marcos-gral Cheat Sheet

by Marcos via cheatography.com/34357/cs/16484/

Python Classes

class Child(Parent):	def __init__(self, *args,* *kwargs):
super().__init__(self, args) # ???	@staticmethod # ?????
@classmethod # ?????	
@property # for getter	@property.setter

Python

```
lambda [parameter_list]: expresion
def func(n):
    """ Documentation """
```

Python Indexes and Slices

len(a)	a[::-1] # reverse
b=a[:] # Shallow copy	

Python Lists

lst.append(item)	lst.pop(item)
lst.count(item)	lst.remove(item) # first item found
del lst[i]	
lst.extend(lst2)	lst.reverse()
lst.index(item)	lst.sort()
sorted(ll) # sorts without modifying	lst.insert(position, item)
[x for x in lst if cond]	

Dictionary Operations

len(d)	del d[key] # KeyError Exception
k in d	d.keys()
d.setdefault(key[,default]) # ???	d.clear() # ???
dict(a=1, b=2)	dict(zip(['a', 'b'], [1, 2]))
dict([('A', 1), ('Z', -1)])	dict({'Z': -1, 'A': 1})

Comprehensions

list	[expression for iterable_clause if optional_filter]
nested lists	[expression for iterable_clause_1 for iterable_clause_2 if optional_filter]
dict	{ key_expression: value_expression for iterable_clause if optional_filter}
set	{ value_expression for iterable_clause if optional_filter}

Python Scripting

sys.argv	parametes pass to the command
sys.exit(0)	0 is success
sys.path	list of paths to packages
sys.version_info	
os.environ	Dictionary of environment Vars
os.curdir	

Python requests

```
r = requests.get(url , params={'foo': 'bar'} )
r = requests.post(url , params={'foo': 'bar'} )
r.status_code # 200
r.text
r.json
r.headers
r.encoding # utf-8 / ISO-xxx
import json
url = 'https://api.github.com/some/endpoint'
payload = {'some': 'data'}
headers = {'content-type': 'application/json'}

r = requests.post(url, data=json.dumps(payload),
headers=headers)
r = requests.put("http://httpbin.org/put")
r = requests.delete("http://httpbin.org/delete")
r = requests.head("http://httpbin.org/get")
r = requests.options("http://httpbin.org/get")
```

marcos-gral Cheat Sheet

by Marcos via cheatography.com/34357/cs/16484/

Python Datetime

dt.today()	dt.now(timezoneinfo)
dt.combine(date, time)	dt.utcnow()
dt.strptime(date, format)	dt.fromtimestamp(timestamp)
dt.timestamp() # ????	dt.utcfromtimestamp(timestamp)

from datetime import datetime as dt

Python Time Methods

time.replace()	time.utcoffset()
time.isoformat()	time.dst()
time.tzname()	

Python Date Formatting

https://www.cheatography.com/davechild/cheat-sheets/python/

Python Iteration

```
for i in range(start,stop,step):
    stuff
for value in [sequence]:
    stuff
for position, value in enumerate(sequence):
    stuff
for a,b in zip(first, second):
   stuff
for ###
else
    stuff to do at end of loop (usually exception when
breaking in loop)
while condition:
    stuff
else
    stuff to do when condition is false
break # breaks the loop
continue # continue at the start of the loop body
```

module itertools provides lots of interesting tools for iteration

varios

"Hello, %s." % name
"Hello, %s. You are %s." % (name, age)
"Hello, {}. You are {}.".format(name, age)
"Hello, {1}. You are {0}.".format(age, name)
person = {'name': 'Eric', 'age': 74}
"Hello, {name}. You are {age}.".format(name=person['name'], age=person['age'])
"Hello, {name}. You are {age}.".format(**person)
>>> name = "Eric"
>>> age = 74
>>> f"Hello, {name}. You are {age}."
F"Hello, {name}. You are {age}."
f"{name.lower()} is funny."
f"{my:func(name)} is funny."
https://realpython.com/python-f-strings/
==========

Operations on Sets ????

|	union
&	intersection
-^	difference/symetric diff
< <= > >=	inclusion relations
s.update(s2)	s.add(key)
s.copy()	s.discard(key)
s.pop()	s.clear()

Python Math

5 // 2 = 2 5 % 2 = 1

Python Generators

yield x	next(func)
generator.send(x)	for in in generator_function(**some_params)

yield from list_comprehension # ????

marcos-gral Cheat Sheet

by Marcos via cheatography.com/34357/cs/16484/

in-built functions

min(values), max(values)	range(start, stop[, step])
filter(function, array) # ???	map(func, array) # ???
id(object)	round(n, [decimal places]

Python Common Exceptions

IndexError	KeyError
StopIteration	TimeoutError
AttributeError	AssertionError

https://docs.python.org/3/library/exceptions.html

Python Random

random.seed(1)	random.randrange(stop)
random.randrange(start, stop[,step])	random.randint(a, b)
random.choice(seq)	random.choices(population, k=1) # IndexError
random.shuffle(x)	random.sample(population, k)

Python File

f = open(path)	f.read() # Read f
f.readline()	f.readlines()
f.write(s)	f.close()
with f = open(path, 'r'):	

Python Regular expressions Module

re.compile(pattern, flags=0)	regex.search(string[,pos][,endpos])
regex.match(string)	regex.fullMatch(string)
match.group([group1, ...])	match.groups()

Python String Methos

s.lstrinp()	s.partition()
s.decode() # ???	s.rjust(wirth[, fillchar])
s.rfind(item)	s.split(sep)
s.splitlines()	s.isalpha()
s.isdigit()	s.startswith(sub)
s.strip()	s.isspace()

Python String Methos (cont)

s.encode('utf-8') # ???	b"string" # bytes object

String Formating

```
"Hello, {0} {1}".format("abe", "jones")
Hello, abe jones
"Hello, {fn} {ln}".format(fn="abe", ln="jones")
Hello, abe jones
"You owe me ${0:,.2f}".format(253422.3)
You owe me $253,422.30
now = datetime.now()
'{:%Y-%m-%d %H:%M:%S}'.format(now)
2012-05-16 15:04:33
```

Exceptions

```
try:
except ExceptionName as e:
except (ExceptionName, OtherException) as e:
else:
    # do something when no exception
finally:
    # do something anyway, exception or not
```

Code Snippets

```
Loop Over Sequence
for index, value in enumerate(seq):
  print("{} : {}".format(index, value))
Loop Over Dictionary
for key in sorted(dict):
  print(dict[key])
Read a File
with open("filename", "r") as f:
  for line in f:
    line = line.rstrip("\n") # Strip newline
    print(line)
```

Python Decorator

```
def wrap(func):
    def wrapper(args, *kwargs):
        # do something about func
        func(args, *kwargs)
        # do something about func
    return wrapper
# Apply decorator
def to_decorate(...):
# body
to_decorate = wrap(to_decorate)
# More idiomatic
@wrap
def to_decorate(...):
#body
from functools import wraps
@wraps(func)
def wrapper(...) # to keep the name and doc from the
wrapped function
# Decorator with args: make a decorator factory
def decorator_factory(factory_args):
    def decorator(func):
        def wrapper(args, *kwargs):
            # do something about func
            func(args, *kwargs)
            # do something about func
        return wrapper
    return decorator
@decorator_factory(1,2...)
def to_decorate(...):
```

Iterator

```
class zrange_iter:
    def __init__(self, n):
        self.i = 0
        self.n = n
    def __iter__(self):
        # Iterators are iterables too.
        # Adding this functions to make them so.
```

Iterator (cont)

```
        return self
    def next(self):
        if self.i < self.n:
            i = self.i
            self.i += 1
            return i
        else:
            raise StopIteration()
```

Generator

```
def firstn(n):
    num = 0
    while num < n:
        yield num
        num += 1
sum_of_first_n = sum(firstn(1000000))
# yield from my_gen(x)
g = my_gen(x)
try:
    next(g)
except StopIteration:
    pass
# Unpacking Generators
>>> g1 = (x for x in range(3))
>>> g2 = (x**2 for x in range(2))
>>> [1, g1, 2, g2]
[1, 0, 1, 2, 2, 0, 1]
>>> g = (x for x in range(3))
>>> a, b, c = g
>>> print(a, b, c)
0 1 2
>>> g = (x for x in range(6))
>>> a, b, *c, d = g
>>> print(a, b, d)
0 1 5
>>> print(c)
[2, 3, 4]
```

IMPORTANTE:

https://www.pythonsheets.com/notes/python-generator.html