pyladies

Bratislava #4

Mastering our skills - if, else, for, while etc.

GIRL'S DAY 2019

25. 04. 2019

Today’s agenda

Practicing fundamental python statements by solving many small problems quickly today

python gotchas and more advance extras in-between for more advanced coders

Do not forget your virtual environment

- Visual Studio Code on Windows
  - python -m venv <name>
  - In Select Interpreter > choose the newly created environment
    - for this to work I had to do:
      - find PowerShell app
      - right-click and select \`Run as admin\`
      - write this command:
        - Set-ExecutionPolicy RemoteSigned
      - save with \`Yes for all\`

  - When the env is activated you see the (<name>) on the beginning of the powershell
- For Linux
  - `sudo apt install virtualenv`
  - `virtualenv -p python3 env` (be in the folder where you want the environment to be created)
  - `source env/bin/activate`
  - `deactivate`

Hard to estimate the right level for this lecture

… so lecture will try to explain everything, even for beginners

… parts marked with asterisk (*) are for advanced coders

… so if anything is too boring for you - use the time to solve more complicated problems online

https://projecteuler.net/

https://www.hackerrank.com/

https://people.ksp.sk/~acm/ (in Slovak)

Recap of `if/elif/else`

```
if “some condition” :
    Code to run if condition is true
elif “other condition” :
    Code to run if the other condition is true
else:
        Default code that runs if no conditions
        are true
```

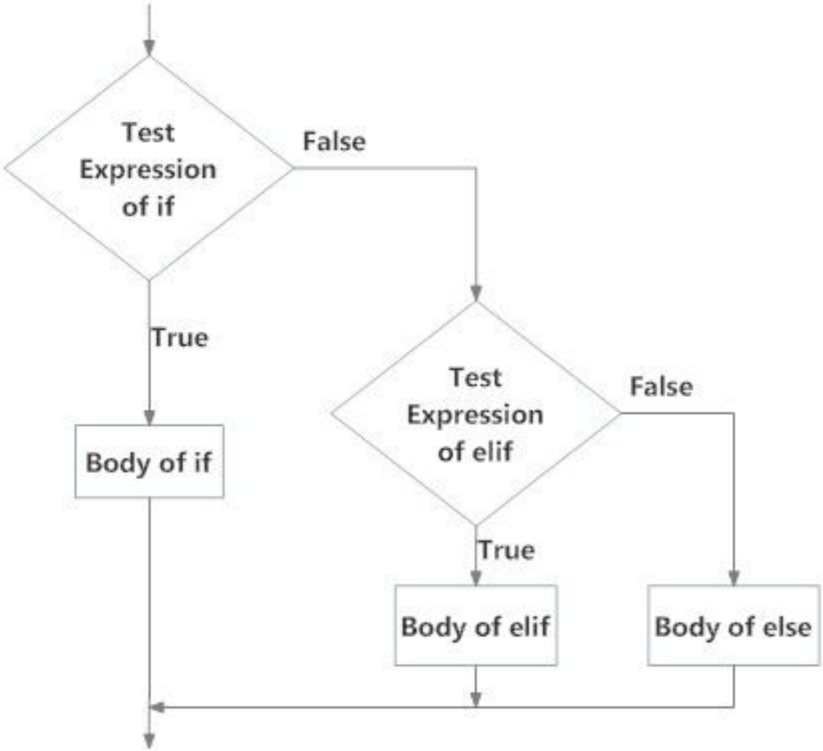


Fig: Operation of if...elif...else statement

Conditions

Symbol	Example	Description
`==`, `!=`	`1 == 1`, `1 != 1`	Equal, not equal
`<`, `>`	`3 < 5`, `3 > 5`	Greater than, less than
`<=`, `>=`	`3 <= 5`, `3 >= 5`	Greater than or equal, less than or equal

Symbol	Example	Description
`and`	`True and False`	„and“
`or`	`True or False`	„or“
`not`	`not False`	„not“

Example

```
>>> x = int(input("Please enter an integer: "))
Please enter an integer: 42
>>> if x < 0:
...     print('Negative number')
... elif x == 0:
...     print('Zero')
... elif x == 1:
...     print('Single')
... else:
...     print('More')
...
```

Exercises

- Write a program that asks the user if they love programming. Answer according to their answer “Y” or “N” or print out that you got an unexpected answer, otherwise.

- Read three numbers from the user. Print them out sorted from the largest to the smallest. (How many conditions did you use? Is it possible to reduce the number?)

Conditional expressions - “quick if”

```
to_do_if_true if condition else to_do_if_false
```

```
>>> my_nymber = 5
>>> print(“non zero number” if my_nymber else “it’s zero”)
non zero number
>>> my_nymber = 0
>>> print(“non zero number” if my_nymber else “it’s zero”)
it’s zero
```

* ShortHand Ternary

```
condition or what to do if not true
```

```
>>> output = None
>>> msg = output or "No data returned"
>>> print(msg)
No data returned
```

Good for quickly testing function returns and giving back meaningful message

* ternary operator with tuples

```
(if_test_is_false, if_test_is_true)[test]
```

```
>>> nice = True
>>> personality = ("mean", "nice")[nice]
>>> print("The cat is ", personality)
The cat is nice
```

However do not use this!!! It’s confusing and can lead to errors.

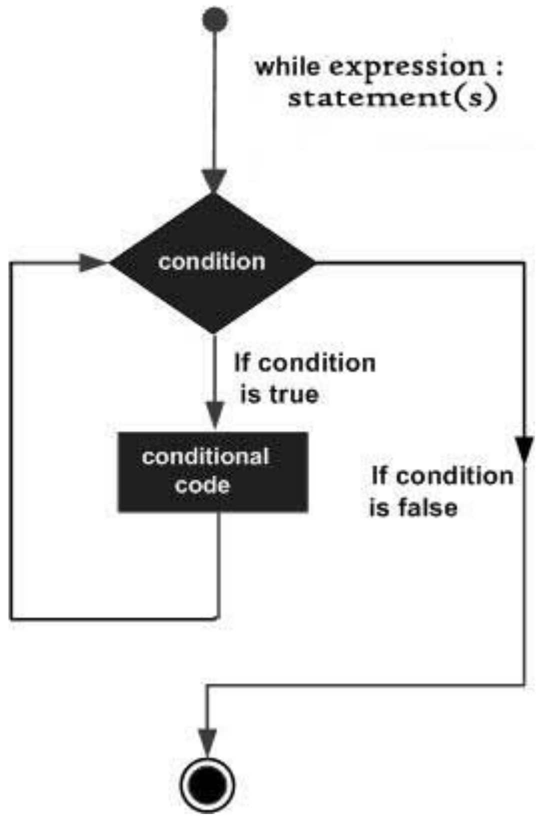
```
>>> condition = True
>>> print(2 if condition else 1/0)
2
#Output is 2, only the first part of
the code is executed

>>> print((1/0, 2)[condition])
#ZeroDivisionError is raised - both
parts of the tuple are first
evaluated
```

Recap of `while` loop

```
while condition:

     Body of while to loop through
```



Example

```
>>> a = 0
>>> while a < 10:
...     a = a + 1
...     print(a)
```

```
>>> while True:
...     print('Write letter a')
...     usr_input = input()
...     if usr_input == ‘a’:
...         break;
Write letter a
B
Write letter a
a
>>>
```

While loop with else

```
>>> sum = 0
>>> while i < 5:
...     print (i)
... else:
...     print("No items left")
0
1
2
3
4
No items left
```

Exercises

- Modify the previous program to re-prompt user until they give an answer that we accept (“Y” or “N”). If they cannot give a correct answer after three tries, make fun of them and re-prompt again :)

- Print out second powers of natural numbers until the results go over 100, 1000, …
  - 1, 4, 9,16, 25, ...

Recap of `for` loop

```
for val in sequence:
    Body of for
```

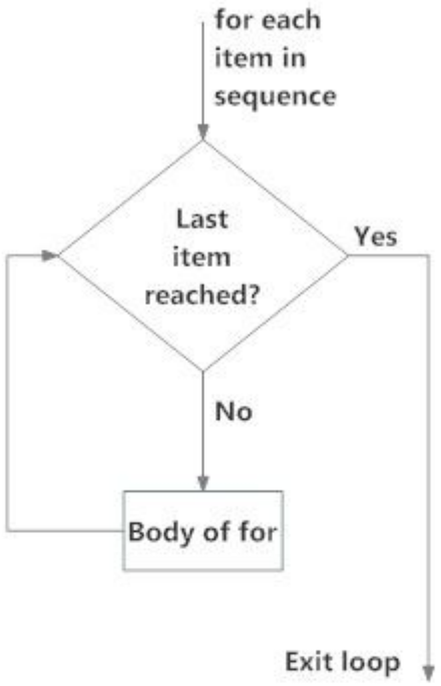


Fig: operation of for loop

Example

```
>>> numbers = [6, 5, 3, 8, 4, 2, 5, 4, 11]
>>> sum = 0
>>> for val in numbers:
...     sum = sum + val
>>> print("The sum is", sum)
48
```

range() function

```
range(start,stop,step size)
```

```
>>> for i in range(3):
... print(i)
0
1
2
```

```
>>> genre = ['pop', 'rock', 'jazz']
>>>
>>> for i in range(len(genre)):
... print("I like", genre[i])
I like pop
I like rock
I like jazz
```

```
>>> for i in range(3, 0, -1):
...     print(i)
3
2
1
```

```
>>> for i in range(-2, 2):
...     print(i)
-2
-1
0
1
```

for loop with else

```
>>> sum = 0
>>> for i in range(0, 10, 2):
...     print (i)
... else:
...     print("No items left")
0
2
4
6
8
No items left
```

*Generator Expressions

```
(what to do for variable in sequence)
```

```
>>> gen_obj = (x*3 for x in range(5))
>>> for val in gn_obj:
...     print(val)
0
3
6
9
12
```

```
>>> gen_obj = (x*2 for x in range(5))
>>> next(gen_obj)
0
```

```
>>> gen_obj = (x*2 for x in range(5))
>>> list(gen_obj)
[0,2,6,4,8]
```

*Generator Expressions Gotchas

```
>>> gen_obj = (x*3 for x in range(5))
>>> sum(gen_obj)
30
>>> sum(gen_obj)
0
```

```
>>> numbers = [1, 2, 3, 3, 7]
>>>
>>> gen_obj = (x**2 for x in numbers)
>>> 9 in gen_obj
True
>>> 9 in gen_obj
True
>>> 9 in gen_obj
False
>>> 4 in gen_obj
False
```

*Making for loops more pythonic

Instead of:

```
>>> my_items = ['a', 'b', 'c']
>>> for i in range(len(my_items)):
...     print (my_items[i])
```

Rather:

```
>>> my_items = ['a', 'b', 'c']
>>> for item in my_items:
...     print (item)
```

Or if the indexes are really needed:

```
>>> my_items = ['a', 'b', 'c']
>>> for i, item in enumerate(my_items):
...     print(f'{i}: {item}')
```

*Making for loops more pythonic

Instead of (and only if you really have to write a C-style loop):

```
>>> my_items = ['a', 'b', 'c', 'd', 'e', 'f']
>>> i = 0
>>> while i < len(my_items):
...     print (my_items[i])
...      i += 2
```

Rather:

```
>>> my_items = ['a', 'b', 'c']
>>> for i in range(0, len(my_items), 2):
...     print (my_items[i])
```

Exercises

- Write a program that draws a triangle formed of asterisks “*” on the screen. The size should be input from the user

  For input 5 it should output:

```
★
★★
★★★
★★★★
★★★★★
```

Exercises

- Write a program that draws a triangle formed of asterisks “*” on the screen. The size should be input from the user

For input 5 it should output:

```
★
★★
★★★
★★★★
★★★★★
```

Can you easily rotate the triangle? E.g.:

```
★★★★★
 ★★★★
  ★★★
   ★★
    ★
```

Or make it isosceles?

```
    ★
   ★★
  ★★★
 ★★★★
★★★★★
```

Exercises

- A permutation can be specified by an array P, where P[i] represents the location of the element at i in the permutation. For example, [2, 1, 0] represents the permutation where elements at the index 0 and 2 are swapped.

  Given an array and a permutation, apply the permutation to the array. For example, given the array ["a", "b", "c"] and the permutation [2, 1, 0], return ["c", "b", "a"].

  (Interview problem asked by Twitter)

More (and more difficult) exercises

- Spreadsheets often use this alphabetical encoding for its columns: "A", "B", "C", ..., "AA", "AB", ..., "ZZ", "AAA", "AAB", ...

  Given a column number, return its alphabetical column id.

  For example, given 1, return "A". Given 27, return "AA".

  (Interview problem asked by Dropbox)

More (and more difficult) exercises

- Given an array and a number k that's smaller than the length of the array, rotate the array by k elements to the right in-place.

(Interview problem asked by Amazon)

More (and more difficult) exercises

- Write a program that checks whether an integer is a palindrome. For example, 121 is a palindrome, as well as 888. 678 is not a palindrome. Do not convert the integer into a string.

(Interview problem asked by Palantir)