Module-01, Python Crash Course

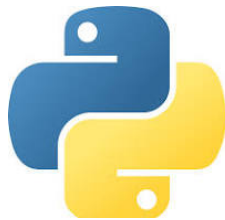
Control flow and loops

Dostdar Ali
Instructor

Data science and Artifical Intellgince
3-Months Course
at
Karakaroum international Univrsity

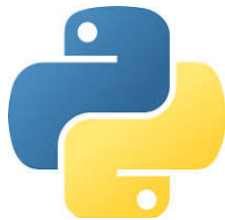
December 17, 2023

Table of Contents



Introduction

A program's control flow is the order in which the program's code executes. The control flow of a Python program is regulated by conditional statements, loops, and function calls. This section covers the if statement and for and while loops; functions are covered next lecture.



if, elif, else

- Purpose:
  Conditional statements allow the execution of different blocks of code based on whether a certain condition is true or false.
- Syntax:
  if condition:
  code block executed if condition is True
  elif another-condition:
  code block executed if another-condition is True
  else:
  code block executed if none of the conditions are True

Example

The if statement checks if x is greater than 0. If true, it prints "x is positive." If false, it moves to the elif statement, checking if x is equal to 0. If true, it prints "x is zero." If both conditions are false, the else statement is executed, printing "x is negative."

if, elif, else

- Purpose:
  Conditional statements allow the execution of different blocks of code based on whether a certain condition is true or false.
- Syntax:
  if condition:
  code block executed if condition is True
  elif another-condition:
  code block executed if another-condition is True
  else:
  code block executed if none of the conditions are True

Example

The if statement checks if x is greater than 0. If true, it prints "x is positive." If false, it moves to the elif statement, checking if x is equal to 0. If true, it prints "x is zero." If both conditions are false, the else statement is executed, printing "x is negative."

if, elif, else

- Purpose:
  Conditional statements allow the execution of different blocks of code based on whether a certain condition is true or false.
- Syntax:
  if condition:
  code block executed if condition is True
  elif another-condition:
  code block executed if another-condition is True
  else:
  code block executed if none of the conditions are True

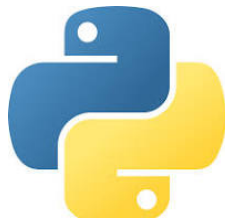
Example

The if statement checks if x is greater than 0. If true, it prints "x is positive." If false, it moves to the elif statement, checking if x is equal to 0. If true, it prints "x is zero." If both conditions are false, the else statement is executed, printing "x is negative."

Nested condition

,

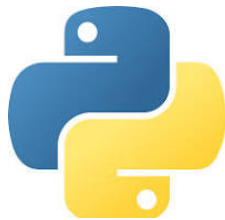
- Purpose:
  Nested conditional statements involve placing one conditional statement inside another. This is useful when you need to check multiple conditions.
- Syntax
  if condition-outer:
  code block for outer condition
  if condition-inner:
  code block for inner condition
  else:
  code block executed if inner condition is false
  else:
  code block executed if outer condition is false



Nested condition

,

- Purpose:
  Nested conditional statements involve placing one conditional statement inside another. This is useful when you need to check multiple conditions.
- Syntax
  if condition-outer:
  code block for outer condition
  if condition-inner:
  code block for inner condition
  else:
  code block executed if inner condition is false
  else:
  code block executed if outer condition is false



Example

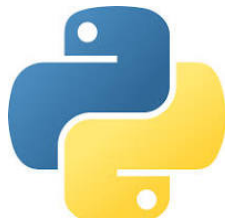
The outer if checks if the number is positive. If true, it goes into the nested if and checks if the number is even or odd, printing the respective message. If the outer if is false, it prints "Not a positive number."

Explanation:

- The outer if statement checks a condition.
- If the outer condition is true, it enters the code block associated with the outer if.
- Inside the outer if block, there is another if statement (nested if) that checks an inner condition.
- If the inner condition is true, it enters the code block associated with the inner if.
- If the inner condition is false, it enters the else block associated with the inner if.
- If the outer condition is false, it enters the else block associated with the outer if.

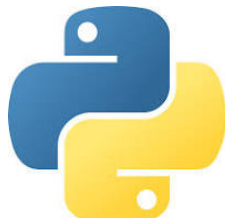
range() Function with for Loop:

- Purpose:
  The for loop is used for iterating over a sequence (such as a list, tuple, string, etc.) or other iterable objects.
- range() Function with for Loop:
- Purpose:
  The range() function generates a sequence of numbers, and when combined with a for loop, it provides a concise way to iterate a specific number of times.



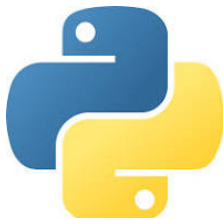
range() Function with for Loop:

- Purpose:
  The for loop is used for iterating over a sequence (such as a list, tuple, string, etc.) or other iterable objects.
- range() Function with for Loop:
- Purpose:
  The range() function generates a sequence of numbers, and when combined with a for loop, it provides a concise way to iterate a specific number of times.



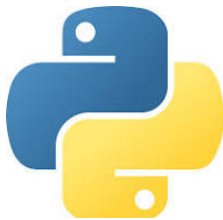
range() Function with for Loop:

- Purpose:
  The for loop is used for iterating over a sequence (such as a list, tuple, string, etc.) or other iterable objects.
- range() Function with for Loop:
- Purpose:
  The range() function generates a sequence of numbers, and when combined with a for loop, it provides a concise way to iterate a specific number of times.



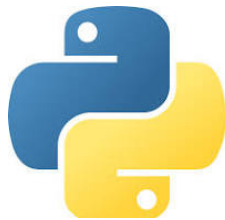
While loop

- Purpose:
  The while loop repeatedly executes a block of statements as long as the given condition is true.
- This while loop continues to execute as long as the condition count ¡ 5 is true. It prints the value of count and increments it in each iteration.
- break and continue Statements:
- break Purpose: Used to exit a loop prematurely based on a certain condition.
- continue Purpose: Skips the rest of the code inside a loop for the current iteration and moves to the next iteration.



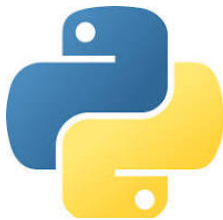
While loop

- Purpose:
  The while loop repeatedly executes a block of statements as long as the given condition is true.
- This while loop continues to execute as long as the condition count ¡ 5 is true. It prints the value of count and increments it in each iteration.
- break and continue Statements:
- break Purpose: Used to exit a loop prematurely based on a certain condition.
- continue Purpose: Skips the rest of the code inside a loop for the current iteration and moves to the next iteration.



While loop

- Purpose:
  The while loop repeatedly executes a block of statements as long as the given condition is true.
- This while loop continues to execute as long as the condition count ¡ 5 is true. It prints the value of count and increments it in each iteration.
- break and continue Statements:
- break Purpose: Used to exit a loop prematurely based on a certain condition.
- continue Purpose: Skips the rest of the code inside a loop for the current iteration and moves to the next iteration.



Great Job
Thank you