

# Variables, Assignments and If-statements

Julia Wippermann   Robin Horn   Kamran  
Vatankhah-Barazandeh   Leonard Frommelt

Apr 10, 2019



- 1 Basic Programming in Python
- 2 What are variables?
- 3 What is a data type?
- 4 How to write if-statements?
- 5 How to deal with errors?

## About me



Figure 1: A little bit about me.

# Feedback sessions and Practice sessions

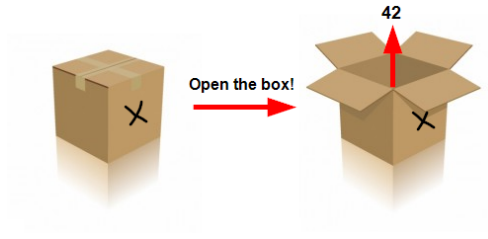
## Feedback sessions

- are great for getting individual feedback
- e.g. discuss new concepts
- sign up on StudIP

## Practice sessions

- you are welcome to come to the practice session any time
- you can collaborate with your fellow students
- we are there to give you guidance on the current homework

# What is a variable?



MVCodeClub (2019)

- you can “store” a value in a variable
- variables are placeholders
- values are the contents

## Assigning a value to a variable

- we use the = operator to assign a value to a variable

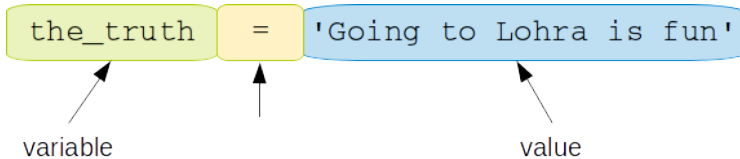


Figure 2: Variable assignment.

# Assigning a value to a variable

## Caution

= in code (denoting assignment)  
is **not** the same as  
= in math (denoting equality)

# Variables in Python

- variables are created when they are first assigned
- variables must be assigned before you use them



## Let's print the variable

```
the_truth = 'Going to Lohra is fun!'
```

```
print(the_truth)
```

**Output:**

```
Going to Lohra is fun!
```

## Variable assignment

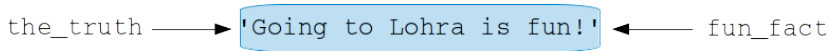
```
the_truth = 'Going to Lohra is fun!'
```

```
the_truth → 'Going to Lohra is fun!'
```

Figure 3: Variable assignment.

## Assigning a variable to another variable

```
the_truth = 'Going to Lohra is fun!'  
fun_fact = the_truth
```



the\_truth → 'Going to Lohra is fun!' ← fun\_fact

Figure 4: Variable assignment.

## Variable assignment

```
the_truth = 'Going to Lohra is fun!'
fun_fact = the_truth
fun_fact = 'Going to Trash is fun!'
```

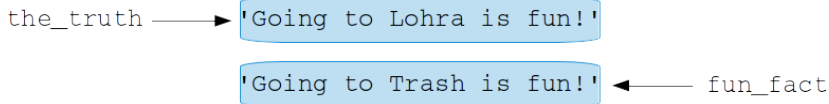


Figure 5: Variable assignment.

## Variable assignment

```
the_truth = 'Going to Lohra is fun!'
fun_fact = the_truth
fun_fact = 'Going to Trash is fun!'
the_truth = 'Learn Python for fun!'
```

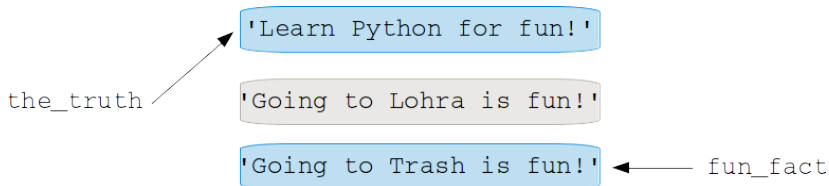


Figure 6: Variable assignment.

## Recap: string concatenation

```
who = 'Most Coxis think: '  
the_truth = 'Going to Lohra is fun!'  
  
# intended output on the command line:  
# Most Coxis think: 'Going to Lohra is fun!'  
  
# your code goes here
```

## Recap: string concatenation

```
who = 'Most Coxis think: '  
the_truth = 'Going to Lohra is fun!'  
  
print(who + " " + the_truth + " ")
```

### Output:

```
Most Coxis think: 'Going to Lohra is fun!'
```

# print()

```
answer = 'We are learning'  
language = 'Python.'
```

```
print(answer + " " + language)  
print()  
print(answer, language)
```

**Output:**

```
We are learning Python.
```

```
We are learning Python.
```



# Variable naming rules in Python

- variable names
  - can be of any length,
  - can contain uppercase and lowercase letters (A-Z, a-z),
  - digits (0-9),
  - and the underscore character (\_).

## Caution

The first character of a variable name cannot be a digit!

# Conventions

- choose meaningful variable names
- do not begin variable names with a capital letter
  - capital letters are reserved for other things (convention)
  - more about this in the next weeks

## Conventions

```
# do not do this  
var_1 = '12.04.2019'  
x = '14.04.2019'  
  
print('Lohra is from ' + var_1 + ' to ' + x)
```

### Output:

```
Lohra is from 12.04.2019 to 14.04.2019
```

## Conventions

```
# do this  
# give your variables descriptive names  
start_date = '12.04.2019'  
end_date = '14.04.2019'  
  
print('Lohra is from ' + start_date  
      + ' to ' + end_date)
```

### Output:

```
Lohra is from 12.04.2019 to 14.04.2019
```

- make long variables readable by using snake\_case
- snake\_case means that each word is separated by an underscore

```
total_number_of_participants_lohra = 958
```

# What is a data type?

- defines how the value of the variable is stored
- defines which operations are valid for the variable

## Data types in Python

- the data type of a variable is interpreted based on the value that is assigned to the variable
- there is no explicit declaration of a data type for a variable
- use the `type()` function to check the type of a variable

	Type	Description	Example
Integer	int	integer of arbitrary magnitude	<code>x = 1</code>
Float	float	floating-point number	<code>x = 1.0</code>
String	str	character string	<code>x = 'one'</code>
Boolean	bool	truth value	<code>x = True</code>

```
# the variable is a string.  
ultimate_answer = 'forty-two'  
print('The answer is', ultimate_answer)  
print('The type is', type(ultimate_answer))
```

## Output:

```
The answer is forty-two  
The type is <class 'str'>
```



```
# the variable is an integer
ultimate_answer = 42
print('The answer is', ultimate_answer)
print('The type is', type(ultimate_answer))
```

### Output:

```
The answer is 42
The type is <class 'int'>
```

```
# the variable is a float
ultimate_answer = 42 + 0.0
print('The answer is', ultimate_answer)
print('The type is', type(ultimate_answer))
```

## Output:

```
The answer is 42.0
The type is <class 'float'>
```

## Operations on data types

Subtraction is a valid operation on two integers.

```
to_be_paid = 12  
given_cash = 15
```

```
money = given_cash - to_be_paid  
print('The cashier will give you', money, 'Euros.')
```

**Output:**

```
The cashier will give you 3 Euros.
```

## Operations on data types

Subtraction is an invalid operation between an integer and a string.

```
to_be_paid = '12'  
given_cash = 15
```

```
money = given_cash - to_be_paid
```

### Output:

```
Traceback (most recent call last):  
  File "<string>", line 4, in <module>  
TypeError: unsupported operand type(s) for -: 'int'  
and 'str'
```

## Casting

```
to_be_paid = '12'  
given_cash = 15
```

```
money = given_cash - int(to_be_paid)  
print('The cashier will give you', money, 'Euros.')
```

### Output:

```
The cashier will give you 3 Euros.
```

## The Boolean data type - True or False

**With a Boolean you express truth values:**

something is True  
(which corresponds to 1)

or

something is False  
(which corresponds to 0)

```
light_on = True  
light_of = False
```

## Checking if a statement is True or False

```
>>> 10 == 10
```

```
True
```

```
>>> 1 > 2
```

```
False
```

```
>>> 2 != 2
```

```
False
```

```
age = 14
```

```
>>> 12 < age < 20
```

```
True
```

## How to make conditional statements?

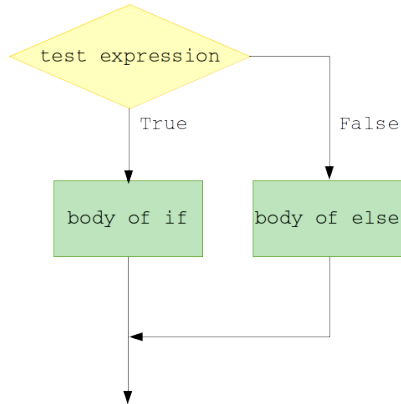


Figure 7: Control Flow of an if-statement



## If - statements

```
age = 14
```

```
if age == 14:  
    print('You are 14 years old!')
```

```
if 13 <= age <= 19 :  
    print('You are a teenager.')
```

```
else:  
    print('You are not a teenager.')
```

```
age = 14

if age == 14:
    print('You are 14 years old!')

if 13 <= age <= 19 :
    print('You are a teenager.')
else:
    print('You are not a teenager.')
```

### Output:

```
You are 14 years old!
You are a teenager.
```

# Errors

Everyone makes mistakes when writing code

You just need to know how to find them and solve them. → Learn how to understand error messages

## Example error

```
# this is a valid Python statement  
goodn8 = 'sleep well'
```

```
# this is an invalid Python statement  
4u = 'for you'
```

## Example traceback

```
# this is a valid Python statement  
goodn8 = 'sleep well'
```

```
# this is an invalid Python statement  
4u = 'for you'
```

### Output:

```
File "<string>", line 5  
    4u = 'for you'  
    ^  
SyntaxError: invalid syntax
```

## Reading the traceback

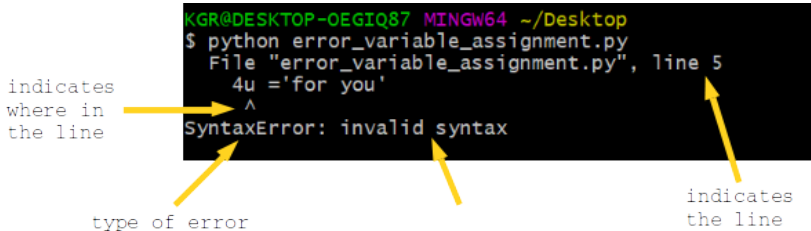


Figure 8: Traceback

# SyntaxError

- your code cannot be interpreted
  - similar to making an ‘orthographic mistake’
- did you only use valid Python statements?

## Different types of errors

- `SyntaxError`
- `NameError`
- `ValueError`
- `TypeError`

You will learn about these type of errors on the exercise sheet



## Tips for the homework

- to understand what code does
  - try to run it → what is the output/error?
  - try to change some variable → how does the output change?
- using a search engine is **no** cheating!
- collaborate with your peers!





See you tomorrow!

I hope to see you in the practice session tomorrow!

# References

MVCodeClub. 2019. “Intro to Scratch Stories 6: Rock-Paper-Scissors.” <https://www.mvcode.com/lessons/intro-to-scratch-stories-6-rock-paper-scissors>.