Python

Leonardo Saurwein - lsaurwein@student.ethz.ch

Version: April 8, 2022
Python 3.10

General

Operators

Define a number variable:

```
i = 0 # Create an integer
f = 0.00 # Create a float
```

x + y: sum of x and y
x - y: subtraction of x and y
x * y: x times y
a ** n: a to the power of n
a / b: division, return a float
a // b: floor division, discard the fractional part
a % b: the operator % returns the remainder of the division
_: console, last printed expression is assigned to the variable _
abs(x): absolute value or magnitude of x
int(x): x converted to integer
float(x): x converted to floating point
complex(re, im): complex number, re: real, im: imaginary
c.conjugate(): conjugate of the complex number c

Operators

Comparison:
<: strictly less than
<=: less than or equal
>: strictly greater than
>=: greater than or equal
==: equal
!=: not equal
is: object identity
is not: negated object identity

Logical:
and: true if both true
or: true if only one true
not: reverse the result

Identity:
is: true if both variable are the same object
is not: true if both variable are not the same object

Membership:
in: true if the value is in the object
not in: true if value not in object

Strings

Define a string variable:

```
s = "" or '' # Create an empty string
```

"doesn't": mix to use single quote
\n: go to the next line **\t:** tab space **\r:** carriage return
r"...": raw strings
"""...""": string literals on multiple lines, use \ to prevent \n
+, *: sum or multiply strings

Strings are immutable (don't support index assignment)
a[n]: to access n index of a (start from 0)
a[-k]: to access n-k index of a (start from n+1)
a[i:j]: range from i to j, leave black to get first or last

Methods (all return a copy of the string):
len(a): return the length of the string
s.capitalize(): first character capitalized and the rest lowercased
s.casefold(): casefolded copy of the string.
s.center(w, c): centered in a string of length w, fill with c
s.count(i): count ammount of substring i in str
s.encode(): encode the string, default: utf-8
s.endswith(s): true if s end with s, false otherwise
s.startswith(s): true if s start with s, false otherwis
s.expandtabs(n): expand the tab (\t)
s.find(i): return smallest index of i in s, -1 if not found
s.index(i): like find but raise an error
s.isalnum(): true if alphanumeric
s.isalpha(): true if all alphabetic
s.isascii(): true if ascii
s.isdecimal(): true if all decimal
s.isdigit(): true if all digit
s.islower(): true if all lowercase
s.lower(): all cased characters converted to lowercase
s.isupper(): true if all uppercase
s.upper(): all cased characters converted to uppercase
s.isnumeric(): true if all numeric character
s.isspace(): true if only whitespace characters
s.istitle(): true if title first upper rest lower
s.title(): convert s to a title
s.strip([c]): strip c from left and right, black by deafult
s.lstrip([c]): -> remove c from left, blank by default
s.rstrip([c]): <- remove c from right, blank by deafult
s.removeprefix(i): remove prefix i
s.removesuffix(i): remove sufflix i
s.replace(old, new, count): replace all old with new
s.split(sep): split on sep return a list
s.splitlines(): split on \n return a list
s.swapcase(): convert upper to lower and viceversa

Tuples and Sets

Define a set and tuple

```
t = () # Create an empty tuple
s = set() # Create an empty set
```

Lists

Define a list

```
l = [] # Create an empty list
```

Lists are mutable (support index assignment)
l[:]: return a copy of l
l[n] = k: set n index to k, work also with a slice (l[i:j])
[i for i in a]: concise way to create lists
del l[n]: delete n index, work also with interval

Data structure:
len(l): return the length of the list
min(l): return smallest item of s
max(l): return largest item of s
l.append(i): add i to the end of the list (a[len(a):] = [x])
l.extend(it): appending all the items (a[len(a):] = it)
l.insert(i, x): insert at i, x all the value shift
l.remove(x): remove first item that equal x. error if don't exist
l.pop([i]): remove index i, if blank the last. return the value
l.clear(): remove all the elements (del a[:])
l.index(x): return index of x. error if don't exist
l.count(x): return the ammount of x
l.sort(): sort the list
l.reverse(): flip the list ([::-1])
l.copy(): return a copy of the list ([:])

Dictionaries

Definition of a empty dictionary

```
d = {}
```

list(d): return a list of keys
sorted(d): return all the keys sorted

Flow Tools

if Statements

loop

Statement
continue: skip the rest of the code but continue the loop
break: stop the loop

While loop:

```
while (statement):
else: #when loop finish without a break
```

For loop:

```
for (iterator):
else: #when loop finish without a break
```

Classes Special Attributes

__dict__: A dictionary or other mapping
__class__: The class to which a class instance belongs
__bases__: The tuple of base classes of a class object.
__name__: The name of the class, function, method, descriptor, or generator instance.

---:
---:
---:
---:
---:
---:
---:
---:
---:
---:

Libraries

Json

Import json package:

```
import json
```

loads(s): convert json string in dictionary
dumps(d): convert dict in json string
Use: **ident=4, sort_keys=True** to prettify

random

Import random module:

```
import random
```

seed(value): Inizialize the random number genetator
randrange(start, stop, step): return a float in the range
ranint(start, stop+1): return an integer in the range
choice(seq): return a random element of the sequence
choices(seg, weights, k): return k element of the sequence
shuffle(seq): shuffle the sequence
random(): return a random float between 0 and 1
uniform(a, b): return a float in the range
triangular(a, b, center): return a float in range with center

Os

Math

Random

Statistics

Matplotlib

Numpy

Pandas

Datetime

Timeit

Pygame

Threading

Requests

Import request module:

```
import requests
```

get(url, params, args): GET request to url
post(url, data, json, args): POST request to url
put(url, data, args): PUT request to url
delete(url, args): DELETE request to url
head(url, args): HEAD request to url
patch(url, data, args): PATCH request to url

Flask