Study Material: Introduction to Python

Cosmic Charade

This study material covers the foundational concepts of Python, focusing on syntax, variables, data types, and control structures. Each section includes explanations and coding examples to help you grasp the basics of Python programming.

1. Python Syntax

Python's syntax is designed to be easy to read and understand. Here are some key features:

- **Indentation** is used to define the structure of the code (functions, loops, etc.), rather than curly braces `{}` used in languages like C or Java.
- **Case Sensitivity**: Python is case-sensitive, meaning that `variable` and `Variable` are two different names.

Example:

```
# A simple Python program
print("Hello, Python!")
```

```
Hello, Python!
```

2. Variables in Python

A variable in Python is a named location in memory used to store data. Variables are dynamically typed, meaning you don't have to declare their type explicitly.

Variable Naming Rules:

- Must begin with a letter (a-z, A-Z) or an underscore _.
- Can have numbers, but cannot start with a number.
- Are case-sensitive.

Example:

```
# Assigning values to variables
name = "Alice"  # String
age = 25        # Integer
height = 5.8    # Float

# Printing variables
print("Name:", name)
print("Age:", age)
print("Height:", height)
```

```
Name: Alice
Age: 25
Height: 5.8
```

3. Data Types in Python

Python supports several built-in data types, including:

- **Numeric Types**: `int`, `float`, `complex`
- **Text Type**: `str`
- **Sequence Types**: `list`, `tuple`, `range`
- **Mapping Type**: `dict` (In Python, a `dict` (short for dictionary) is a built-in data structure that stores a collection of key-value pairs.)
- **Set Types**: `set`, `frozenset`
- **Boolean Type**: `bool`

Example:

```
# Numeric types
x = 10           # Integer
y = 3.14         # Float
z = 3 + 4j       # Complex

# String type
name = "Python"

# List type
fruits = ["apple", "banana", "cherry"]

# Dictionary type
person = {"name": "John", "age": 30}

# Boolean type
is_raining = False

# Output data types
print(type(x))       # <class 'int'>
print(type(name))    # <class 'str'>
print(type(person))  # <class 'dict'>
```

```
<class 'int'>
<class 'str'>
<class 'dict'>
```

4. Control Structures

Control structures allow you to control the flow of execution in a program. Python supports:

- **Conditional Statements**: `if`, `elif`, `else`
- **Loops**: `for`, `while`
- **Break, Continue, and Pass**: To alter loop behavior

4.1. Conditional Statements

The `if` statement is used to test a condition. The `elif` (else if) and `else` keywords can be used to test multiple conditions.

Example:

```
# Conditional statement example
age = 16

if age >= 18:
    print("You are an adult.")
elif age >= 13:
    print("You are a teenager.")
else:
    print("You are a child.")
```

```
You are a teenager.
```

4.2. `for` Loop

The `for` loop is used to iterate over a sequence (such as a list, tuple, or string).

```
# for loop example
fruits = ["apple", "banana", "cherry"]

for fruit in fruits:
    print(fruit)
```

```
apple
banana
cherry
```

4.3. `while` Loop

The `while` loop is used to execute a block of code as long as a condition is true.

```
# while loop example
count = 5

while count > 0:
    print(count)
    count -= 1  # Decrease count by 1
```

```
5
4
3
2
1
```

4.4. `break`, `continue`, and `pass`

- **`break`**: Exits the loop entirely.
- **`continue`**: Skips the current iteration and proceeds to the next one.
- **`pass`**: Does nothing (a placeholder for future code).

```python
# break, continue, pass example
for num in range(1, 6):
    if num == 3:
        continue  # Skip the number 3
    elif num == 5:
        break  # Stop the loop
    else:
        print(num)
```

```
1
2
4
```

This study material provided an introduction to basic Python concepts, covering:

- **Syntax**: Basic structure and readability of Python.
- **Variables**: Dynamically typed, case-sensitive, and easily assignable.
- **Data Types**: Various built-in types like `int`, `float`, `str`, `list`, `dict`, and `bool`.
- **Control Structures**: `if-else` conditions, `for` and `while` loops, and the `break`, `continue`, `pass` commands.

With these foundational concepts, you are ready to explore more advanced topics like functions, file handling, and object-oriented programming (OOP) in Python.


thank you