

Types

Dr. Giuseppe  
Maggiore

Introduction

Python type  
system basics

Type  
restrictions

Assignment

# Types

Dr. Giuseppe Maggiore

Hogeschool Rotterdam  
Rotterdam, Netherlands

Types

Dr. Giuseppe  
Maggiore

Introduction

Python type  
system basics

Type  
restrictions

Assignment

## Lecture topics

- We introduce the Python type system
- Numbers
- Boolean values
- Arithmetic and boolean expressions

Types

Dr. Giuseppe  
Maggiore

Introduction

Python type  
system basics

Type  
restrictions

Assignment

## Introduction

- Is everything an integer number?
- Yes and no

## Everything is an integer number

- For the CPU everything is a string of bits
- So yes, everything is (*almost*<sup>a</sup>) an integer number
- Complex data structures like a GUI, a 3D model, a picture, etc. are made up of collections of numbers

---

<sup>a</sup>also floats are recognized by the CPU

Types

Dr. Giuseppe  
Maggiore

Introduction

Python type  
system basics

Type  
restrictions

Assignment

## Everything is an integer number

- Low-level languages expose this view
- Everything is encoded with numbers
- It can become quite messy

## Not everything is an integer number

- For the programmer, there exist different kinds of values
- So common and useful that Python offers them out of the box
- Even if the CPU does not manipulate them directly

## Kinds of values

- Python has a **type system**
- Variables have different **data types**, often shortened to **types**
  - Integer numbers
  - Rational (floating point) numbers
  - Boolean truth values
  - Strings of text

## Integers

- Numbers without dot<sup>a</sup>
  - 0
  - 100
  - -500

---

<sup>a</sup>comma in Dutch



## Integers

- Typical arithmetic operations on numbers (**not in Python 3**)
  - $3 + 5 = 8$
  - $5 / 2 = 2$
  - $40 * 5 = 200$

## Floating points

- Numbers with dot<sup>a</sup>
  - 0.0
  - 2.5
  - 10.0e3
  - 3.1e-5
  - -.1e-5

---

<sup>a</sup>comma in Dutch

## The scientific notation

- 0.00001 is annoying to write
- we can write `1.e-5` instead
- the sign `e-N` means *add N zeros right after the dot*

Types

Dr. Giuseppe  
Maggiore

Introduction

Python type  
system basics

Type  
restrictions

Assignment

## The scientific notation

- 1000000.0 is annoying to write
- we can write 1.e6 instead
- the sign eN means *add N zeros right before the dot*

## Floating points

- Typical arithmetic operations on numbers
  - $5.0 / 2.0 = ?$
  - $10.0e3 / 0.1 = ?$
  - $3.1e-5 + 1.0e5 = ?$
- **Can you guess the results?**

## Floating points

- Typical arithmetic operations on numbers
  - $5.0 / 2.0 = ?$
  - $10.0e3 / 0.1 = ?$
  - $3.1e-5 + 1.0e5 = ?$
- **Can you guess the results?**
  - $5.0 / 2.0 = 2.5$
  - $10.0e3 / 0.1 = 10.0e4$
  - $3.1e-5 + 1.0e5 = 100000.000031$

Types

Dr. Giuseppe  
Maggiore

Introduction

Python type  
system basics

Type  
restrictions

Assignment

## Conversion to and from floating point

- Integers can be converted to floating points with `float(n)`
- Floating points can be converted to integers with `int(n)`

## Conversion to and from floating point

- Given the following expressions:
  - `int(2.5) = ?`
  - `float(3) = ?`
- **Can you guess the results?**



## Conversion to and from floating point

- Given the following expressions:
  - `int(2.5) = ?`
  - `float(3) = ?`
- **Can you guess the results?**
  - `int(2.5) = 2`
  - `float(3) = 3.0`

## Types

Dr. Giuseppe  
Maggiore

## Introduction

## Python type system basics

## Type restrictions

## Assignment

### Conversion to and from floating point

- Floating points can lose their decimal values
- They stay float's, but always end in .0
- `math.floor(n)` truncates the tail
- `math.ceil(n)` fills the tail and increases to the next unit

## Conversion to and from floating point

- Given the following expressions:
  - `floor(2.5) = ?`
  - `ceil(2.5) = ?`
- **Can you guess the results?**

## Conversion to and from floating point

- Given the following expressions:
  - `floor(2.5) = ?`
  - `ceil(2.5) = ?`
- **Can you guess the results?**
  - `floor(2.5) = 2.0`
  - `ceil(2.5) = 3.0`

Types

Dr. Giuseppe  
Maggiore

Introduction

Python type  
system basics

Type  
restrictions

Assignment

## Conversion to and from floating point

- Some conversions happen automatically
- Python operations try to preserve information
- $5 / 2.0 = 2.5$ , and 5 is converted to 5.0 right before the division

## Types

Dr. Giuseppe  
Maggiore

## Introduction

## Python type system basics

## Type restrictions

## Assignment

### Python 3 integer division

- The new version of Python has a new integer division: it always converts to float
- It is **very different** from most other programming languages
- $5 / 2 = 2.5$

Types

Dr. Giuseppe  
Maggiore

Introduction

Python type  
system basics

Type  
restrictions

Assignment

## Python 3 integer division

- Traditional integer division is now “//”
- $5 // 2 = 2$

Types

Dr. Giuseppe  
Maggiore

Introduction

Python type  
system basics

Type  
restrictions

Assignment

## Boolean values

- Truth values
- True, False
- “Answers to yes/no questions”



## Boolean values

- Logical operators on truth values
- Compose the answers to multiple questions
- Both questions in parallel:
  - Do you like chocolate? Yes.
  - Do you like vanilla? Yes.
  - **Do you like chocolate and vanilla? Yes.**
- Both questions concurrently:
  - Do you like chocolate? Yes.
  - Do you like vanilla? No.
  - **Do you like chocolate or vanilla? Yes.**
- Turn questions around:
  - Do you like chocolate? Yes.
  - **Do you dislike chocolate? No.**

## Boolean values

- Logical operators take one or two input
- This means that we have no more than four possible combinations of input values
- Since the inputs are so few, we can enumerate all combinations
- This is done with a **truth table**

## Boolean values

- Truth tables enumerate all input values and the result of

	A	B	$(A \odot B)$
	<i>True</i>	<i>True</i>	...
their operator	<i>True</i>	<i>False</i>	...
	<i>False</i>	<i>True</i>	...
	<i>False</i>	<i>False</i>	...

## Boolean values

- Logical operators on truth values

- & for and

A	B	( A & B )
<i>True</i>	<i>True</i>	<i>True</i>
<i>True</i>	<i>False</i>	<i>False</i>
<i>False</i>	<i>True</i>	<i>False</i>
<i>False</i>	<i>False</i>	<i>False</i>

## Boolean values

- Logical operators on truth values

- | for or

A	B	( A   B )
<i>True</i>	<i>True</i>	<i>True True</i>
<i>True</i>	<i>False</i>	<i>True True</i>
<i>False</i>	<i>True</i>	<i>False True</i>
<i>False</i>	<i>False</i>	<i>False False</i>

Types

Dr. Giuseppe  
Maggiore

Introduction

Python type  
system basics

Type  
restrictions

Assignment

## Boolean values

- Logical operators on truth values

- `not`

A	<i>not</i>	A
<i>True</i>	<i>False</i>	<i>True</i>
<i>False</i>	<i>True</i>	<i>False</i>

Types

Dr. Giuseppe  
Maggiore

Introduction

Python type  
system basics

Type  
restrictions

Assignment

## Boolean values

- Comparison operators on numeric values
  - >
  - <
  - ==
  - >=
  - <=

## Boolean values

- Given the following expressions:
  - $5.0 > 2.0 = ?$
  - $(3 > 4) \mid (5 == (3 + 2)) = ?$
  - $\text{True} \ \& \ \text{False} = ?$
- **Can you guess the results?**



## Boolean values

- Given the following expressions:
  - $5.0 > 2.0 = ?$
  - $(3 > 4) \mid (5 == (3 + 2)) = ?$
  - $\text{True} \& \text{False} = ?$
- **Can you guess the results?**
  - $5.0 > 2.0 = \text{True}$
  - $(3 > 4) \mid (5 == (3 + 2)) = \text{True}$
  - $\text{True} \& \text{False} = \text{False}$

Types

Dr. Giuseppe  
Maggiore

Introduction

Python type  
system basics

Type  
restrictions

Assignment

## String values

- Text
- "Hello!", "Hello world!", "", ...

## Types

Dr. Giuseppe  
Maggiore

## Introduction

## Python type system basics

## Type restrictions

## Assignment

### String values

- String literals are sequences of characters, on a single line, between double " or single ' quotes
- Some characters do not fit this description
- We need special markings for such characters
- These special markings are called *escape characters*

## Types

Dr. Giuseppe  
Maggiore

Introduction

Python type  
system basics

Type  
restrictions

Assignment

## String values

- `\'` for single quote
- `\"` for double quote
- `\a` for ASCII Bell (BEL)
- `\b` for ASCII Backspace (BS)
- `\f` for ASCII Formfeed (FF)
- `\n` for ASCII Linefeed (LF)
- `\r` for ASCII Carriage Return (CR)
- `\t` for ASCII Horizontal Tab (TAB)
- `\v` for ASCII Vertical Tab (VT)

## String values

- `"Hello\n world"` is a string on two lines
- `"Hello\n world\n of Python"` is a string on three lines
- ...

## String values

- The most common operator is string concatenation
- `"Hello" + "\n" + "world" + "\n" + "on" + "\n" + "different" + "\n" + "lines"`

## Operations, types, and restrictions

- Not all operations are allowed on all possible variable types
  - Some operations are allowed (integer addition)
  - Some operations are not allowed (string division)
  - Some operations change meaning (addition of integers versus concatenation of strings)

## Operations, types, and restrictions

- Examples of allowed operators
  - Addition, subtraction, division, multiplication, etc. between numbers
  - Concatenation between strings
  - Multiplication of strings and integers
  - Arithmetic comparison between numbers or strings
  - Conjunction, disjunction, negation<sup>a</sup> between booleans
  - Treating integers as booleans (1=True, 0=False)
  - Treating strings as booleans (anything else=True, ""=False)

---

<sup>a</sup>and, or, not



## Operations, types, and restrictions

- Examples of not-allowed operators
  - Most arithmetic operations on strings and non-strings  
(`"Hello" + True`)
  - Most boolean operations on strings and non-strings  
(`"Hello" & True`)

## Not-allowed operators generate *type errors*

```
Traceback (most recent call last):  
  File "C:\Users\Giuseppe\Desktop\DEV_I\samples\  
    DEV_I_samples.py", line 8, in <module>  
      print("Oh_noes,_a_bug!" + 4)  
TypeError: cannot concatenate 'str' and 'int'  
objects
```

## Operations, types, and restrictions

- Variables may change type in Python
- An integer variable becomes later on a string variable
- This is allowed, but dangerous
- A variable should never lose reasonable meaning
- Many type errors stem from *changes in meaning*, connected with *changes in type* of a variable

Types

Dr. Giuseppe  
Maggiore

Introduction

Python type  
system basics

Type  
restrictions

Assignment

## Introduction

- Multiple operators in a single expression are ambiguous
- For example: `not True | True`
  - `(not True) | True = False | True = True`
  - `not (True | True) = not True = False`

Types

Dr. Giuseppe  
Maggiore

Introduction

Python type  
system basics

Type  
restrictions

Assignment

## Introduction

- Python defines which operators are evaluated first, and which later
- Removes ambiguity
- Makes parentheses not required
  - Still, might remain better for readability

## Types

Dr. Giuseppe  
Maggiore

Introduction

Python type  
system basics

Type  
restrictions

Assignment

## Introduction

- From lowest precedence (least binding) to highest precedence (most binding)

## Introduction

- From lowest precedence (least binding) to highest precedence (most binding)
- Some operators share the same precedence
  - $+$ ,  $-$
  - $*$ ,  $/$

## Introduction

- From lowest precedence (least binding) to highest precedence (most binding)
- Some operators share the same precedence
  - $+$ ,  $-$
  - $*$ ,  $/$
- Unless the syntax is explicitly given (example by mean of parenthesis)



## Types

Dr. Giuseppe  
Maggiore

## Introduction

Python type  
system basics

Type  
restrictions

## Assignment

### Introduction

- From lowest precedence (least binding) to highest precedence (most binding)
- Some operators share the same precedence
  - $+$ ,  $-$
  - $*$ ,  $/$
- Unless the syntax is explicitly given (example by mean of parenthesis)
- A complete table of precedence can be found on <https://docs.python.org/2/reference/expressions.html#operator-precedence>

Types

Dr. Giuseppe  
Maggiore

Introduction

Python type  
system basics

Type  
restrictions

Assignment

## Introduction

- Example: integer operations in Python like  $*$  and  $/$  have higher precedence than  $+$  and  $-$
- $1 + 4 * 2 = 9$

## Introduction

- Example: integer operations in Python like  $*$  and  $/$  have higher precedence than  $+$  and  $-$
- $1 + 4 * 2 = 9$
- Use parenthesis to group expressions
- $(1 + 4) * 2 = 10$

Types

Dr. Giuseppe  
Maggiore

Introduction

Python type  
system basics

Type  
restrictions

Assignment

## Introduction

- Given the following expressions what are the results:
  - $(20 + 10) * 15 / 5 = ?$
  - $((20 + 10) * 15) / 5 = ?$
  - $20 + (10 * 15) / 5 = ?$

## Introduction

- Given the following expressions what are the results:
  - $(20 + 10) * 15 / 5 = ?$
  - $((20 + 10) * 15) / 5 = ?$
  - $20 + (10 * 15) / 5 = ?$
- Results:
  - $(20 + 10) * 15 / 5 = 90$
  - $((20 + 10) * 15) / 5 = 90$
  - $20 + (10 * 15) / 5 = 50$

## Types

Dr. Giuseppe  
Maggiore

Introduction

Python type  
system basics

Type  
restrictions

Assignment

## Instructions

- Split into four groups.
- Use the data types you saw in this lesson to model an RPG character in a Python program.
- Example: health, team color, ...
- Make sure the program runs without errors.
- Draw on a sheet what the soldier should look like.
- Hand over the code to another group and make them draw the soldier.
- If the pictures are the same then you have succeeded, otherwise adjust your code.

## Types

Dr. Giuseppe  
Maggiore

Introduction

Python type  
system basics

Type  
restrictions

Assignment

## Hand-in

- Write your names and student numbers on your sheets
- Hand them in
- *They may be used at your oral check* in the form of questions such as “how would you rewrite this after the course”

Types

Dr. Giuseppe  
Maggiore

Introduction

Python type  
system basics

Type  
restrictions

Assignment

The best of luck, and thanks for the  
attention!