = 85
science_score = 92
english_score = 78

# Calculate average
average_score = (math_score + science_score + english_score) / 3

print(f"Student: {student_name}")
print(f"Math: {math_score}, Science: {science_score}, English: {english_score}")
print(f"Average Score: {average_score:.1f}")

# Determine grade and feedback using if-elif-else
if average_score >= 95:
    grade = "A+"
    feedback = "Outstanding performance! Excellent work across all subjects."
elif average_score >= 90:
    grade = "A"
    feedback = "Excellent work! Keep up the great effort."
elif average_score >= 85:
    grade = "B+"
    feedback = "Very good performance. You're doing well!"
elif average_score >= 80:
    grade = "B"
    feedback = "Good work! There's room for improvement."
elif average_score >= 75:
    grade = "C+"
    feedback = "Satisfactory performance. Consider studying more."
elif average_score >= 70:
    grade = "C"
    feedback = "Average performance. You need to work harder."
elif average_score >= 65:
    grade = "D+"
    feedback = "Below average. Please seek help from teachers."
elif average_score >= 60:
    grade = "D"
    feedback = "Poor performance. Immediate attention required."
else:
    grade = "F"
    feedback = "Failed. Please retake the exams."

print(f"Final Grade: {grade}")
print(f"Feedback: {feedback}")
```

```
Student: Alice
Math: 85, Science: 92, English: 78
Average Score: 85.0
Final Grade: B+
Feedback: Very good performance. You're doing well!
```

## Key If-Else Rules to Remember

Let's review the important rules and best practices for working with if-else statements:

- Use proper indentation (4 spaces) for code blocks under if, elif, and else
- Conditions are evaluated from top to bottom - first True condition wins
- Use `elif` instead of multiple separate `if` statements for mutually exclusive conditions
- The `else` clause is optional and executes when no previous condition is True
- You can nest if-else statements inside other if-else statements
- Use parentheses to group complex boolean expressions for clarity
- Use meaningful variable names and clear conditions for better readability
- Test your conditions with different input values to ensure they work correctly
- Consider edge cases and boundary values when writing conditions
- Use comparison operators (==, !=, <, >, <=, >=) to compare values
- Use logical operators (and, or, not) to combine multiple conditions

In [8]:

```python
# Examples of good conditional logic practices

# Example 1: Clear and readable conditions
user_status = "premium"
account_balance = 1000

if user_status == "premium" and account_balance > 500:
    discount = 0.20  # 20% discount
    print("Premium member with high balance - 20% discount applied")
elif user_status == "premium":
    discount = 0.10  # 10% discount
    print("Premium member - 10% discount applied")
elif account_balance > 1000:
    discount = 0.05  # 5% discount
    print("High balance - 5% discount applied")
else:
    discount = 0.00  # No discount
    print("Standard pricing applies")

print(f"Discount percentage: {discount * 100}%")
```

```
Premium member with high balance - 20% discount applied
Discount percentage: 20.0%
```

---

# Course Information

**Learn Python Programming from Scratch**

*Author:* Prakash Ukhalkar

*Topic:* Python Control Flow - If-Else Statements

---

*Built with ❤ for the Python learning community*