Elan Python course
Session 1

Mart Lubbers: `mart.lubbers@mpi.nl`
Fransisco Torreira: `francisco.torreira@mpi.nl`

2014-09-30

Table of contents

Introduction

Variables
  Assignment
  Printing
  Numbers and Operators
  Booleans
  Strings
  Lists and dicts
  Objects and methods

Control structures
  If statements
  {For,While}loops

Excercise
  File i/o
  Excercise

Installing python

Installing python

Installation instructions

`https://developers.google.com/edu/python/set-up`

Installing python

Installation instructions

`https://developers.google.com/edu/python/set-up`

Notepad++

On windows, please use Notepad++

Installing python

Installation instructions

`https://developers.google.com/edu/python/set-up`

Notepad++

On windows, please use Notepad++

Interpreter

```
Python 2.7.3 (default, Apr 14 2012, 08:58:41) [GCC] on linux2
Type ”help”, ”copyright”, ”credits” or ”license” for more information.
>>>
```

Working with python

Working with python

What is a script?

```
a = ’Mart’
b = ’Hello world’
print ’Hello my name is {}’.format(a)
print b
```

Working with python

What is a script?

```python
a = 'Mart'
b = 'Hello world'
print 'Hello my name is {}'.format(a)
print b
```

Command line

```
C:\pythoncourse> python hello.py
$ ~/pythoncourse> python hello.py
```

Table of contents

- Introduction
- Variables
  - Assignment
  - Printing
  - Numbers and Operators
  - Booleans
  - Strings
  - Lists and dicts
  - Objects and methods
- Control structures
  - If statements
  - {For,While}loops
- Excercise
  - File i/o
  - Excercise

Assignment

Assignment

Save the value for later

a = 6
b = 3.14

Assignment

Save the value for later

$a = 6$
$b = 3.14$

And use the value for other purposes

$a = 6$
$b = a/2$

Printing

Printing

Printing example

```
a = 5
b = ’Hello world!’
print a
print b
print a, b
print a, 3.14, b
```

Printing

Printing example

```
a = 5
b = ’Hello world!’
print a
print b
print a, b
print a, 3.14, b
```

Printing example output

```
5
Hello world!
5 Hello world!
5 3.14 Hello world!
```

Numbers

Numbers

Types

- int, long, complex
- float
- bool

Numbers

Types

- int, long, complex
- float
- bool

Symbols

Operator	Expression	Answer
Addition	$5 + 7$	13
Subtraction	$5 - 7$	-2
Division	$7/5$	1
Multiplication	$5 * 7$	35
Remainder	$7\%5$	2
Power	$7 * *5$	16807

Numbers

Types

- int, long, complex
- float
- bool

Symbols

Operator	Expression	Answer
Addition	$5+7$	13
Subtraction	$5-7$	-2
Division	$7/5$	1
Multiplication	$5*7$	35
Remainder	$7\%5$	2
Power	$7**5$	16807

Note

Precision

Booleans

Booleans

Boolean operators

a	b	op	a op b
True	True	and	True
True	False	and	False
False	True	and	False
False	False	and	False
True	True	or	True
True	False	or	True
False	True	or	True
False	False	or	False

Booleans

Boolean operators

a	b	op	a op b
True	True	and	True
True	False	and	False
False	True	and	False
False	False	and	False
True	True	or	True
True	False	or	True
False	True	or	True
False	False	or	False

Parenthesis

not((True **and** False)**or** True)

$3/((1+4)*3)$

Booleans

Boolean operators

a	b	op	a op b
True	True	and	True
True	False	and	False
False	True	and	False
False	False	and	False
True	True	or	True
True	False	or	True
False	True	or	True
False	False	or	False

Parenthesis

not((True **and** False)**or** True)

$3/((1+4)*3)$

Comparison operators

Equal	==
Not equal	! =
Greater	>
Smaller	<
Greater or equal	<=
Smaller or equal	>=

Strings

Strings

Conventions

```
s = 'I am a single line string with single quotes and can use double quotes(") freely'
s = "I use double quotes so that I don't have to escape single quotes(')"
s = 'When I want to use single quotes here I have to escae them(\')'
s = "Same for double(\")"
s = """Or you can make multi line strings
And it goes on
And on
And on"""
```

Lists

Lists

Sequence of items

```
a = [3, 1, 4, 1, 5]
print a[0]
print a[3]
print a[-1]
print a[1:3]
print a[:3]
print a[1:]
```

Lists

Sequence of items

```
a = [3, 1, 4, 1, 5]
print a[0]
print a[3]
print a[-1]
print a[1:3]
print a[:3]
print a[1:]
```

```
0
1
5
[1, 4]
[3, 1, 4]
[1, 4, 1, 5]
```

Dictionaries

Dictionaries

Mapping of items

```
a = {'five': 5, 3: 'three', 1: 'one', 2: 'two', 4: 'four'}
print a[3]
print a['five']
a['six-seven'] = '67'
print a['six-seven']
```

```
'three'
5
'67'
```

Objects

Objects

Example objects

- int, float, long, complex
- str, unicode
- dict, list
- file

Methods

Methods

Example dictionary methods

```
a = {’five’: 5, 3: ’three’, 1: ’one’, 2: ’two’, 4: ’four’}
print a.keys()
print a.values()
print a.items()
```

```
[1, 2, 3, ’five’, 4]
[’one’, ’two’, ’three’, 5, ’four’]
[(1, ’one’), (2, ’two’), (3, ’three’), (’five’, 5), (4, ’four’)]
```

Methods

Example dictionary methods

```
a = {'five’: 5, 3: 'three’, 1: 'one’, 2: 'two’, 4: 'four’}
print a.keys()
print a.values()
print a.items()
```

```
[1, 2, 3, 'five’, 4]
['one’, 'two’, 'three’, 5, 'four’]
[(1, 'one’), (2, 'two’), (3, 'three’), ('five’, 5), (4, 'four’)]
```

`https://docs.python.org/2/library/stdtypes.html#mapping-types-dict`

Example string methods

```
s1 = ”Hello world!”
print s1.lower()
print s1.lower().capitalize()
print s1.replace(’!’, ’?’)
print ’test’.join([’a’, ’b’, ’c’])
```

```
’hello world!’
’Hello world!’
’Hello world?’
’atestbtestc’
```

Example string methods

```
s1 = "Hello world!"
print s1.lower()
print s1.lower().capitalize()
print s1.replace('!', '?')
print 'test'.join(['a', 'b', 'c'])
```

```
'hello world!'
'Hello world!'
'Hello world?'
'atestbtestc'
```

`https://docs.python.org/2/library/stdtypes.html#sequence-types-str-unicode-list-tuple-bytearray-buffer-xrange`

Table of contents

- Introduction
- Variables
  - Assignment
  - Printing
  - Numbers and Operators
  - Booleans
  - Strings
  - Lists and dicts
  - Objects and methods
- Control structures
  - If statements
  - {For,While}loops
- Excercise
  - File i/o
  - Excercise

Only do this when that is the case

Only do this when that is the case

```
a = 4
if a < 5:
   print a, ' is smaller then 5'
elif a >= 5 and a<10:
   print a, ' is bigger then 5 and smaller then 10'
else:
   print a, ' is bigger then 10'
```

Do this until I say you can stop

Do this until I say you can stop

While

```
a = 1
while a < 10:
   print 'a: ', a
   a = a + 1
```

For

```
a = [3, 1, 4, 1, 5]
for x in a:
   print 'x: ', x
```

Nesting

Nesting

```
# Print if a number is even for all numbers from 5 to 0, 6 to 0 and 7 to 0
a = [5, 6, 7]
for x in a:
    while x > 0:
        if x % 2 == 0:
            print 'x: ', x, ' is even'
        else:
            print 'x: ', x, ' is odd'
        x = x - 1
```

Table of contents

- Introduction
- Variables
  - Assignment
  - Printing
  - Numbers and Operators
  - Booleans
  - Strings
  - Lists and dicts
  - Objects and methods
- Control structures
  - If statements
  - {For,While}loops
- Excercise
  - File i/o
  - Excercise

Reading a file

Reading a file

Old (unsafe) method

```
file_in = open('datafile.txt', 'r')
for line in file_in:
    print 'this is the current line: ', line
file_in.close()
```

Reading a file

Old (unsafe) method

```
file_in = open('datafile.txt', 'r')
for line in file_in:
    print 'this is the current line: ', line
file_in.close()
```

New (safe) method

```
with open('datafile.txt', 'r') as file_in:
    for line in file_in:
        print 'this is the current line: ', line
```

Reading a file

Old (unsafe) method

```
file_in = open('datafile.txt', 'r')
for line in file_in:
    print 'this is the current line: ', line
file_in.close()
```

New (safe) method

```
with open('datafile.txt', 'r') as file_in:
    for line in file_in:
        print 'this is the current line: ', line
```

Other functions

```
file_in.readline() # Read single line
file_in.read() # Read everything
file_in.read(8) # Read 8 bytes/characters
file_in.readlines() # Read everything and put it in a list, one line per item
```

Writing a file, now only the new method

Writing a file, now only the new method

New file

```
with open(’datafile.txt’, ’w’) as file_in:
    file_in.write(’some data’)
```

Writing a file, now only the new method

New file

```
with open('datafile.txt', 'w') as file_in:
    file_in.write('some data')
```

Append to existing file

```
with open('datafile.txt', 'a') as file_in:
    file_in.write('some data')
```

Practical example, reading tab separated file

Practical example, reading tab separated file

data.txt

```
id name age
0 john 12
1 joe 11
2 jan 19
3 jacob 16
4 jessy 18
```

Practical example, reading tab separated file

data.txt

```
id name age
0 john 12
1 joe 11
2 jan 19
3 jacob 16
4 jessy 18
```

Reading it into a dictionary

```
data = []
with open(’data.txt’, ’r’) as in:
   hds = in.readline()
   hds = hds.strip()
   hds = hds.split(’\t’)
   for line in in:
      line = line.strip().split(’\t’)
      data.append({hds[0]: line[0], hds[1]: line[1], hds[2]: line[2]})
print data[2][’name’]
print data[1][’age’]
```

Writing a tab separated file

Writing a tab separated file

```
hds = data.keys()
with open(’data2.txt’, ’w’) as out:
# Verbose method:
    out.write('\t’.join(hds))
    for line in data:
        out.write(line[’id’] + '\t’ + line[’name’] + '\t’ + line[’age’] + '\n’)
```

Excercise

Excercise

1. Alice in Wonderland
2. Frequency table