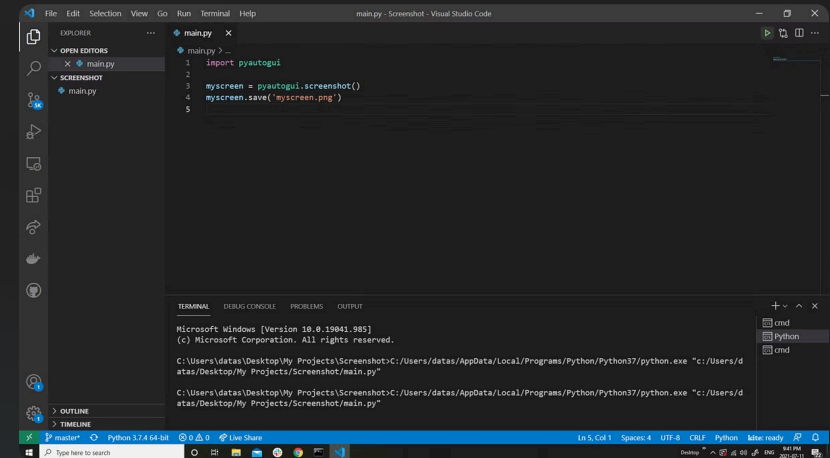


Control Statements in Python

Control statements are essential for controlling the flow of execution in a Python program. They allow you to make decisions, repeat actions, and manage the order in which your code runs.

 by Prabh Jit



```
main.py x
1 import pyautogui
2
3 myscreen = pyautogui.screenshot()
4 myscreen.save("myscreen.png")
5
```

```
Microsoft Windows [Version 10.0.19041.985]
(c) Microsoft Corporation. All rights reserved.

C:\Users\d\atas\Desktop\My Projects\Screenshot>C:\Users\d\atas\AppData\Local\Programs\Python\Python37\python.exe "c:/Users/d/atas/Desktop/My Projects/Screenshot/main.py"

C:\Users\d\atas\Desktop\My Projects\Screenshot>C:\Users\d\atas\AppData\Local\Programs\Python\Python37\python.exe "c:/Users/d/atas/Desktop/My Projects/Screenshot/main.py"
```

Conditional Statements: if, elif, else

Conditional statements allow you to execute different blocks of code based on whether a certain condition is true or false. The most common conditional statement is the `if` statement, followed by optional `elif` and `else` clauses.

1

if

The `if` statement is used to execute a block of code if a condition is true.

2

elif

The `elif` statement is used to execute a block of code if the previous `if` or `elif` condition is false, but the current condition is true.

3

else

The `else` statement is used to execute a block of code if all previous `if` and `elif` conditions are false.



Operator	Description	Example
==	If the values of two operands are equal, then the condition becomes true.	(A == B) is not true
!=	If values of two operands are not equal, then condition becomes true.	(A != B) is true
>	If the value of left operand is greater than the value of right operand, then condition becomes true.	(a > b) is not true
<	If the value of left operand is less than the value of right operand, then condition becomes true.	(a < b) is true
>=	If the value of left operand is greater than or equal to the value of right operand, then condition becomes true.	(a >= b) is not true
<=	If the value of left operand is less than or equal to the value of right operand, then condition becomes true.	(a <= b) is true

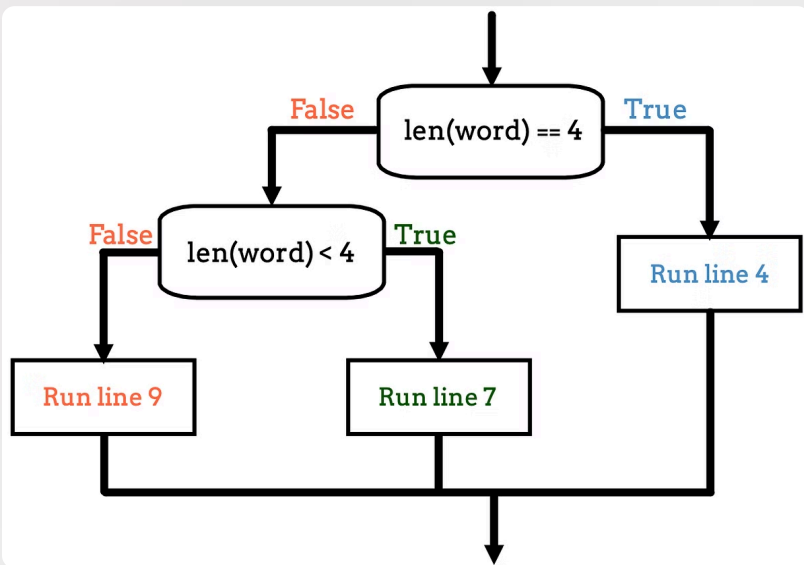
Comparison Operators in Conditional Statements

Comparison operators are used to compare values and determine whether a condition is true or false. These operators are often used in conjunction with conditional statements.

Operator	Description
==	Equal to
!=	Not equal to
>	Greater than
<	Less than
>=	Greater than or equal to
<=	Less than or equal to

Nested Conditional Statements

Nested conditional statements allow you to create more complex decision-making structures. This involves placing one or more conditional statements within another conditional statement.



1

Outer Condition

The outer conditional statement checks an initial condition.

2

Inner Condition

If the outer condition is true, the inner conditional statement is executed. The inner condition checks a secondary condition.

3

Code Execution

The appropriate block of code within the nested structure is executed based on the outcome of both conditions.

Loops: for, while

Loops are used to repeat a block of code multiple times. Python has two main types of loops: `for` loops and `while` loops.

for Loop

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

1. Iterate over each element in the sequence
2. Execute the code within the loop for each element

while Loop

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

1. Check the condition before each iteration
2. Execute the code within the loop if the condition is true
3. Repeat the process until the condition becomes false

Break and Continue Statements

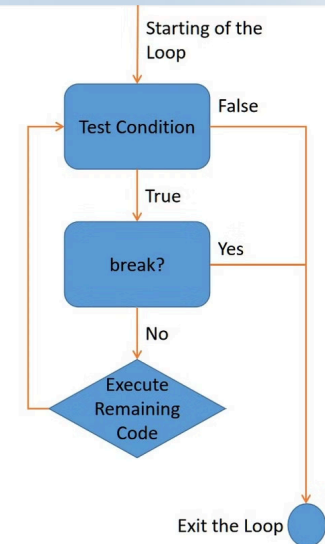
The `break` and `continue` statements can be used to modify the behavior of loops. These statements allow you to control the flow of execution within loops.

break

The `break` statement immediately exits the current loop. It's useful for stopping a loop prematurely based on a specific condition.

continue

The `continue` statement skips the remaining code within the current iteration of the loop and jumps to the beginning of the next iteration. This allows you to skip specific elements or iterations within a loop.



Ternary Operator

The ternary operator is a concise way to write conditional expressions. It provides a shorter syntax for choosing between two values based on a condition.



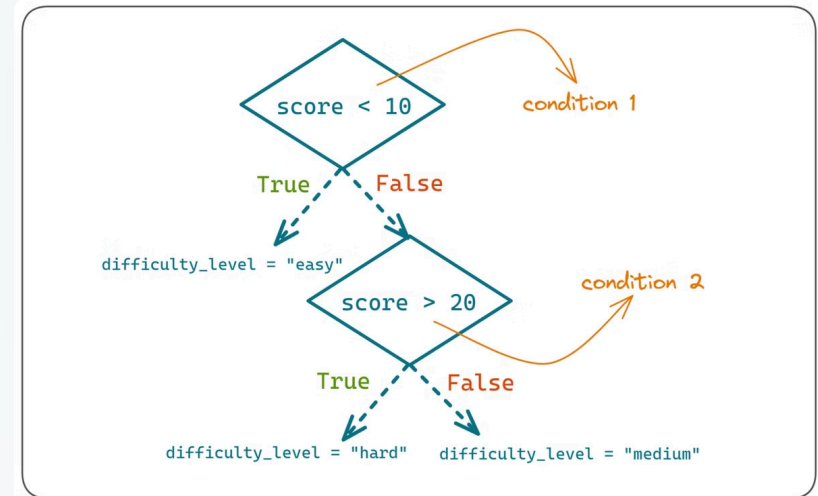
Syntax

The general syntax is `condition ? value_if_true : value_if_false`.



Example

```
``python result = "Even" if number % 2 == 0 else "Odd" ``
```



Conclusion and Best Practices

Control statements are fundamental to Python programming. By mastering these statements, you can create flexible and dynamic programs.

1 Indentation

Use consistent indentation to define code blocks within control statements. This ensures readability and correct execution.

2 Comments

Add comments to your code to explain the purpose of control statements and the logic behind your decisions.

3 Logic

Carefully plan the logic of your control statements to ensure they achieve the desired results.

