Python: Flow Control Statements

Connect with me: Youtube | LinkedIn | WhatsApp Channel | Web | Facebook | Twitter

- Download PDF
- To access the updated handouts, please click on the following link: https://yasirbhutta.github.io/python/docs/control-flow.html

Flow control statements

Flow control statements in Python determine the order in which your code is executed. They allow you to make decisions, repeat actions, and control the program's flow based on specific conditions.

1. Conditional Statements (if, else, elif)
2. Looping Statements (for, while)
3. Loop Control Statements (break, continue, pass)
4. Match Statement (Python 3.10+)

Conditional Statements (if, else, elif)

- The `if` statement in Python is a conditional statement that allows you to execute a block of code only if a certain condition is met.
- Make decisions using `if`, `elif`, and `else` statements.

The general **syntax** of the if statement is as follows:

```
if condition:
    # code to execute if condition is True
```

The `condition` can be any logical expression. If the condition is evaluated to `true`, the block of `statements` is executed. Otherwise, the block of `statements` is skipped.

Here is a simple example of an if statement in MATLAB:

Example #:

```
x = 10

if x > 5:
    print('x is greater than 5.')
```

This code will print the message `x is greater than 5.` to the console.

You can also use `elif` statements to check for multiple conditions. The general **syntax** of the **elif statement** is as follows:

```
if condition1:
    # code to execute if condition1 is True
elif condition2:
    # code to execute if condition2 is True
```

If the `condition` for the if statement is evaluated to `false`, the python interpreter will check the `condition` for the first elif statement. If the condition for the elif statement is evaluated to `true`, the corresponding block of `statements` is executed. Otherwise, the python interpreter will check the `condition` for the next elif statement, and so on.

Here is an example of an if statement with an elif statement:

Example #1:

```
x = 3

if x > 5:
    print('x is greater than 5.')
elif x < 5:
    print('x is less than 5.')
```

This code will print the message "x is less than 5." to the console.

You can also use an else statement to check for all other conditions. The general syntax of the else statement is as follows:

```
if condition1:
    # code to execute if condition1 is True
elif condition2:
    # code to execute if condition2 is True
else:
    # code to execute if none of the conditions are True
```

If all of the conditions for the if and elseif statements are evaluated to `false`, the block of `statements` in the `else` statement is executed.

Here is an example of an if statement with an `elif` statement and an `else` statement:

```
x = 2

if x > 5:
    print('x is greater than 5.')
elif x == 5:
    print('x is equal to 5.')
else:
    print('x is less than 5.')
```

This code will print the message "x is less than 5." to the console.

Example #2: Using if-elif-else

```
x = 10

if x < 0:
    print("Negative")
elif x == 0:
    print("Zero")
else:
    print("Positive")
```

Example #3: Video: How to check if a number is odd or even **Example #4:** Video: Python Program to Find Grade of a Student Using if elif else

- Video: Use Walrus Operator with if-else (New)
- Video: How to Write Single-Line Code Instead of If-Else Statements

Example #:

- Python Quiz -IF

loops

- There are two ways to create loops in Python: with the for-loop and the while-loop.
- Repeat actions using `for` and `while` loops.

for loop

- A for loop in Python is a programming statement that repeats a block of code a fixed number of times.
- The for-loop is always used in combination with an iterable object[^1], like a list or a range.
- The Python for statement iterates over the members of a sequence in order, executing the block each time.
- Video: For loops in Python

Syntax:

```
for item in iterable:
    # code block
```

`iterable` is a sequence of elements such as a list, tuple, dictionary, set, or string. item is a variable that takes on the value of each element in the sequence, one at a time. The code block is executed once for each element in the sequence.

range() function:

- We can use the `range()` function as an iterable in a for loop in Python. The range() function returns a sequence of numbers, starting from 0 by default, and increments by 1 (by default), and ends at a specified number.
- We can also specify the starting value and the increment value of the sequence using the range() function. For example, range(2, 10, 2) returns a sequence of numbers starting from 2, incrementing by 2, and ending at 8. read more ...

Example #: Print Numbers from 1 to 5

Question: Write a MATLAB program to print the numbers from 1 to 5, using a for loop.

Example #:

```
for i in range(6):
    print(i)
```

Example #: Printing "Building the future, one line at a time." 5 Times Using a for Loop

Question: Write a Python program to print the string "Building the future, one line at a time." 5 times, using a for loop.

Example #:

```
for i in range(5):
    print("Building the future, one line at a time.")
```

Example #: Sum of Numbers from 1 to N

Question: Write a python program to calculate the sum of the first N natural numbers using a for loop.

```
N = 10
sum = 0
for i in range(1,N+1):
    sum = sum + i
print(f"Sum = {sum}")
```

Example #: Print Even Numbers from 2 to 10

Question: Write a Python program to display the even numbers from 2 to 10, inclusive, using a for loop.

```
sum = 0
for i in range(2,11):
    if i%2 == 0:
        sum +=i
```

```
        print(i)
print(f"Sum of even numbers: {sum}")
```

Example #: Strings as an iterable

```
string = "python is versatile"
for x in string:
    print(x)
```

Example #: Lists as an iterable

```
collection = ['python', 5, 'd']
for x in collection:
    print(x)
```

Example #: Loop over Lists of lists

```
list_of_lists = [ [1, 2, 3], [4, 5, 6], [7, 8, 9]]
for list in list_of_lists:
    for x in list:
        print(x)
```

- A **nested loop** is a loop inside another loop. It is a powerful programming technique that can be used to solve a wide variety of problems.

Example #: Nested Loops - Multiplication Tables

```
for i in range(1, 11):
    print(f"Multiplication table of {i}")
    for j in range(1, 11):
        print('%d * %d = %d' % (i, j, i*j))
```

for loop examples:

- Video: String as an iterable
- Video: 6 Ways to use List in For loop in Python
- Video: Underscore to Ignore Values in for loop
- How to Print Multiplication Tables in Python
- String as an iterable using for loop
- Calculate the sum of the first N natural numbers

Further reading:

- For loop - Python wiki

while loop

- A while loop in python is a control flow statement that repeatedly executes a block of code until a specified condition is met.

Syntax:

```
while condition:
    # code block
```

Here, condition is a boolean expression that is evaluated before each iteration of the loop. If the condition is `True`, the code block is executed. The loop continues to execute as long as the condition remains True.

Example #: Print numbers from 1 to 10 using while loop

Question: Write a Python program to print the numbers from 1 to 10, using a while loop?

```
count = 1  # Start counting at 1
while count <= 10:  # Keep counting as long as we're less than or equal to 10
    print(count)  # Print the current number
    count += 1  # Add 1 to the count for the next round
```

Example #: Print "Hello, world!" 5 times using while loop

Question: Write a Python program to print the string "Hello, world!" 5 times, using a while loop?

```
i = 1
while i <= 10:
    print('Hello, world!')
    i += 1
```

Example #: Sum of numbers from 1 to 100 using while loop

Question: Write a MATLAB program to calculates the sum of the numbers from 1 to 100 using a while loop.

```
i = 1
sum = 0

while i <= 100:
    sum += i
    i += 1

print(f'Sum = {sum}');
```

Example #: Sum of even numbers from 2 to 20 using while loop

Question: Write a Python program to calculate the sum of the even numbers from 2 to 20 using a while loop.

```
sum = 0  # Initialize a variable to store the sum
number = 1

while number <= 20:
    if number%2 == 0:
      sum += number  # Add the current number to the sum
    number += 1

print(f'The sum of even numbers from 1 to 20 is: {sum}')
```

Example: Square of numbers less than 5 using while loop

Question : Write a program that prints the sum of the squares of all the numbers from 1 to 4, using a while loop.

```
i = 1
while i < 5:
    square = i ** 2
    print(f'Square of {i} is {square}')
    i += 1
```

Output:

```
Square of 1 is 1
Square of 2 is 4
Square of 3 is 9
Square of 4 is 16
```

Example #:

Question: Write a Python program to prompt the user to enter lines of text until the user enters a blank line. The program should then display the message "You entered a blank line.".

```
inputStr = 'Start';

while inputStr != "":
  inputStr = input("Enter a line of text:")

print('You entered a blank line.')
```

Example: Sum of given numbers till the number entered is zero

Question: Write a Python program to add all the numbers entered by the user until the user enters zero. The program should display the sum of the numbers.

```
sum = 0; # Initialize the sum

# Prompt the user to enter a number
number = int(input('Enter a number: ')) # int() Convert string input to integer

# While the number entered is not zero, add the number to the sum and prompt the
user to enter another number
while number != 0:
    sum += number
    number = int(input('Enter another number: ')) # int() Convert string input to
integer

# Display the sum of the numbers
print(f'The sum of the numbers is: {sum}')
```

Example : While loop from 1 to infinity, therefore running forever.

```
x = 1
while True:
    print("To infinity and beyond! We're getting close, on %d now!" % (x))
    x += 1
```

while loop examples:

- How to Count Digits in a Positive Integer using Python
- How to Count Occurrence of a Specific Digit in a Number using Python

See also:

- Video: Learn how to use while loops
- Video: Learn how to use INFINITE while loop
- Video: Python while loop

The Range() function

- Video: The range() Function
- Video: Use of range() in for loop

Loop Control Statements (break, continue, pass)

These statements modify the behavior of loops.

break: Terminates the loop entirely. **continue:** Skips the current iteration and moves to the next one. **pass:** Does nothing, often used as a placeholder.

`break`

Exits the loop prematurely.

```
for item in sequence:
    if some_condition:
        break  # exit the loop
```

`continue`

Skips the current iteration and proceeds to the next iteration of the loop.

```
for item in sequence:
    if some_condition:
        continue  # skip the rest of the code in this iteration
    # code to execute if some_condition is False
```

`pass`

A null statement, used as a placeholder.

```
if condition:
    pass  # do nothing
```

Example #:

```
for x in range(3):
    if x == 1:
        break
```

Example #:

```
for i in range(10):
    if i == 5:
        break  # exit the loop when i is 5
    if i % 2 == 0:
        continue  # skip even numbers
    print(i)  # print only odd numbers less than 5
```

- Video: How to Effectively Use Break and Continue Statements

- Video: Using Python break statement with a while loop

The else Clauses on Loops

Example #: for..else

```
for x in range(3):
    print(x)
else:
    print('Final x = %d' % (x))
```

Match Statement (Python 3.10+)

The match statement offers a more concise way to handle multiple comparison cases.

```
def get_fruit_color(fruit):
    match fruit:
        case "apple":
            return "red"
        case "banana":
            return "yellow"
        case _:
            return "unknown"
```

Python Challenge to test your knowledge: Quiz1 | Quiz2 | Quiz3 | Quiz4 | Quiz5

Key Terms

- for
- while
- range
- pass
- else
- match
- break
- continue
- f-string

True/False (Mark T for True and F for False)

Multiple Choice (Select the best answer)

for loop:

> What is the output of the following code?

```
fruits = ["Apple", "Banana", "Cherry"]
for i, fruit in enumerate(fruits):
    if i == 1:
        print(fruit)
```

1. ☐ Apple
2. ☐ Banana
3. ☐ Cherry
4. ☐ Error

related video: https://youtu.be/-FErgsl9njQ

> What is the correct syntax for a for loop in Python?

1. ☐ for (int i = 0; i < 10; i++):
2. ☐ for i in range(10):
3. ☐ for i = 0 to 9:
4. ☐ for i in 10:

What will be the output of the following code?

```
for i in range(5):
  print(i * 2)
```

a) 0 1 2 3 4 b) 2 4 6 8 10 c) 10 8 6 4 2 d) 0 2 4 6 8

How many times will the following loop execute?

for i in [1, 2, 3, 3]: print("Hello")

a) 3 b) 4 c) 1 d) Infinitely

What is the primary purpose of a for loop in Python? a. To define a function b. To iterate over a sequence c. To create a conditional statement d. To perform mathematical operations

In Python, what does the range() function do when used in a for loop? a. Generates a sequence of numbers b. Defines a list c. Calculates the average d. Determines the length of a string

How is the syntax for a for loop in Python? a. for x in range(10): b. while x < 10: c. loop for x in 10: d. for x = 0; x < 10; x++:

In a for loop, what is the role of the loop variable? a. It is used to define the loop b. It holds the result of the loop c. It is the counter for the loop iterations d. It is optional and can be omitted

Which of the following is the correct way to iterate over elements in a list using a for loop? a. for item in my_list: b. for i = 0; i < len(my_list): c. for element = my_list: d. for (i, item) in enumerate(my_list):

How can you iterate over both the index and elements of a list using a for loop? a. for i in my_list: b. for (i, element) in enumerate(my_list): c. for element in range(my_list): d. for index, element in my_list:

What is the output of the following code?

```
numbers = [1, 2, 3, 4, 5]
for x in numbers:
    print(x * 2)
```

a. 2 4 6 8 10 b. 1 2 3 4 5 c. 1 4 9 16 25 d. 2 4 8 16 32

while loop

What is the output of the following code?

```
count = 0
while count < 3:
    print(count)
    count += 1
```

(a) 0 1 2 (b) 0 1 (c) 1 2 3 (d) The code will run indefinitely.

What will happen if you try to modify the loop variable within the body of a while loop?

(a) The loop will continue as normal. (b) The loop will terminate immediately. (c) The loop may behave unexpectedly, depending on how the variable is modified. (d) The loop will always run indefinitely.

What is the correct syntax for a while loop in Python?

(a) while (condition): (b) while condition {} (c) while condition: (d) while (condition) {}

What is the output of the following code?

```
x = 10
while x > 0:
    print(x)
    x -= 2
```

(a) 10 8 6 4 2 (b) 9 7 5 3 1 (c) 10 9 8 7 6 (d) The code will run indefinitely.

What is the purpose of the else clause in a while loop?

(a) To execute a block of code if the loop condition is never true. (b) To execute a block of code if the loop completes without being terminated by a break statement. (c) To execute a block of code if the loop encounters an error. (d) The else clause cannot be used with a while loop.

if statement:

Which of the following correctly represents the syntax of an if statement in Python? a) if condition { block of code } b) if(condition) { block of code } c) if condition: block of code

What is the syntax for a simple if statement in Python? a. if x == 10: b. for x in range(10): c. while x < 10: d. if (x == 10) then:

What is the purpose of the else block in an if statement?

a) To execute a code block when the if condition is True b) To execute a code block when the if condition is False c) To create an infinite loop d) To define a function

What happens when none of the conditions in an if-elif-else chain are True, and there is no else block?

a) The program raises an error. b) The program executes the first if block. c) The program executes the last elif block. d) The program does nothing and continues to the next statement.

What will be the output of the following code?

```
x = 10
y = 5
if x < y:
    print("x is greater than y")
```

a) "x is greater than y" b) "x is less than y" c) Nothing will be printed d) An error will occur

What happens if none of the conditions in an if-elif chain are True, and there is no else block?

a) The program raises an error b) The program executes the first if block c) The program does nothing and continues to the next statement d) The program executes the last elif block

Which of the following statements is True about indentation in Python?

a) Indentation is optional b) Indentation is used to define code blocks c) Indentation is four spaces wide d) Both b and c

What will be the output of the following code?

```
number = 10
if number % 2 == 0:
    print("Number is even")
else:
    print("Number is odd")
```

Which of the following is the correct way to compare if two variables are equal in an if statement? a. if x equals 5: b. if x == 5: c. if x = 5: d. if x := 5:

What does the following code snippet do?

```
if x > 0:
    print("Positive")
elif x < 0:
```

```
    print("Negative")
else:
    print("Zero")
```

a. Prints "Positive" if x is greater than 0, "Negative" if x is less than 0, and "Zero" if x is 0. b. Prints "Positive" if x is less than 0, "Negative" if x is greater than 0, and "Zero" if x is 0. c. Prints "Positive" if x is 0, "Negative" if x is greater than 0, and "Zero" if x is less than 0. d. Causes an error because the conditions are conflicting.

How can you check if a value is NOT equal to 10 in an if statement?

a. if x != 10: b. if x <> 10: c. if x =! 10: d. if x not 10:

Exercises

if statement

Review Questions

for loop:

- What is the difference between a for loop and a while loop in Python?
- Can you use a for loop to iterate over a string?

if statement:

- Can you have multiple elif blocks in an if-elif-else statement?
- What is the purpose of nesting if statements?
-

References and Bibliography

[^1]: In Python, an iterable object is an object that you can loop over using a "for" loop. It's any object that can return its elements one at a time.

Which of the following correctly fixes the syntax error in the code below?

```
if x == 10  # Missing colon
    print("x is 10")
```

A) Remove the comment. B) Add a colon after `if x == 10`. C) Add parentheses around `x == 10`. D) Indent the print statement correctly.

Which of the following will NOT cause a syntax error in Python?

A)

```
1st_variable = 10
```

B)

```
if x == 10:
print("x is 10")
```

C)

```
print("Hello World!"
```

D)

```
print "Hello World!)
```