EVU Python LESSON II

Dagens program

Velkomst
- Dagens program
- præsentation

Data Typer

recap

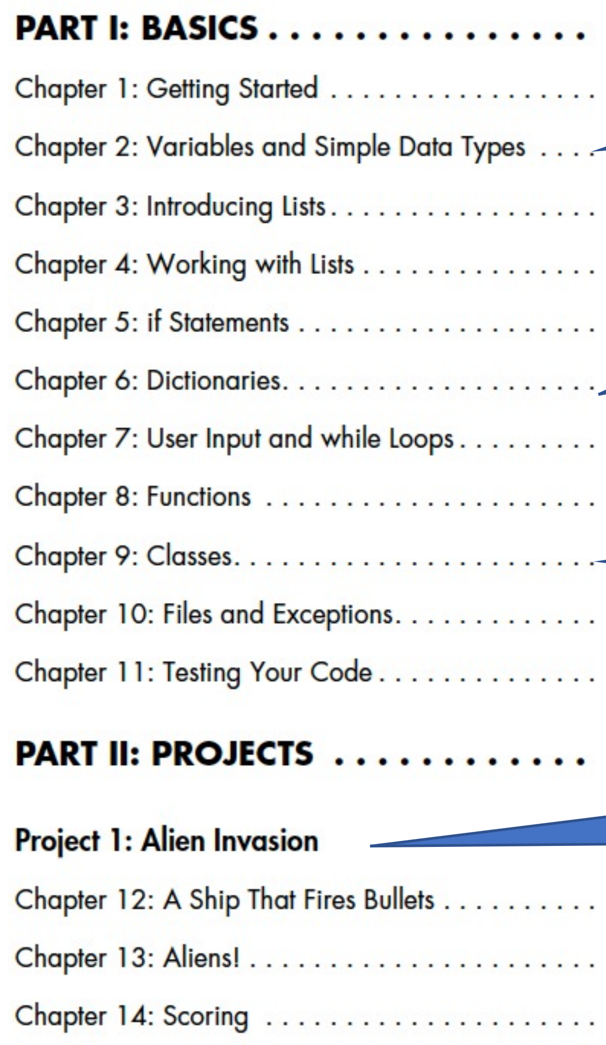
Dicts
- What?
- how-to
- Loops

Functions

Basic

Velkomst

kursets forløb



27/1

10/2

24/2

10/3

Teaser

Sports Analytics

```
{
  "eventId": 10,
  "subEventName": "Shot",
  "tags": [
    {
      "id": 401
    },
    {
      "id": 201
    },
    {
      "id": 1215
    },
    {
      "id": 1802
    }
  ],
  "playerId": 12536,
  "positions": [
    {
      "y": 33,
      "x": 87
    },
    {
      "y": 0,
      "x": 0
    }
  ],
  "matchId": 2499725,
  "eventName": "Shot",
  "teamId": 1613,
  "matchPeriod": "1H",
  "eventSec": 283.438159,
  "subEventId": 100,
  "id": 178442509
},
```

Semantics

Token

The smallest individual unit in a program is known as *Tokens* or *lexical units.*

Expression

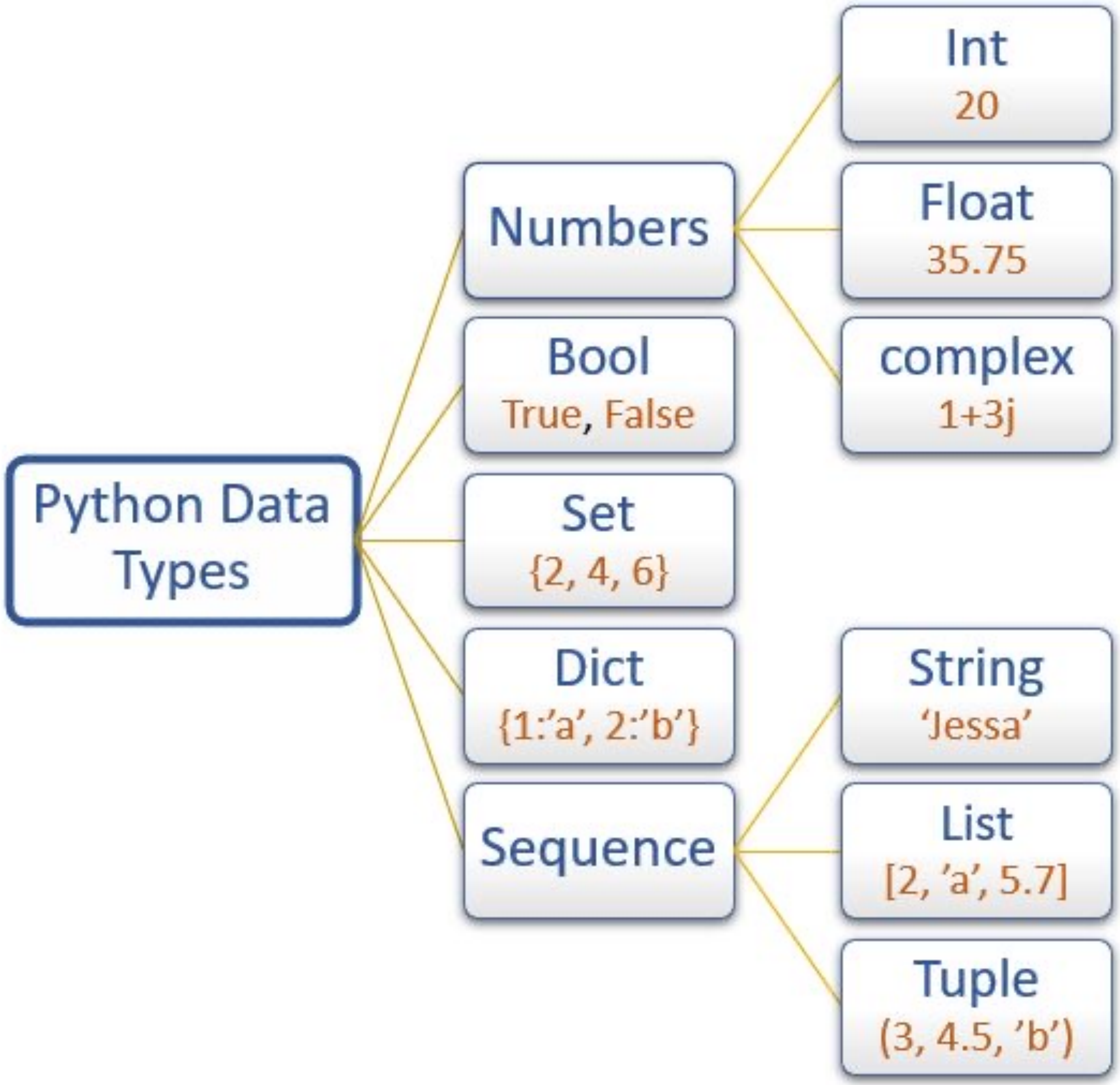
Single, sequence or combination of values, variables, operators and function calls that always produces or returns a result value

Statement

Any Instruction that a python interpreter can execute (carry out) is called a Statement. Every statement can be an expression – but not all expressions can be statements.

Datatyper

Overview



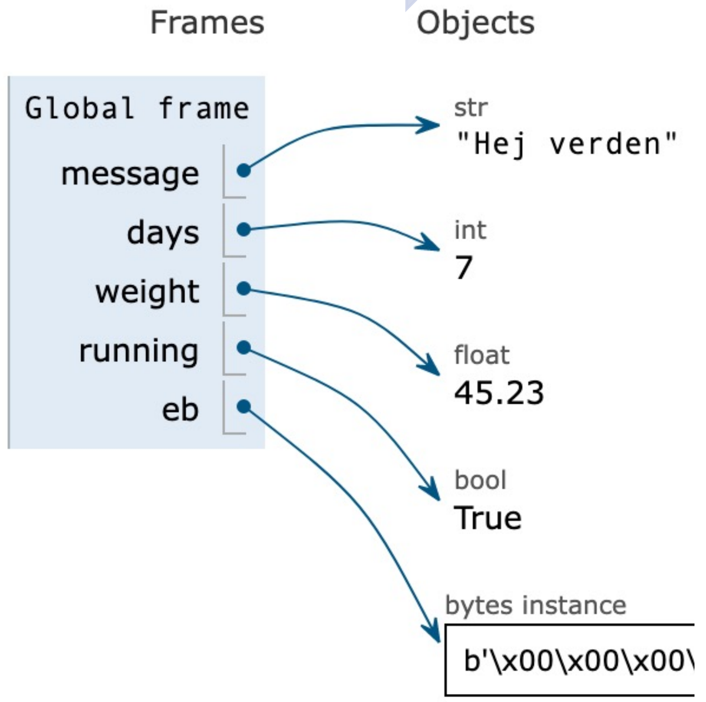
A Python variable is a reserved memory location to store values.

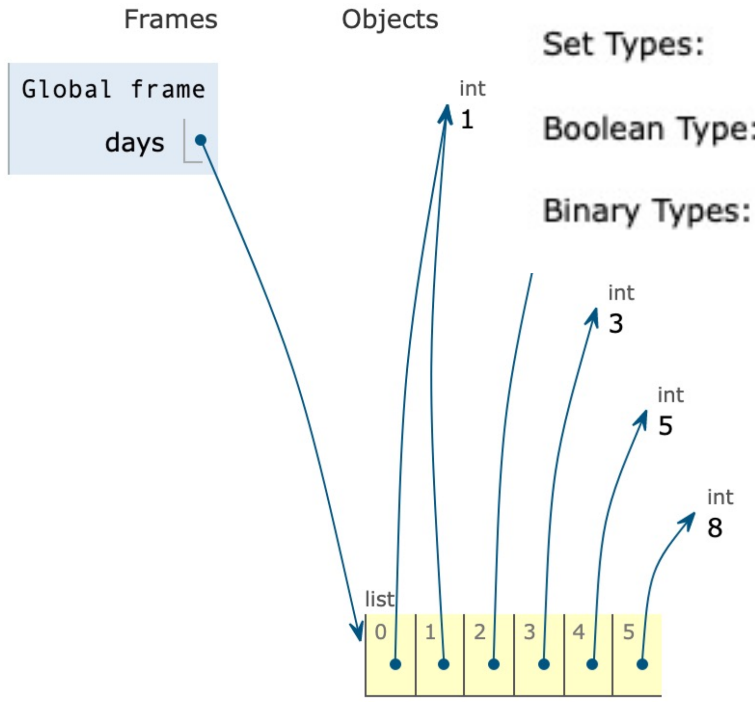
Every value in Python has a datatype.

Datatyper ”Alle”

A Python variable is a reserved memory location to store values.
Every value in Python has a datatype.

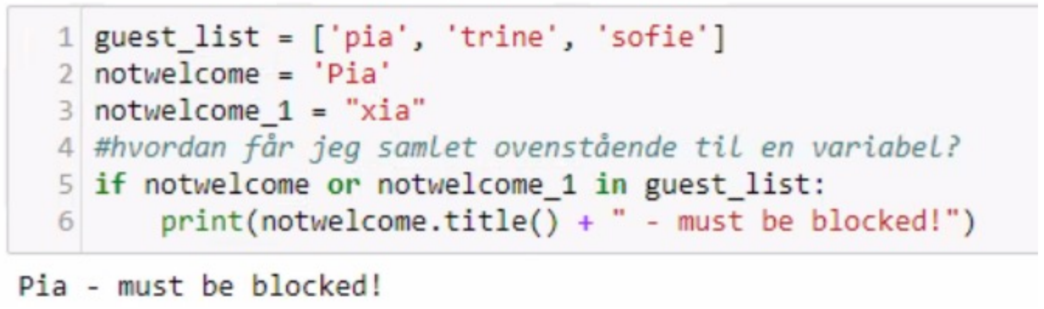
Text Type:	`str`
Numeric Types:	`int`, `float`, `complex`
Sequence Types:	`list`, `tuple`, `range`
Mapping Type:	`dict`
Set Types:	`set`, `frozenset`
Boolean Type:	`bool`
Binary Types:	`bytes`, `bytearray`, `memoryview`





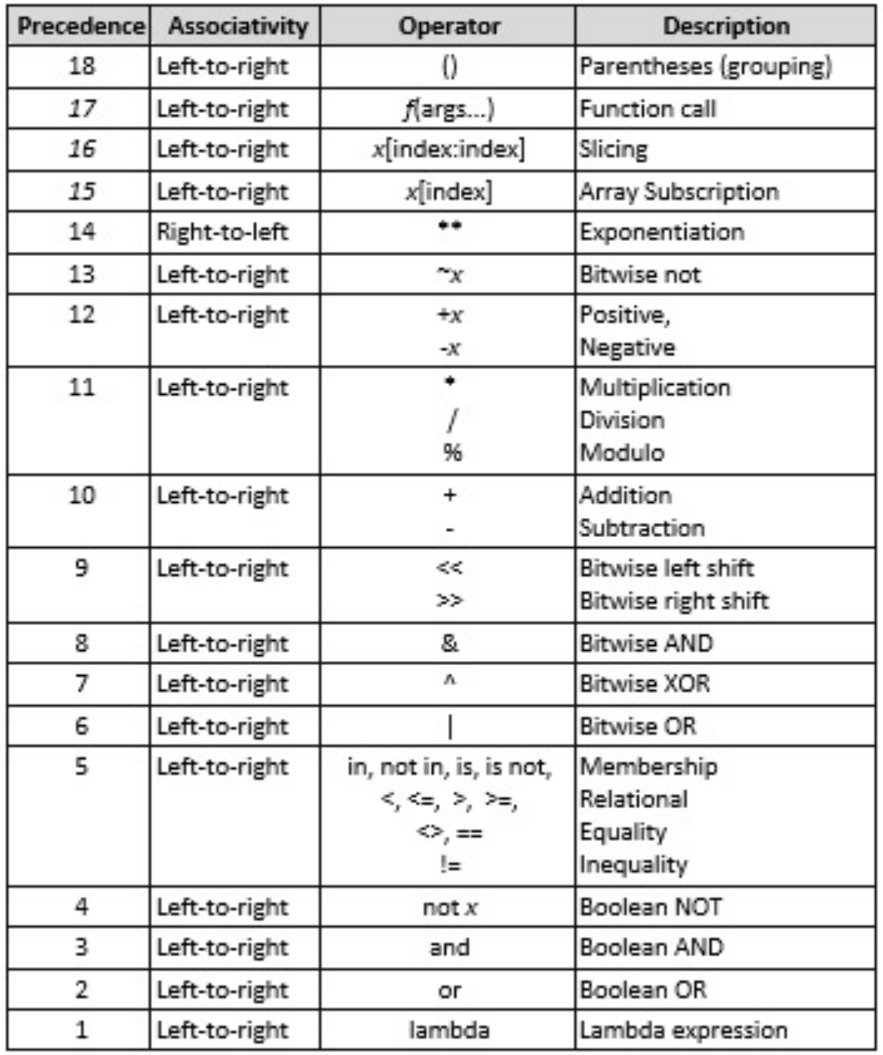
Datatyper

Exerc 5-10



```python
guest_list = ['pia', 'trine', 'sofie']
notwelcome = 'Pia'
notwelcome_1 = "xia"
#hvordan får jeg samlet ovenstående til en variabel?
if notwelcome or notwelcome_1 in guest_list:
    print(notwelcome.title() + " - must be blocked!")
```

```
Pia - must be blocked!
```



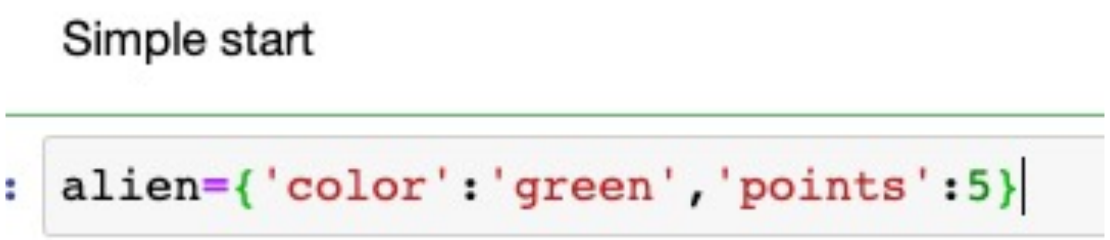
Precedence	Associativity	Operator	Description
18	Left-to-right	()	Parentheses (grouping)
17	Left-to-right	*f*(args...)	Function call
16	Left-to-right	*x*[index:index]	Slicing
15	Left-to-right	*x*[index]	Array Subscription
14	Right-to-left	**	Exponentiation
13	Left-to-right	~*x*	Bitwise not
12	Left-to-right	+*x* -*x*	Positive, Negative
11	Left-to-right	* / %	Multiplication Division Modulo
10	Left-to-right	+ -	Addition Subtraction
9	Left-to-right	<< >>	Bitwise left shift Bitwise right shift
8	Left-to-right	&	Bitwise AND
7	Left-to-right	^	Bitwise XOR
6	Left-to-right	|	Bitwise OR
5	Left-to-right	in, not in, is, is not, <, <=, >, >=, <>, == !=	Membership Relational Equality Inequality
4	Left-to-right	not *x*	Boolean NOT
3	Left-to-right	and	Boolean AND
2	Left-to-right	or	Boolean OR
1	Left-to-right	lambda	Lambda expression

Pause

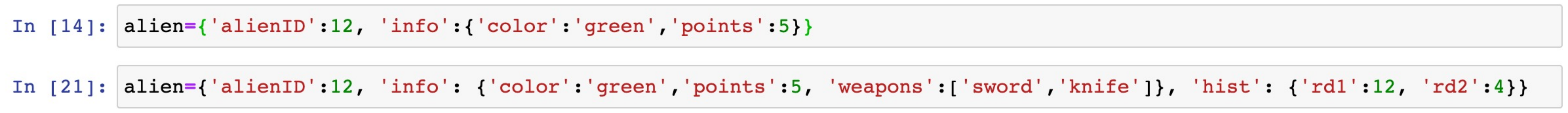
Datatyper

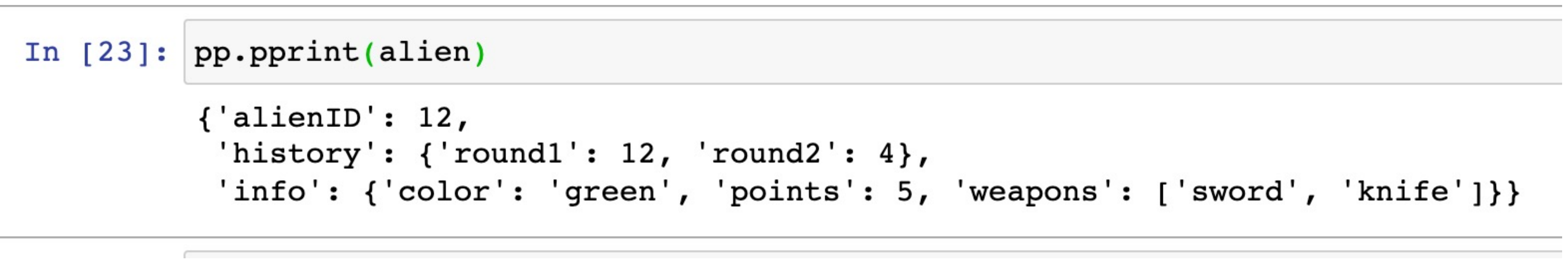
Dicts – hvad?

- A dictionary in Python is a collection of key-value pairs.
- Each key is connected to a value.
- You can use a key to access the value associated with that key.
- A key's value can be a number, a string, a list, a dictionary or any object

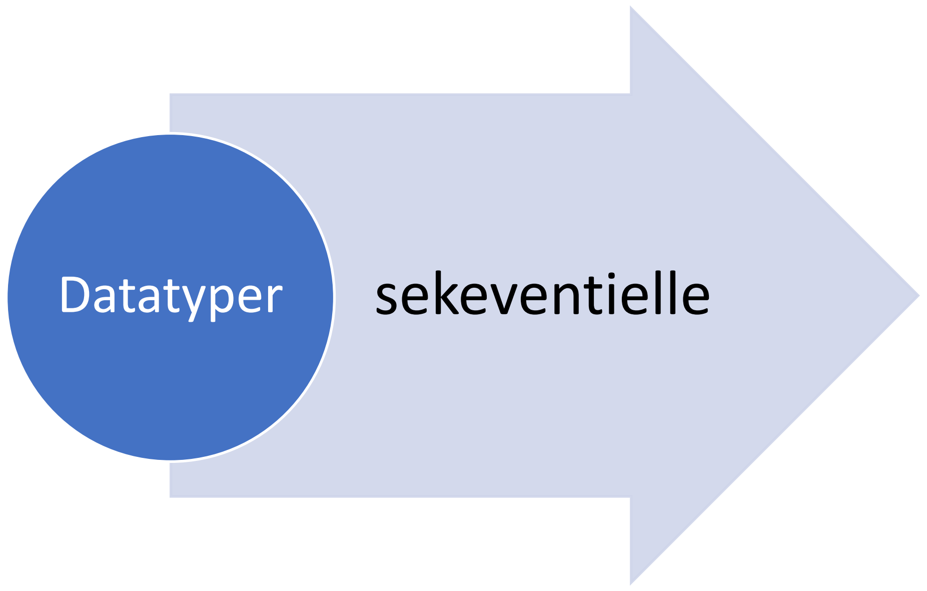

Simple start

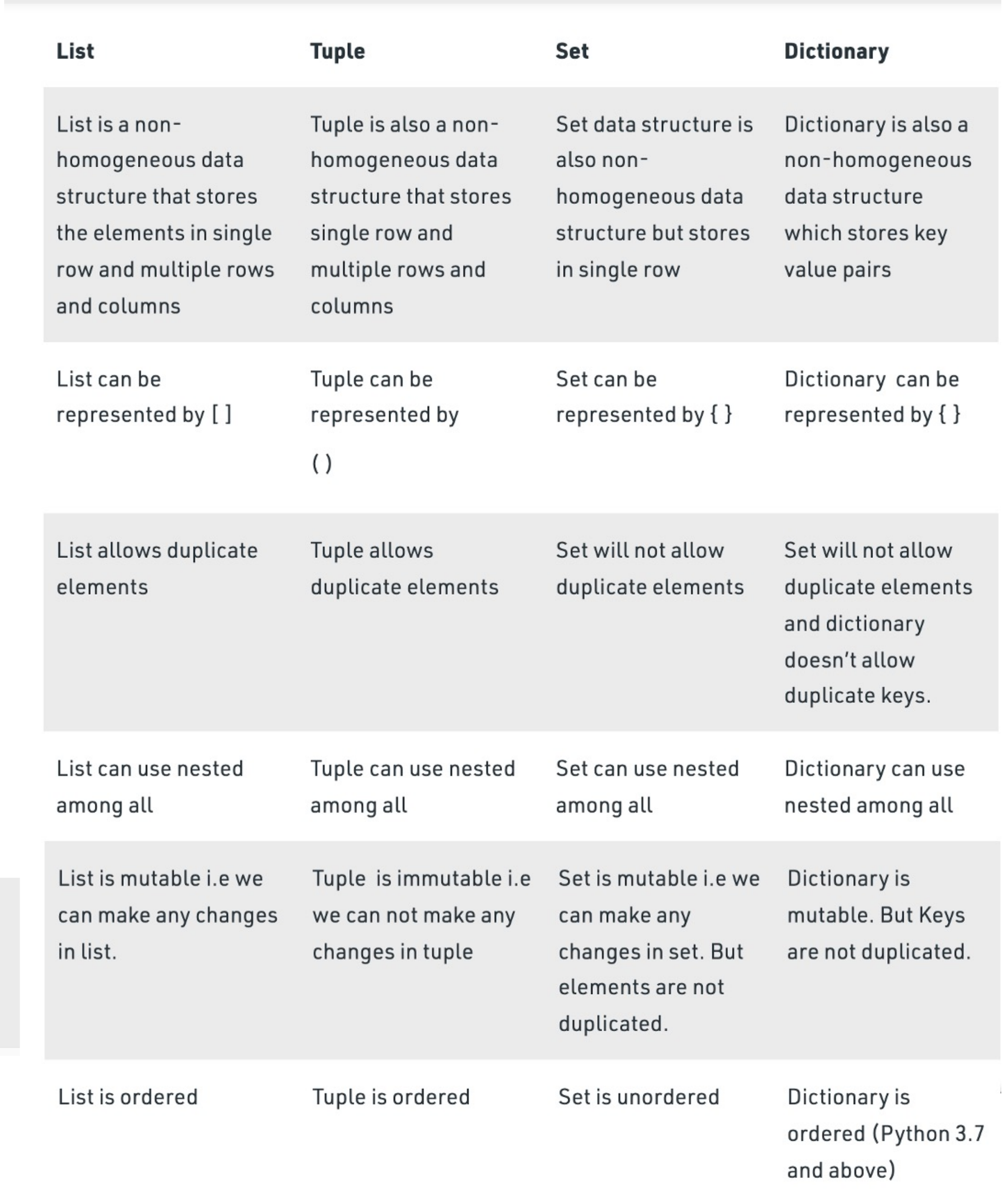
```
alien={'color':'green','points':5}
```


```
In [14]: alien={'alienID':12, 'info':{'color':'green','points':5}}

In [21]: alien={'alienID':12, 'info': {'color':'green','points':5, 'weapons':['sword','knife']}, 'hist': {'rd1':12, 'rd2':4}}
```

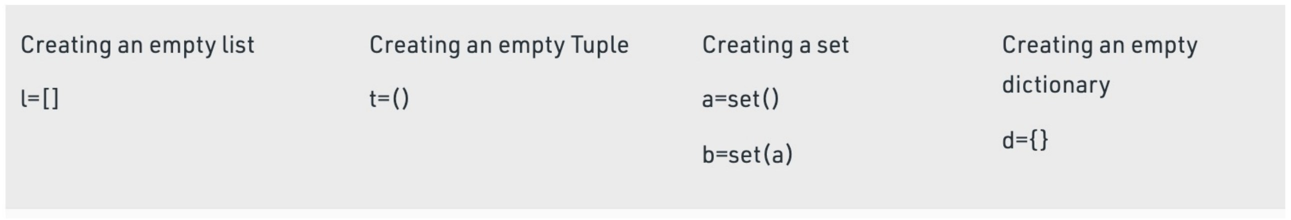

```
In [23]: pp.pprint(alien)

         {'alienID': 12,
          'history': {'round1': 12, 'round2': 4},
          'info': {'color': 'green', 'points': 5, 'weapons': ['sword', 'knife']}}
```

"Pretty Print" json,xml





List	Tuple	Set	Dictionary
List is a non-homogeneous data structure that stores the elements in single row and multiple rows and columns	Tuple is also a non-homogeneous data structure that stores single row and multiple rows and columns	Set data structure is also non-homogeneous data structure but stores in single row	Dictionary is also a non-homogeneous data structure which stores key value pairs
List can be represented by [ ]	Tuple can be represented by ( )	Set can be represented by { }	Dictionary can be represented by { }
List allows duplicate elements	Tuple allows duplicate elements	Set will not allow duplicate elements	Set will not allow duplicate elements and dictionary doesn't allow duplicate keys.
List can use nested among all	Tuple can use nested among all	Set can use nested among all	Dictionary can use nested among all
List is mutable i.e we can make any changes in list.	Tuple is immutable i.e we can not make any changes in tuple	Set is mutable i.e we can make any changes in set. But elements are not duplicated.	Dictionary is mutable. But Keys are not duplicated.
List is ordered	Tuple is ordered	Set is unordered	Dictionary is ordered (Python 3.7 and above)



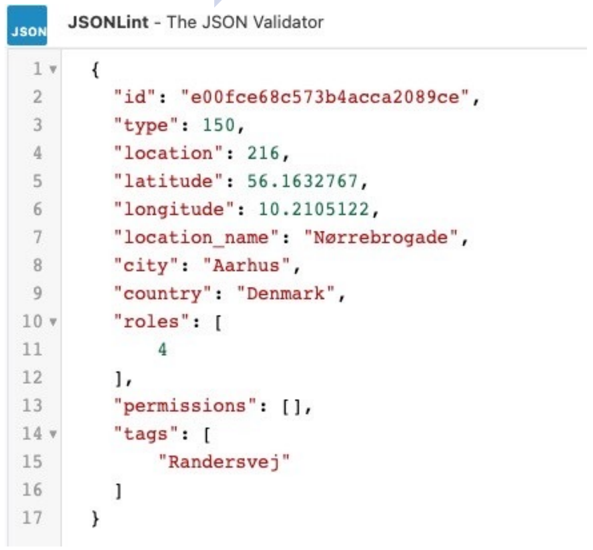
Creating an empty list
l=[]
	Creating an empty Tuple
	t=()
		Creating a set
		a=set()
		b=set(a)
			Creating an empty dictionary
			d={}

Datatyper

Json og dicts

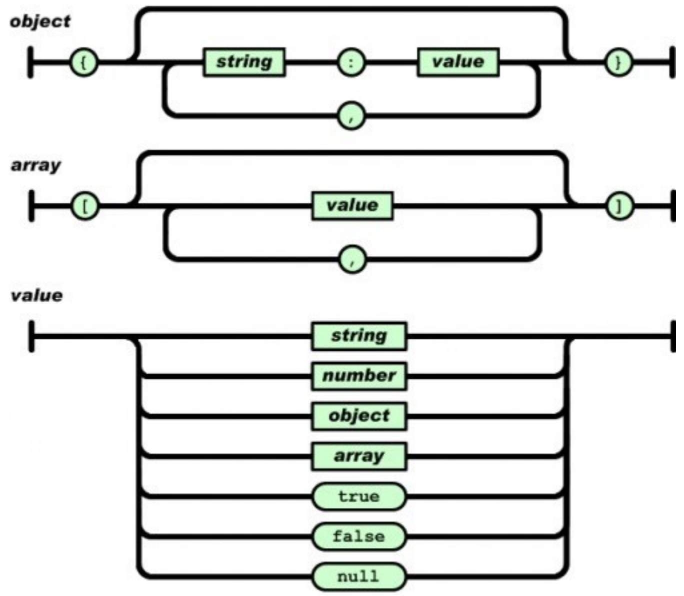
Cityflow – REST API Data Format

- ”The above command returns JSON structured like this”
- JSON – JavaScript Object Notation



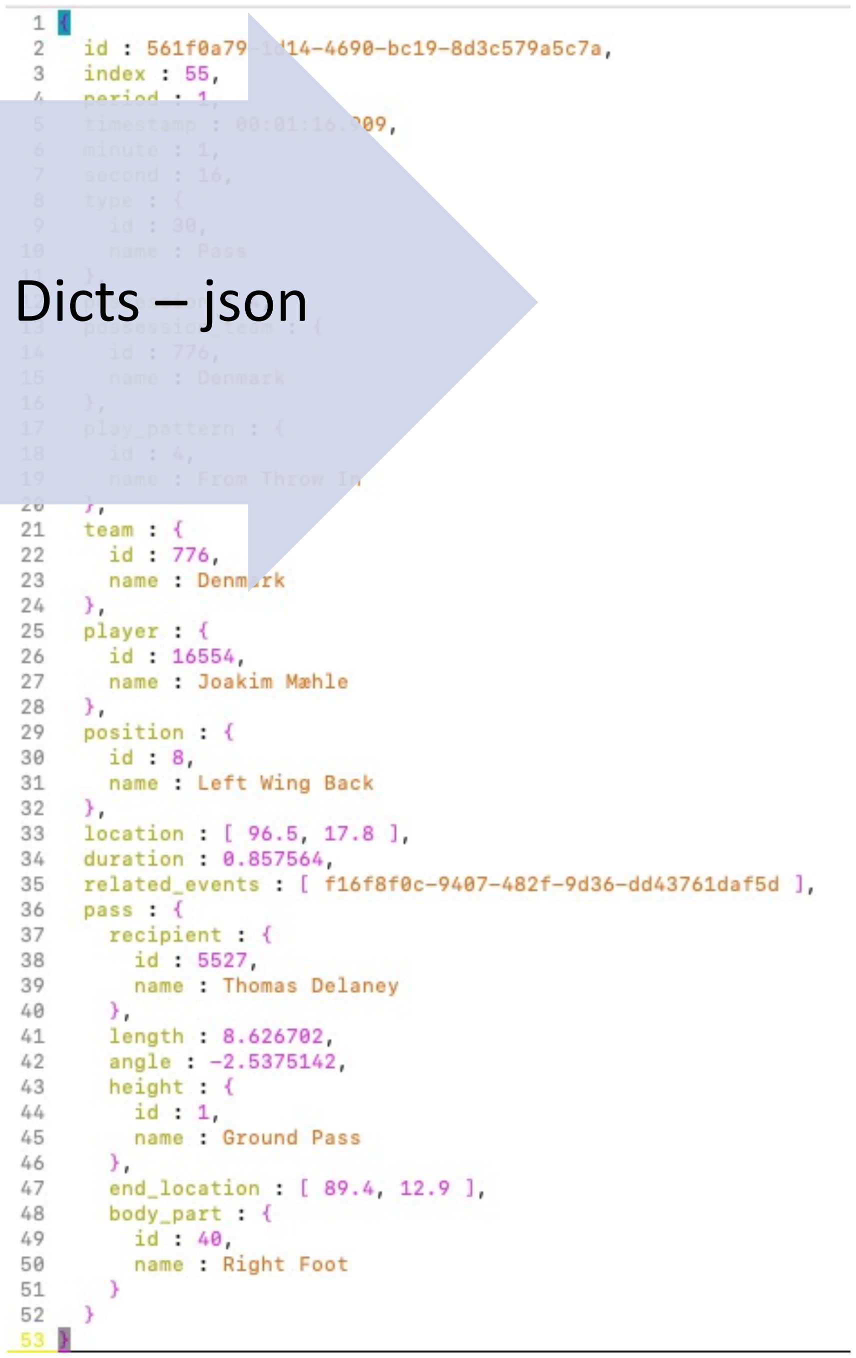
JSONLint - The JSON Validator

```
{
  "id": "e00fce68c573b4acca2089ce",
  "type": 150,
  "location": 216,
  "latitude": 56.1632767,
  "longitude": 10.2105122,
  "location_name": "Nørrebrogade",
  "city": "Aarhus",
  "country": "Denmark",
  "roles": [
      4
  ],
  "permissions": [],
  "tags": [
      "Randersvej"
  ]
}
```

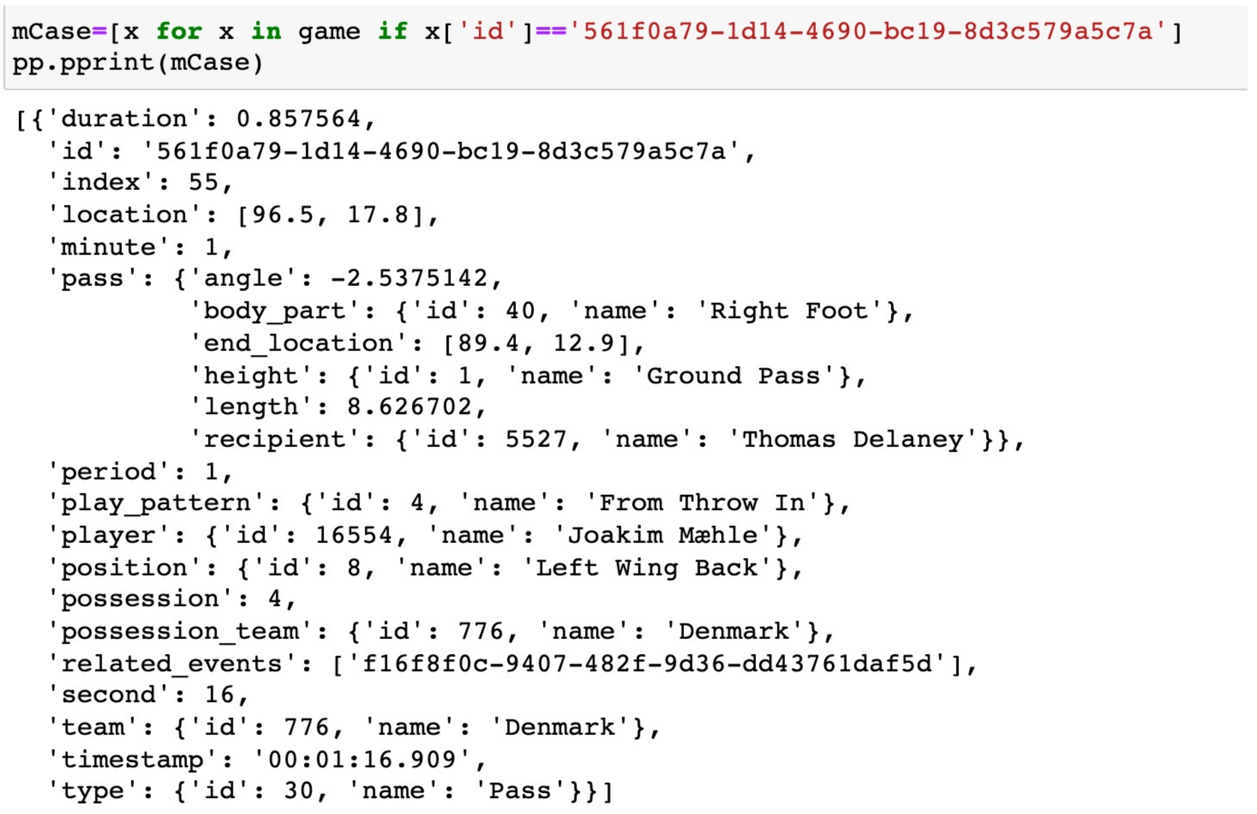


Datatyper

Dicts – json



```
{
  id : 561f0a79-1d14-4690-bc19-8d3c579a5c7a,
  index : 55,
  period : 1,
  timestamp : 00:01:16.909,
  minute : 1,
  second : 16,
  type : {
    id : 30,
    name : Pass
  },
  possession : 4,
  possession_team : {
    id : 776,
    name : Denmark
  },
  play_pattern : {
    id : 4,
    name : From Throw In
  },
  team : {
    id : 776,
    name : Denmark
  },
  player : {
    id : 16554,
    name : Joakim Mæhle
  },
  position : {
    id : 8,
    name : Left Wing Back
  },
  location : [ 96.5, 17.8 ],
  duration : 0.857564,
  related_events : [ f16f8f0c-9407-482f-9d36-dd43761daf5d ],
  pass : {
    recipient : {
      id : 5527,
      name : Thomas Delaney
    },
    length : 8.626702,
    angle : -2.5375142,
    height : {
      id : 1,
      name : Ground Pass
    },
    end_location : [ 89.4, 12.9 ],
    body_part : {
      id : 40,
      name : Right Foot
    }
  }
}
```



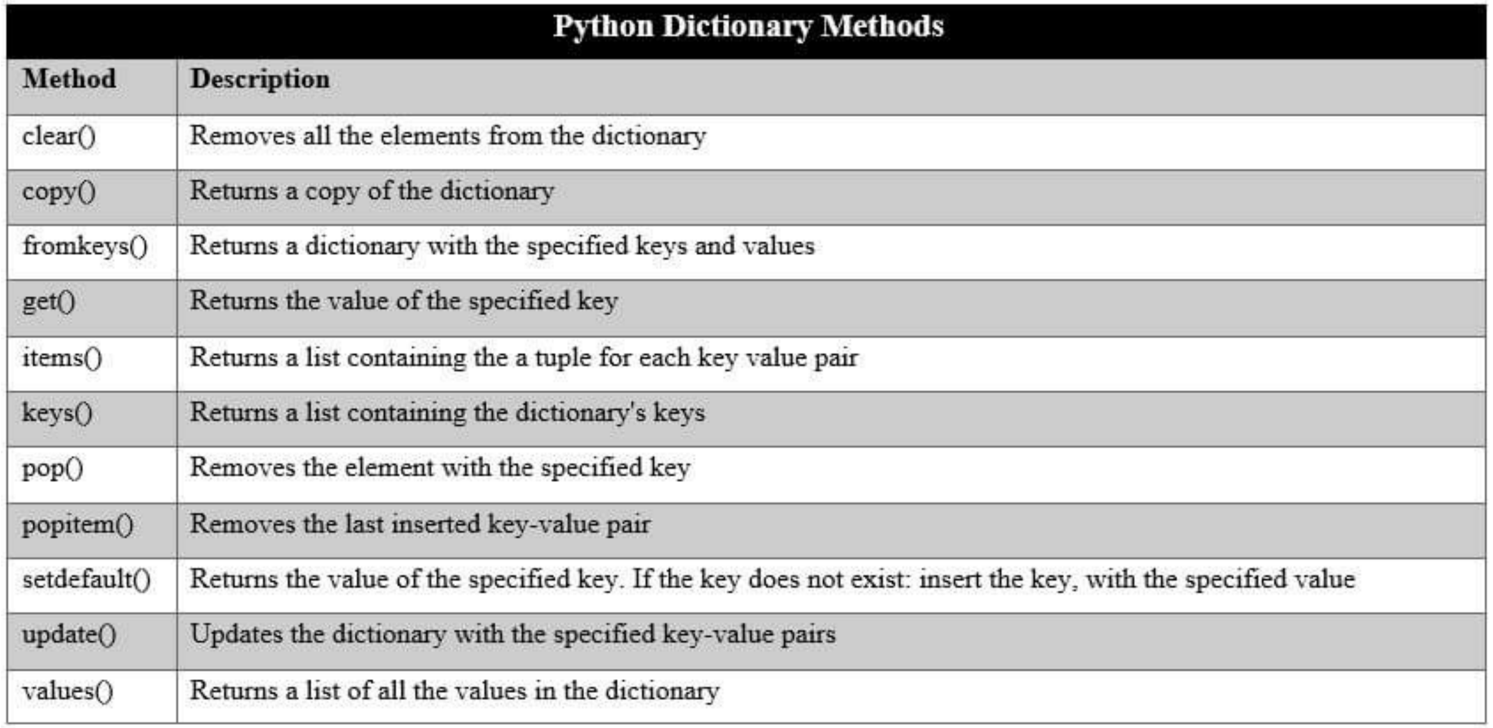
```python
mCase=[x for x in game if x['id']=='561f0a79-1d14-4690-bc19-8d3c579a5c7a']
pp.pprint(mCase)
```

```
[{'duration': 0.857564,
  'id': '561f0a79-1d14-4690-bc19-8d3c579a5c7a',
  'index': 55,
  'location': [96.5, 17.8],
  'minute': 1,
  'pass': {'angle': -2.5375142,
           'body_part': {'id': 40, 'name': 'Right Foot'},
           'end_location': [89.4, 12.9],
           'height': {'id': 1, 'name': 'Ground Pass'},
           'length': 8.626702,
           'recipient': {'id': 5527, 'name': 'Thomas Delaney'}},
  'period': 1,
  'play_pattern': {'id': 4, 'name': 'From Throw In'},
  'player': {'id': 16554, 'name': 'Joakim Mæhle'},
  'position': {'id': 8, 'name': 'Left Wing Back'},
  'possession': 4,
  'possession_team': {'id': 776, 'name': 'Denmark'},
  'related_events': ['f16f8f0c-9407-482f-9d36-dd43761daf5d'],
  'second': 16,
  'team': {'id': 776, 'name': 'Denmark'},
  'timestamp': '00:01:16.909',
  'type': {'id': 30, 'name': 'Pass'}}]
```

Datatyper

Dicts in action I

- Creating dicts
- Accessing keys and/or values
- Adding items
- Updating items
- Printing (format)
- Complicated dicts
  - Lists in dicts
  - Dicts in dicts
  - List of Dicts



Python Dictionary Methods

Method	Description
clear()	Removes all the elements from the dictionary
copy()	Returns a copy of the dictionary
fromkeys()	Returns a dictionary with the specified keys and values
get()	Returns the value of the specified key
items()	Returns a list containing the a tuple for each key value pair
keys()	Returns a list containing the dictionary's keys
pop()	Removes the element with the specified key
popitem()	Removes the last inserted key-value pair
setdefault()	Returns the value of the specified key. If the key does not exist: insert the key, with the specified value
update()	Updates the dictionary with the specified key-value pairs
values()	Returns a list of all the values in the dictionary

Datatyper

Dicts in action I

Accessing Values in a Dictionary

Adding New Key-Value Pairs

Starting with an Empty Dictionary

Modifying Values in a Dictionary

Removing Key-Value Pairs

A Dictionary of Similar Objects

A List of Dictionaries

A List in a Dictionary

A Dictionary in a Dictionary

Looping Through All Key-Value Pairs

Looping Through All the Keys in a Dictionary

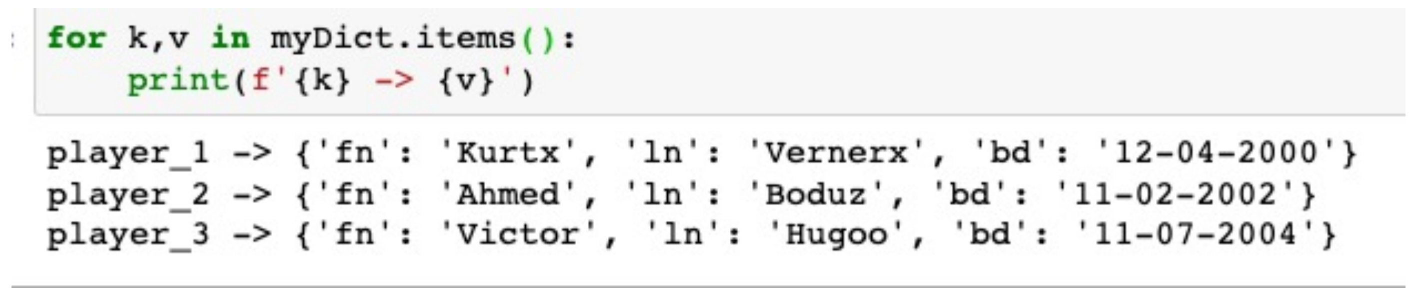
Looping Through a Dictionary’s Keys in Order

Looping Through All Values in a Dictionary

Datatyper

Dicts in action III

Looping



```
for k,v in myDict.items():
    print(f'{k} -> {v}')
```

```
player_1 -> {'fn': 'Kurtx', 'ln': 'Vernerx', 'bd': '12-04-2000'}
player_2 -> {'fn': 'Ahmed', 'ln': 'Boduz', 'bd': '11-02-2002'}
player_3 -> {'fn': 'Victor', 'ln': 'Hugoo', 'bd': '11-07-2004'}
```

Datatyper

Øvelser

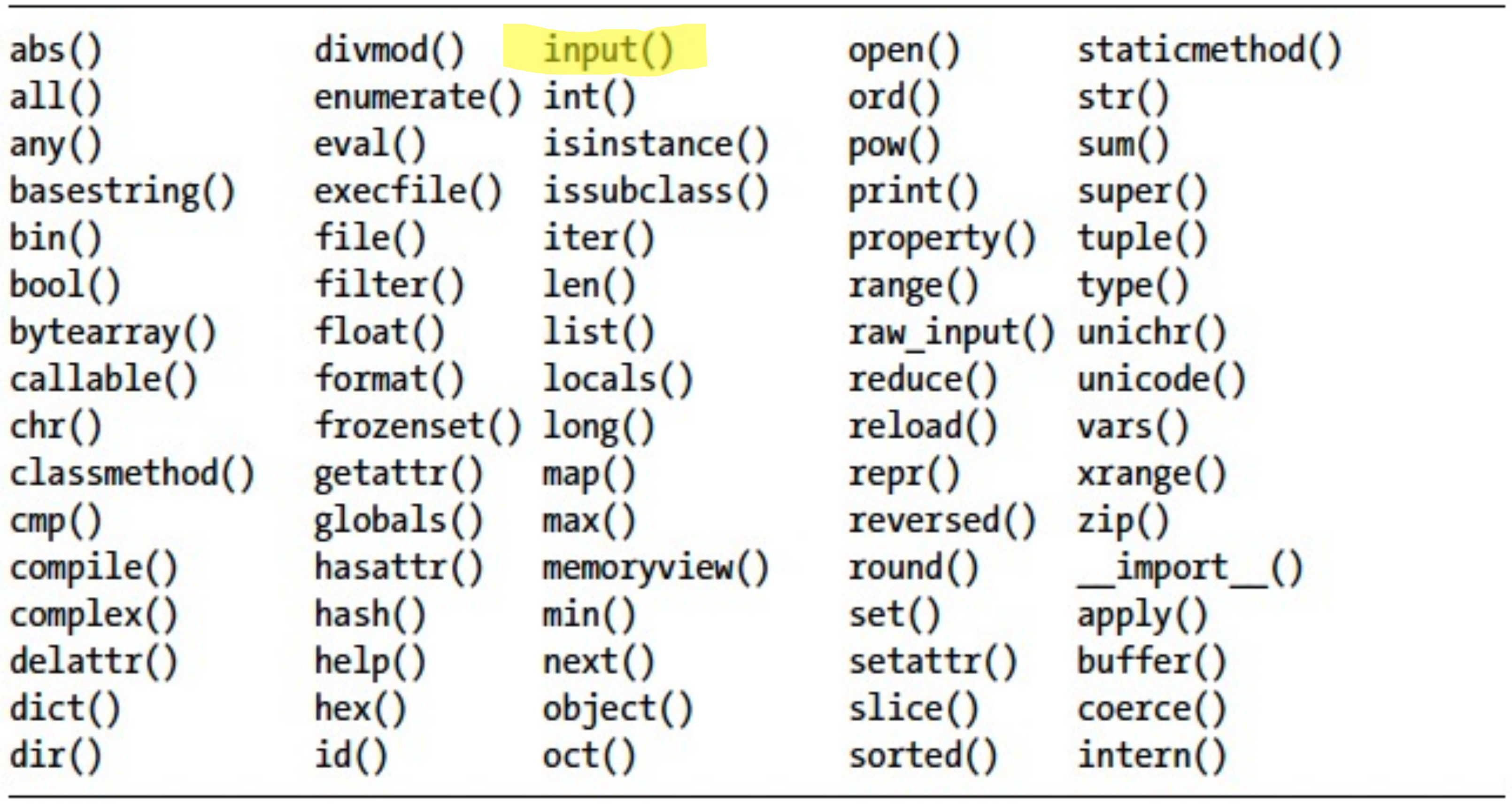
- 6-3 Ordbog med 5 <udtryk> <betydning>
- 6-4 Loop igennem k,v
- 6-8 List of Pets (key: Skully, {kat,”kurt”})

Pause

Input

Interaktion

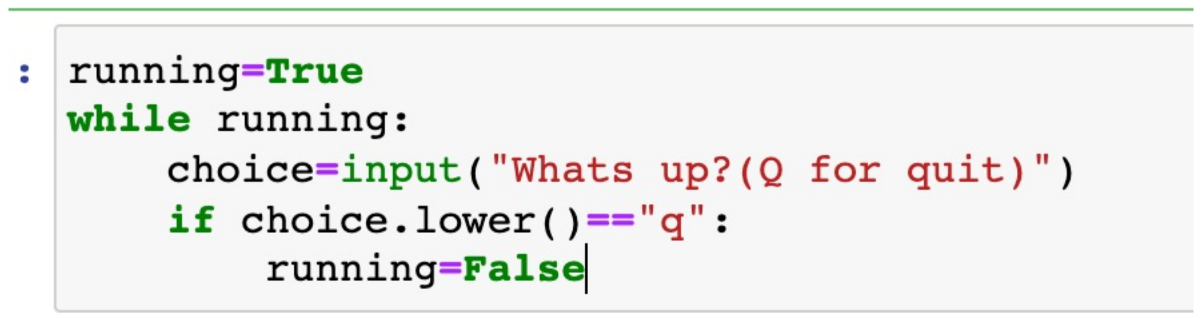
The input() function pauses your program and waits for the user to enter some text.



abs()	divmod()	input()	open()	staticmethod()
all()	enumerate()	int()	ord()	str()
any()	eval()	isinstance()	pow()	sum()
basestring()	execfile()	issubclass()	print()	super()
bin()	file()	iter()	property()	tuple()
bool()	filter()	len()	range()	type()
bytearray()	float()	list()	raw_input()	unichr()
callable()	format()	locals()	reduce()	unicode()
chr()	frozenset()	long()	reload()	vars()
classmethod()	getattr()	map()	repr()	xrange()
cmp()	globals()	max()	reversed()	zip()
compile()	hasattr()	memoryview()	round()	__import__()
complex()	hash()	min()	set()	apply()
delattr()	help()	next()	setattr()	buffer()
dict()	hex()	object()	slice()	coerce()
dir()	id()	oct()	sorted()	intern()

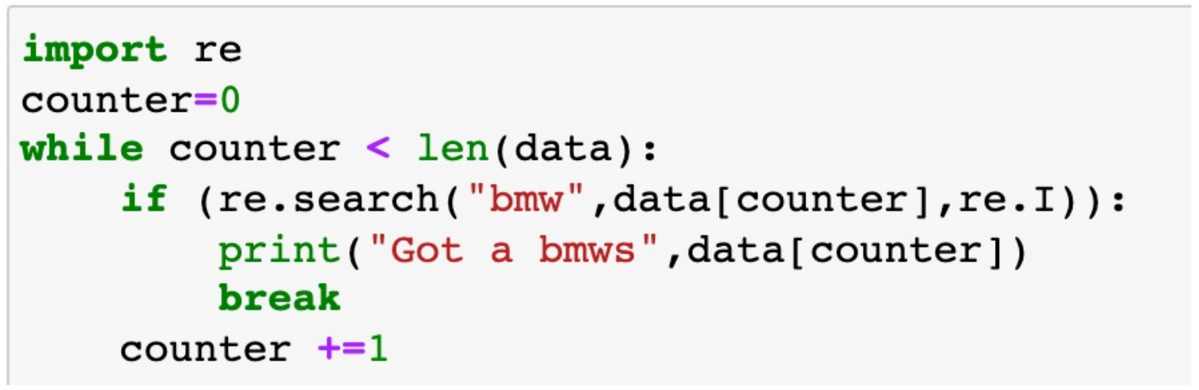
Input

Interaktion

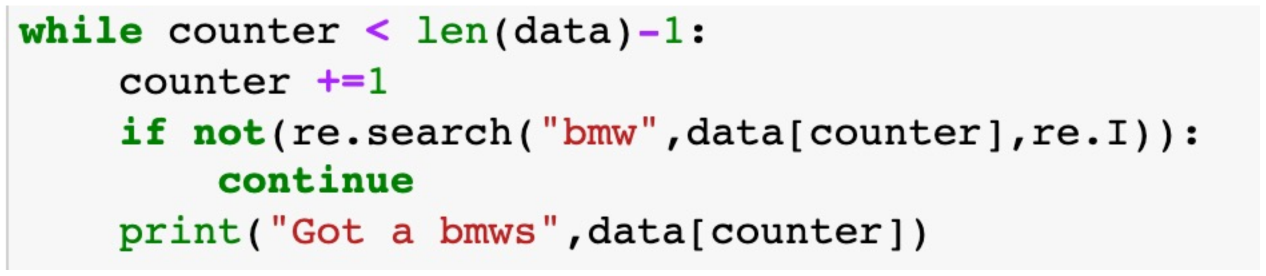


```
:  running=True
   while running:
       choice=input("Whats up?(Q for quit)")
       if choice.lower()=="q":
           running=False
```

Exit,break,
continue



```
import re
counter=0
while counter < len(data):
    if (re.search("bmw",data[counter],re.I)):
        print("Got a bmws",data[counter])
        break
    counter +=1
```



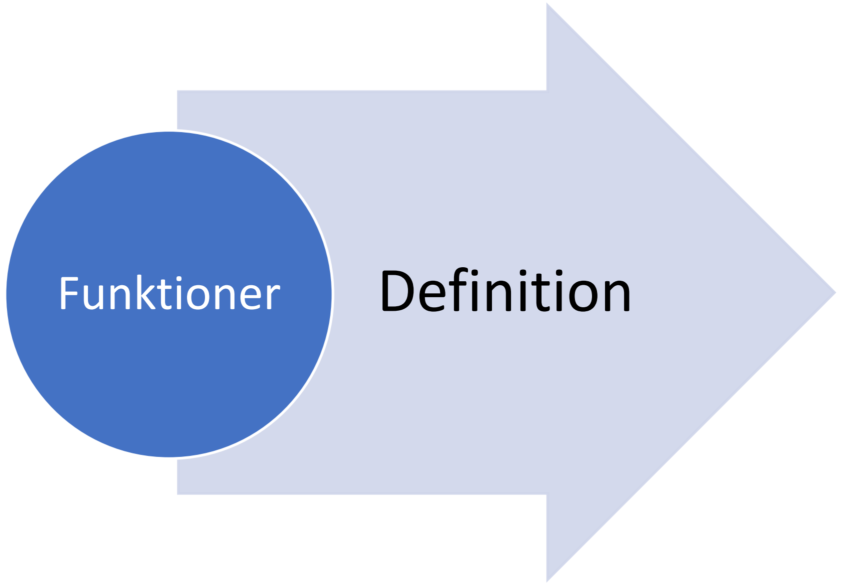
```
while counter < len(data)-1:
    counter +=1
    if not(re.search("bmw",data[counter],re.I)):
        continue
    print("Got a bmws",data[counter])
```

Input

Øvelser

- 7-2 Seats in restaurent (+8 then wait)
- 7-5 Ticket-loop (-3 gratis, +3 er 10, +12 er 15)

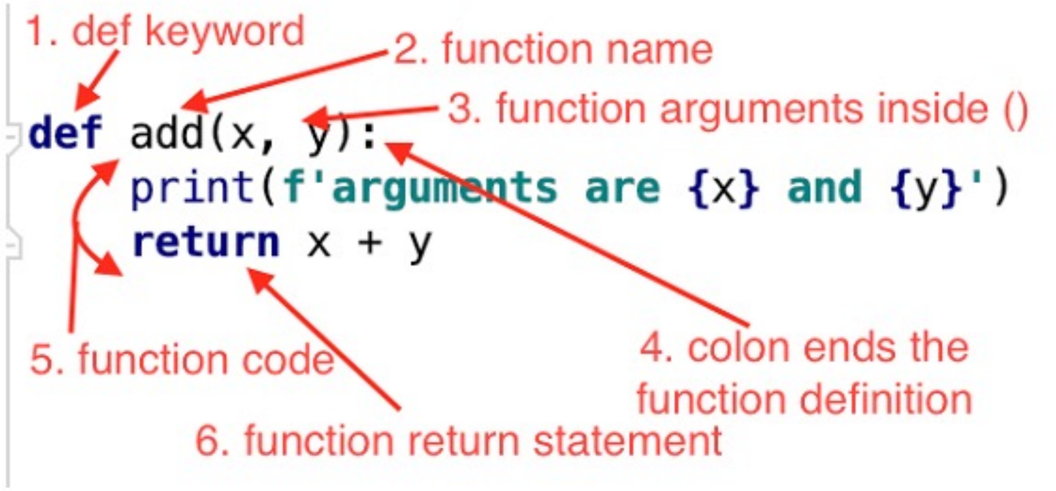
Pause



Function

A named blocks of code that is designed to do one specific job.

Structure of a function



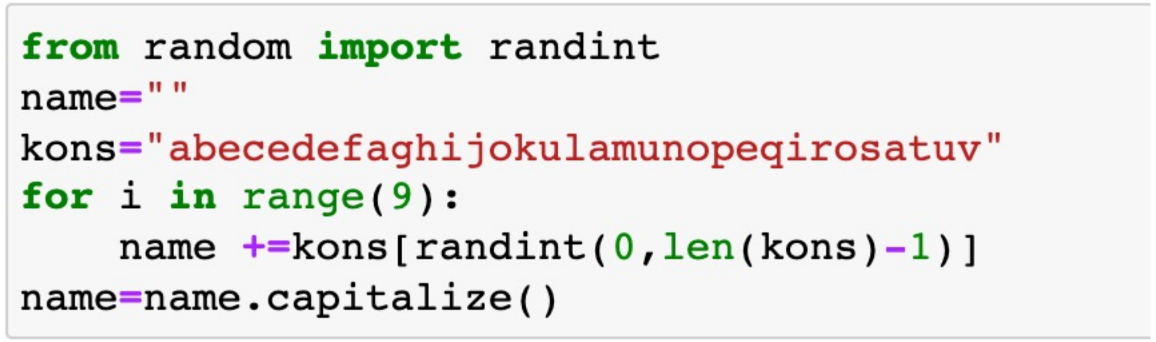
```
def add(x, y):
    print(f'arguments are {x} and {y}')
    return x + y
```

1. def keyword
2. function name
3. function arguments inside ()
4. colon ends the function definition
5. function code
6. function return statement

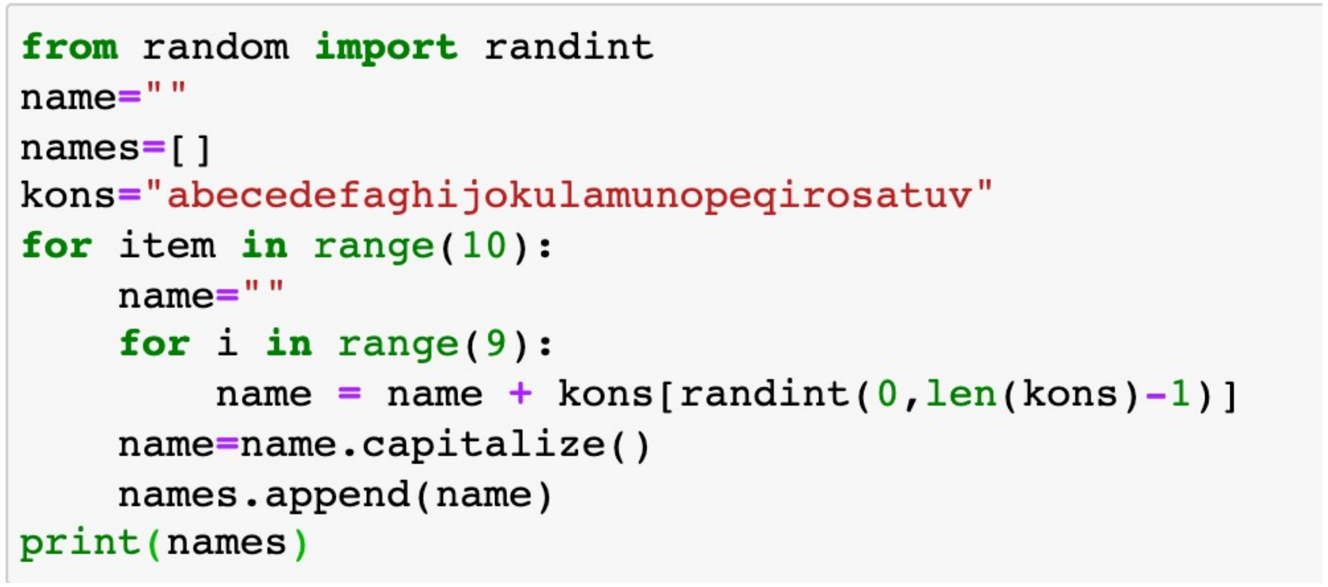
Funktioner

Intro

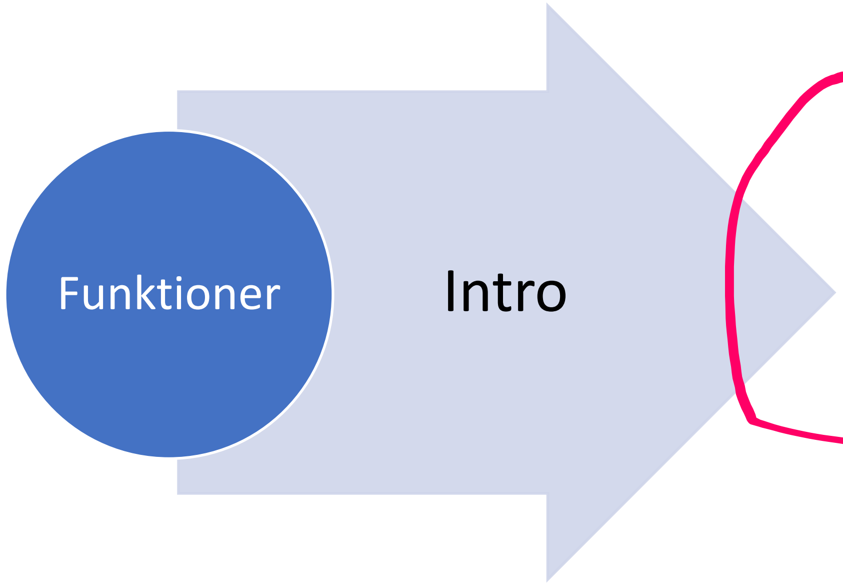
opg1:Vi skal kunne lave tilfældige navne ud fra alfabetet.
Det kan gøres manuelt på flg måde:

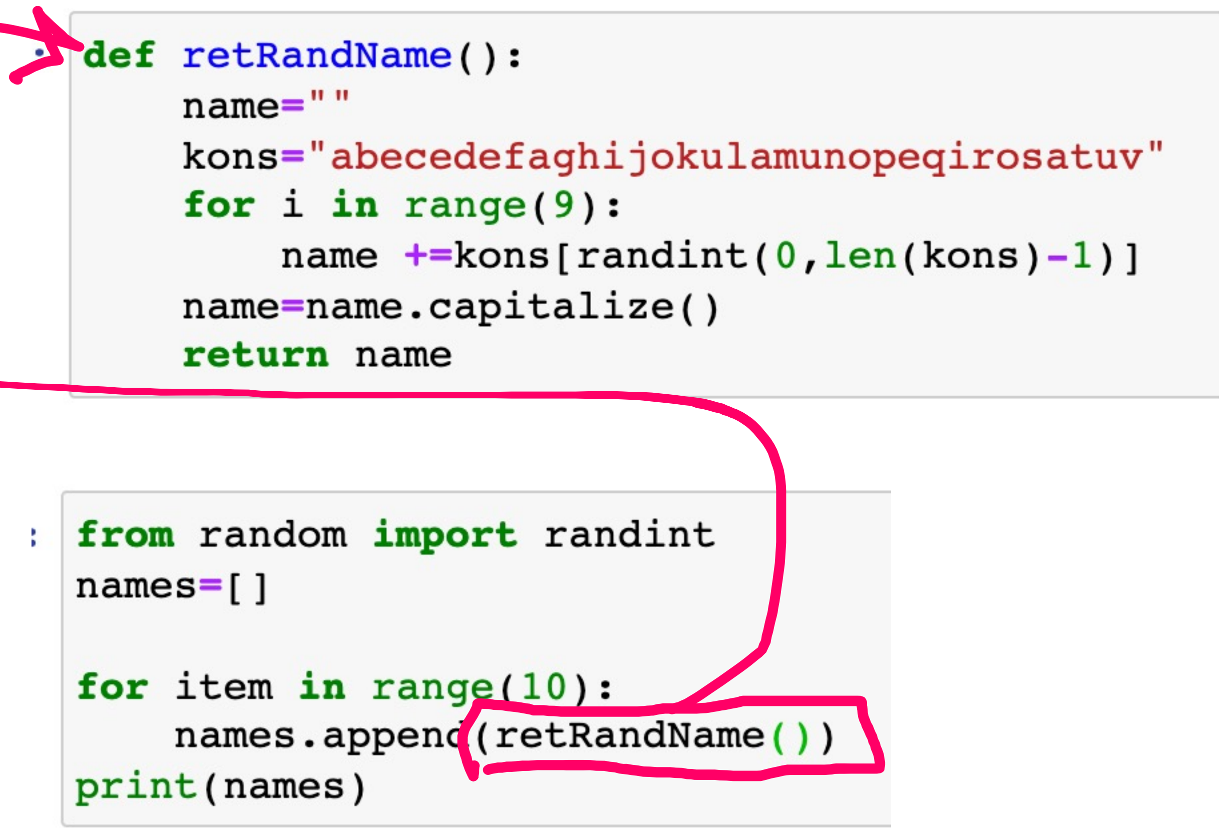


```python
from random import randint
name=""
kons="abecedefaghijokulamunopeqirosatuv"
for i in range(9):
    name +=kons[randint(0,len(kons)-1)]
name=name.capitalize()
```



```python
from random import randint
name=""
names=[]
kons="abecedefaghijokulamunopeqirosatuv"
for item in range(10):
    name=""
    for i in range(9):
        name = name + kons[randint(0,len(kons)-1)]
    name=name.capitalize()
    names.append(name)
print(names)
```



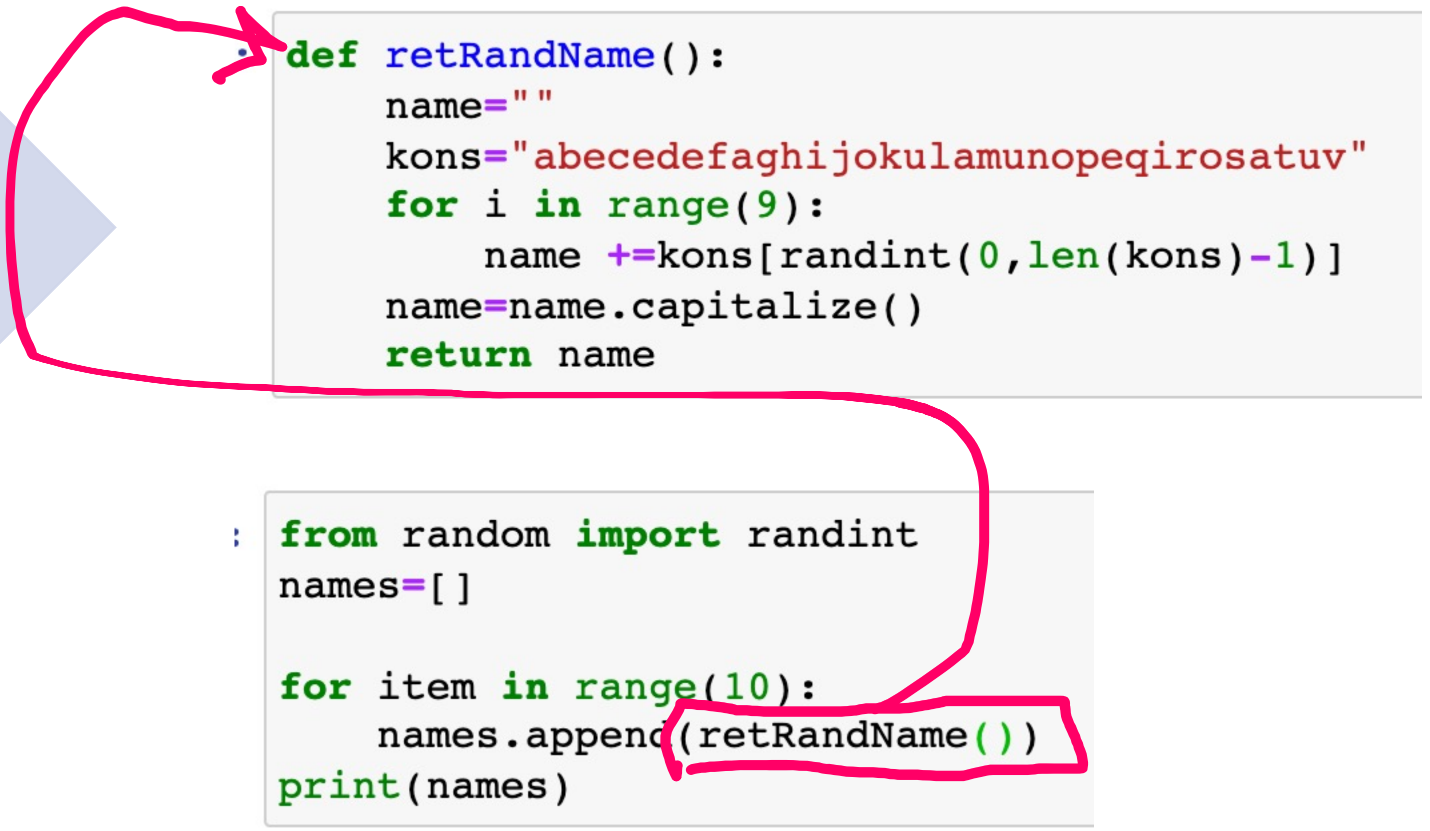


```
def retRandName():
    name=""
    kons="abecedefaghijokulamunopeqirosatuv"
    for i in range(9):
        name +=kons[randint(0,len(kons)-1)]
    name=name.capitalize()
    return name
```

```
from random import randint
names=[]

for item in range(10):
    names.append(retRandName())
print(names)
```

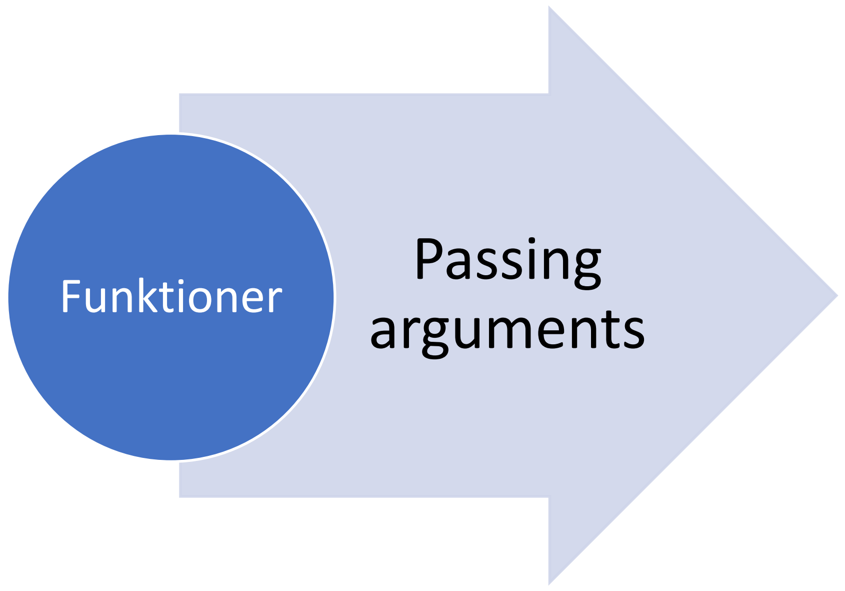
Funktioner

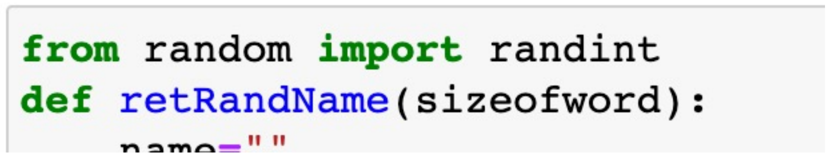
Intro



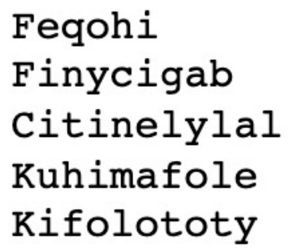
```
def retRandName():
    name=""
    kons="abecedefaghijokulamunopeqirosatuv"
    for i in range(9):
        name +=kons[randint(0,len(kons)-1)]
    name=name.capitalize()
    return name
```

```
from random import randint
names=[]

for item in range(10):
    names.append(retRandName())
print(names)
```





```python
from random import randint
def retRandName(sizeofword):
    name=""
```



```
Feqohi
Finycigab
Citinelylal
Kuhimafole
Kifolototy
```

Ændre funktionen så den tager en parameter
Som angiver længden af ordet.
Sørg for at hvert andet bogstav er en vokal

- Positional arguments
- Keyword arguments
- Default values

Funktioner

Øvelser

- 8-1
- 8-3
- 8-7