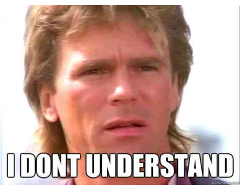
Controlling the flow of programs

Introducing the control keywords



How to deal with confusion

- Look at examples
- Practice, practice, practice

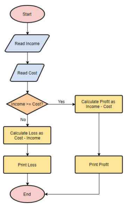
- This is the hardest part. But also the most rewarding.

So, far...

- Linear processing:
  - Step 1
  - Step 2
  - ...
  - Step n

Controlling the flow of your program

- Do something if (not)...
- Do something n times
- Do something for every element in a collection
- Do something as long as...

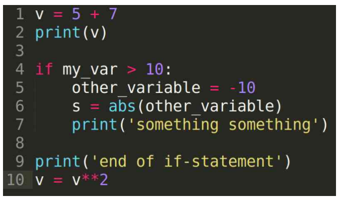


Most important flow control keywords

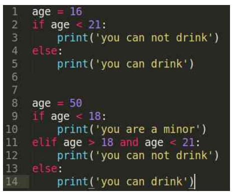
Keyword	Use	Example 1	Example 2	Example 3
if	**Do something if something is true (or false)**	If the value of a variable is larger than 5, print something to the screen.	If the value of a sensor is smaller than 15, stop the robot.	If two variables are equal, open and read a file.
for	**Do something for each item in a collection**	For each value in a list of values, print it to the screen.	For each sensor, check whether it detects an obstacle.	Repeat something a specific number of times.
while	**Do something as long as a given statement is true**	Drive the robot 10 cm forward as long as no obstacle is detected.	While I have not found 10 prime numbers, check whether the next number is a prime.	Do something forever.

The if statement

- Example of an if-statement:



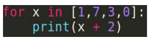
```python
v = 5 + 7
print(v)

if my_var > 10:
    other_variable = -10
    s = abs(other_variable)
    print('something something')

print('end of if-statement')
v = v**2
```



```python
age = 16
if age < 21:
    print('you can not drink')
else:
    print('you can drink')


age = 50
if age < 18:
    print('you are a minor')
elif age > 18 and age < 21:
    print('you can not drink')
else:
    print('you can drink')
```

For statement

- Read as for every element in a collection, do the following...
- Example:



```python
for x in [1,7,3,0]:
    print(x + 2)
```

- What happens:
  - Python takes first element from the list and assigns it to variable x (x = 1)
  - With x = 1, it executes the code
  - Python returns to the top and does the same thing for x = 7
  - And so on...

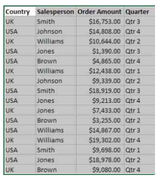
The for loop allows for all kinds of processing

- Analogy: for every box in my collection, put a stamp on it and ship...

- More examples: https://www.w3schools.com/python/python_for_loops.asp

Data analysis example

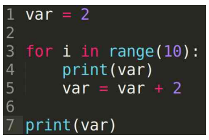
- For every name in a list: filter data, plot the mean value



Country	Salesperson	Order Amount	Quarter
UK	Smith	$16,753.00	Qtr 3
USA	Johnson	$14,808.00	Qtr 4
UK	Williams	$10,644.00	Qtr 2
USA	Jones	$1,390.00	Qtr 3
USA	Brown	$4,865.00	Qtr 4
UK	Williams	$12,438.00	Qtr 1
UK	Johnson	$9,339.00	Qtr 2
USA	Smith	$18,919.00	Qtr 3
USA	Jones	$9,213.00	Qtr 4
UK	Jones	$7,433.00	Qtr 1
USA	Brown	$3,255.00	Qtr 2
USA	Williams	$14,867.00	Qtr 3
UK	Williams	$19,302.00	Qtr 4
USA	Smith	$9,698.00	Qtr 1
USA	Jones	$18,978.00	Qtr 2
UK	Brown	$9,080.00	Qtr 4

The for-statement: special use

- The for loop can be used to repeat something a specific number of times



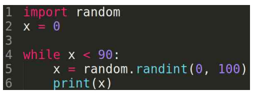
```python
var = 2

for i in range(10):
    print(var)
    var = var + 2

print(var)
```

Technical details

- Range creates as collection of ordered numbers.
  - range(10) --> 0 ,1 ,..., 8, 9
  - range(5, 10) --> 5 ,6 ,...., 8, 9
  - range(5, 10, 2) --> 5 ,7 , 9
- You can get the range as a list using `list(range(5, 10, 2))`
- Next, the for-loop iterates over the elements of those ranges.

The while statement

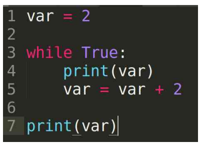
- Can be read as: as long as ... repeat the following ...
- Example: pick random values for x as long as x < 90:



```
import random
x = 0

while x < 90:
    x = random.randint(0, 100)
    print(x)
```

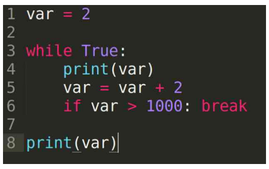
The while statement

- Using it to repeat something forever.



```
var = 2

while True:
    print(var)
    var = var + 2

print(var)
```

The while statement and the break statement



```
var = 2

while True:
    print(var)
    var = var + 2
    if var > 1000: break

print(var)
```

Summary

Keyword	Use	Example 1	Example 2	Example 3
if	**Do something if something is true (or false)**	If the value of a variable is larger than 5, print something to the screen.	If the value of a sensor is smaller than 15, stop the robot.	If two variables are equal, open and read a file.
for	**Do something for each item in a collection**	For each value in a list of values, print it to the screen.	For each sensor, check whether it detects an obstacle.	Repeat something a specific number of times.
while	**Do something as long as a given statement is true**	Drive the robot 10 cm forward as long as no obstacle is detected.	While I have not found 10 prime numbers, check whether the next number is a prime.	Do something forever.

Exercises

- exercises.md on GitHub