# Foundation

## Variable and value

Value literal

Variable

* Type
* Design time and run time

### System variables

## Statements

### Comments

### Expressions

### Operators

+ - \* / \*\*

## Common functions

dir()

print

# Primitive types

## String

### Literal

Quotations

>>> 'spam eggs' # single quotes

'spam eggs'

>>> " spam eggs" # double quotes

'spam eggs'

>>> 'doesn\'t' # use \' to escape the single quote...

"doesn't"

>>> "doesn't" # ...or use double quotes instead

"doesn't"

>>> '"Yes," he said.'

'"Yes," he said.'

>>> "\"Yes,\" he said."

'"Yes," he said.'

>>> '"Isn\'t," she said.'

'"Isn\'t," she said.'

>>> print('"Isn\'t," she said.')

"Isn't," she said.

Special characters

>>> s = 'First line.\nSecond line.' # \n means newline

>>> s # without print(), \n is included in the output

'First line.\nSecond line.'

>>> print(s) # with print(), \n produces a new line

First line.

Second line.

Escaping

>>> print('C:\some\name') # here \n means newline!

C:\some

ame

>>> print(r'C:\some\name') # note the r before the quote

C:\some\name

Multiple lines

String literals can span multiple lines. One way is using triple-quotes: """...""" or '''...'''.

print("""\

Usage: thingy [OPTIONS]

-h Display this usage message

-H hostname Hostname to connect to

""")

### Concatenation

### String and characters

### String formatting

## Numbers

>>> 2 + 2

4

>>> 50 - 5\*6

20

>>> (50 - 5\*6) / 4

5.0

>>> 8 / 5 # division always returns a floating point number

1.6

>>> 17 / 3 # classic division returns a float

5.666666666666667

>>> 17 // 3 # floor division discards the fractional part

5

>>> 17 % 3 # the % operator returns the remainder of the division

2

>>> 5 \* 3 + 2 # result \* divisor + remainder

17

>>> 5 \*\* 2 # 5 squared

25

>>> 2 \*\* 7 # 2 to the power of 7

128

### Integer

### Float

### random

### Number formatting

### Converting

str()

int()

## Text

### Unicode and Encoding

### Regular expression

## Date and time

### datetime module

### time module

>>> import time #time is a module

>>> time.localtime()

time.struct\_time(tm\_year=2017, tm\_mon=1, tm\_mday=12, tm\_hour=13, tm\_min=25, tm\_sec=32, tm\_wday=3, tm\_yday=12, tm\_isdst=0)

>>> time.asctime()

'Thu Jan 12 13:26:56 2017'

# Programming Logic

## Equality

## Comparison

## Control flow

### if

### for

numbers

range(gt\_eq, lt, step)

collection

### while

### exception

## Function

### Input

#### Parameter

#### Arguments

### Output

#### Void

#### Return

# Collections

## List

## Dictionary

## Set

## Tuple

### for

>>> table = {'Sjoerd': 4127, 'Jack': 4098, 'Dcab': 7678}

>>> for name, phone in table.items():

... print('{0:10} ==> {1:10d}'.format(name, phone))

...

Jack ==> 4098

Dcab ==> 7678

Sjoerd ==> 4127

# Modules

## Package

# Package management

## pip

install

pip install package\_name

pip install SomePackage==1.0.4 # specific version

pip install "SomePackage>=1.0.4" # minimum version

--trusted-host

There was a problem confirming the ssl certificate: …

pip install --trusted-host pypi.python.org pylint

pip install --trusted-host pypi.python.org pygame

--upgrade

pip install --trusted-host pypi.python.org --upgrade pip  
pip install --trusted-host pypi.python.org --upgrade pylint

pip install --trusted-host pypi.python.org --upgrade --force-reinstall pip

pip install --trusted-host pypi.python.org --upgrade setuptools

--force-reinstall

pip install --trusted-host pypi.python.org --upgrade --force-reinstall pylint

Common tasks:

<http://stackoverflow.com/questions/2720014/upgrading-all-packages-with-pip>

## Useful packages

pylint

https://www.pylint.org/

pip install --trusted-host pypi.python.org --upgrade --force-reinstall pylint

# Resources

## Official

<https://docs.python.org/3/>

## Community

<http://effbot.org>

<http://inventwithpython.com/bookshelf/>

<http://learnpythonthehardway.org/>