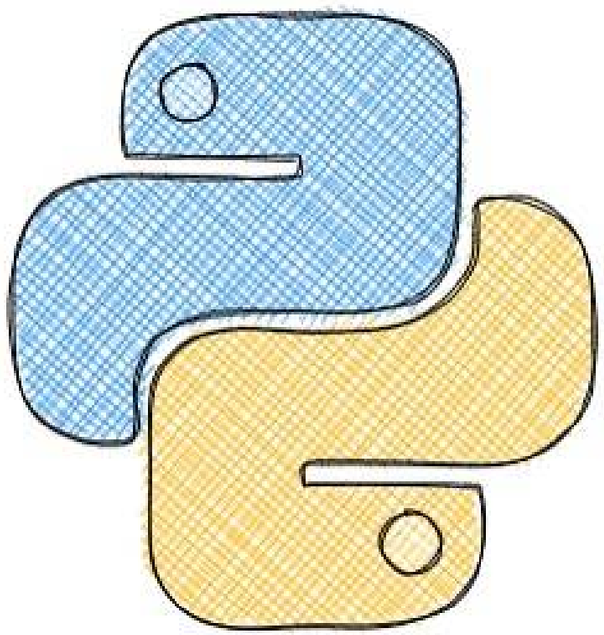


Introduction to Python (Part I)
Variables and data types

Week1

Middlesex University Dubai. Winter ‘23, CST4050
Instructor: Dr. Ivan Reznikov

Plan

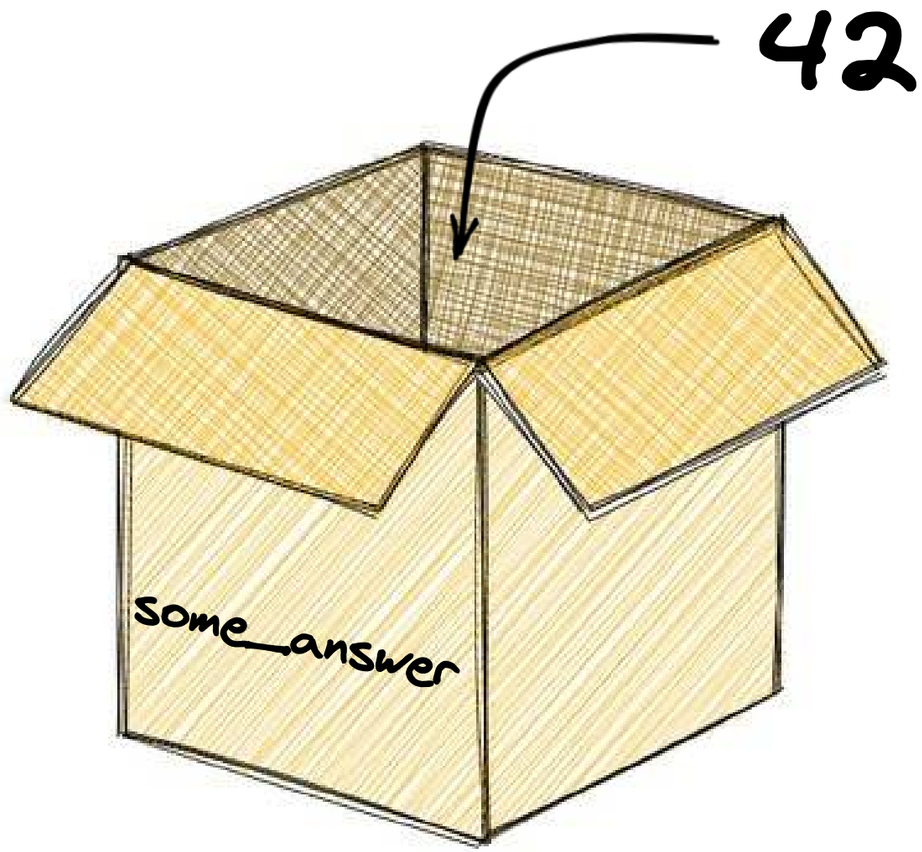
Python Basics:

- Variables
- Comments
- Basic Data Types:
  - String
  - Integer
  - Float
  - Boolean
  - List
  - Dictionary

Variables

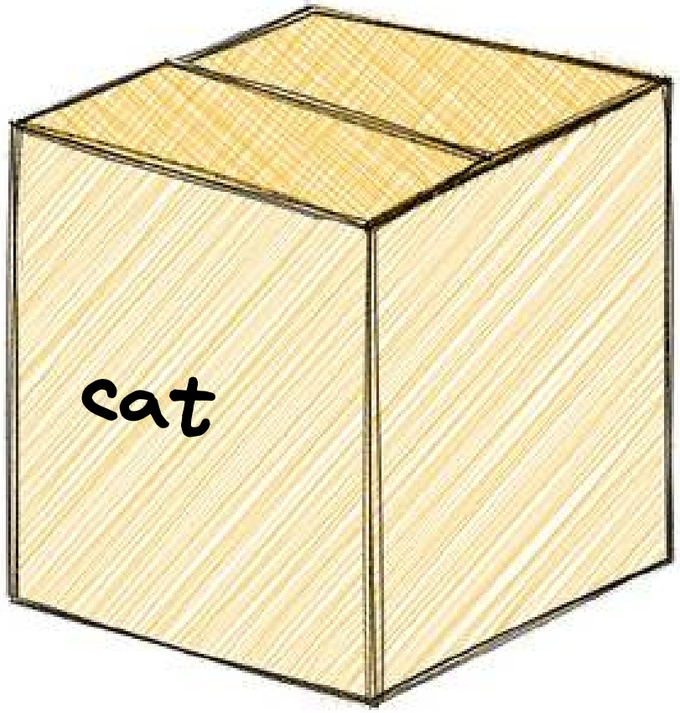
```
some_answer = 42
```

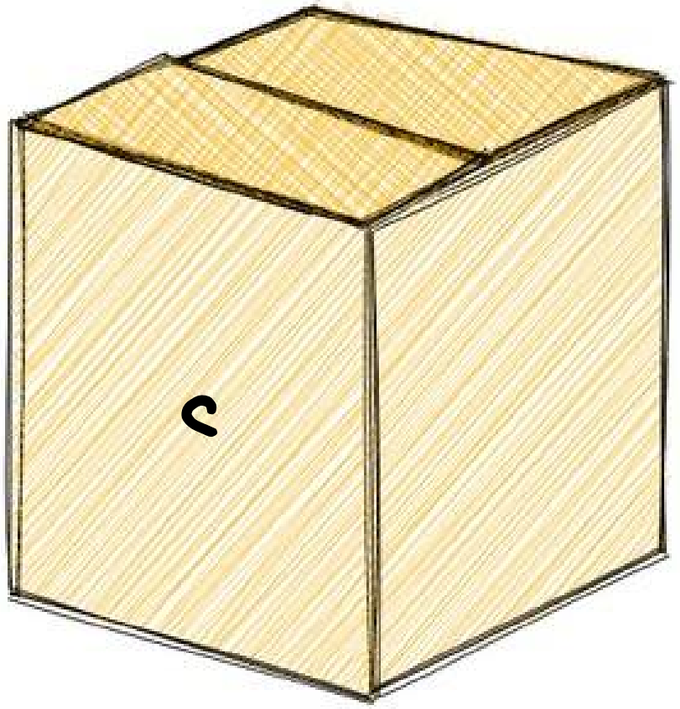
```
print(some_answer)
>>>42

print(type(some_answer))
>>>int
```

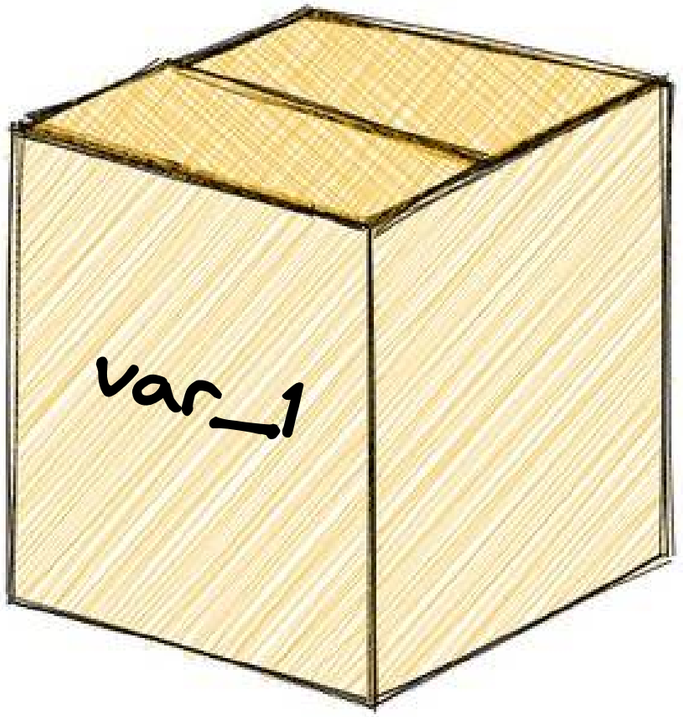


Naming variables

Can you guess what is in each box?







Variables: Naming convention

Bad examples

'l' (lowercase letter “el”),
'O' (uppercase letter “oh”)
'I' (uppercase letter “eye”)

'some_variable'
'NOT_GLOBAL_VAR'
'Averagecamelsizeindesert'
'number-of-cans-in-a-sixpack'
'xxrstaqwe'

Good examples

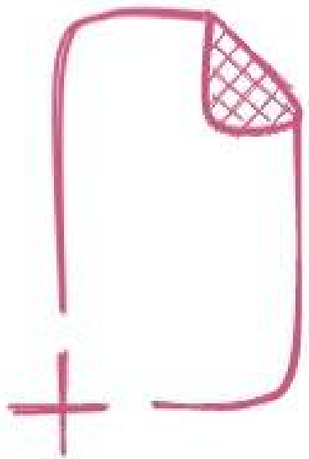
temp_celsius
station_id
current_timestamp

---

tempCelsius
stationID
currentTimestamp

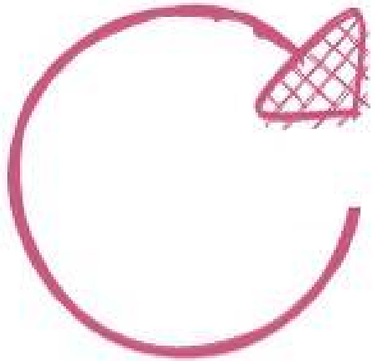
Basic operations with variables

Create

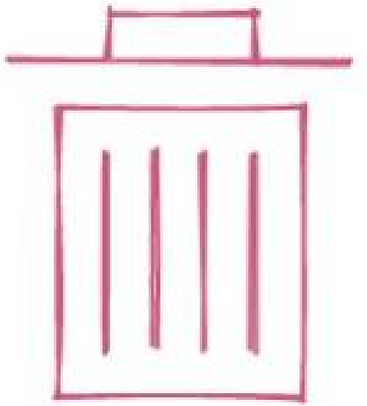


Read

Update



Delete



Comments

```
# This is a comment before some code
print("Hello Python!")
```

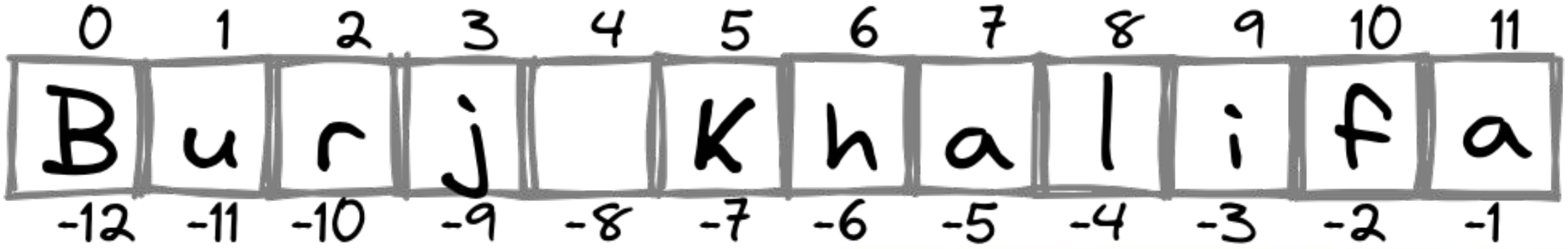
```
'''
multi-row
comment
'''
```

```
print("Winter is coming!") # this is an in-line comment
```

Comments are left to increase the understandability of ones code.
Very often to later version of yourself.

Data Type: String

my_string = "**Burj Khalifa**"



my_string[x_{start}:x_{end}:step]

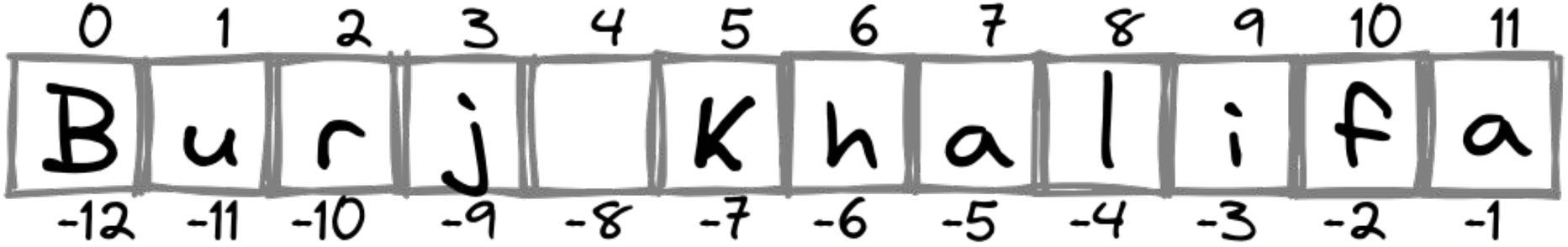
```
my_string[0:4]      >>>"Burj"
my_string[2]        >>>"r"
my_string[-1]       >>>"a"
my_string[2:-5]     >>>"rj Kh"
my_string[::3]      >>>"Bjhi"
```

Speed Task:

```
my_string[5:]       >>>
my_string[4]        >>>
my_string[-0]       >>>
my_string[3:9:2]    >>>
my_string[::-1]     >>>
```

Data Type: String

my_string = "**Burj Khalifa**"



my_string[x_{start}:x_{end}:step]

```
my_string[0:4]      >>>"Burj"
my_string[2]        >>>"r"
my_string[-1]       >>>"a"
my_string[2:-5]     >>>"rj Kh"
my_string[::3]      >>>"Bjhi"
```

Speed Task:

```
my_string[5:]       >>>"Khalifa"
my_string[4]        >>>" "
my_string[-0]       >>>"B"
my_string[3:9:2]    >>>"jKa"
my_string[::-1]
                    >>>"afilahK jruB"
```

Data Type: Updating Strings

String concatenation:

```
string_one = "Expo2020 took"
string_two = "place in 2021"
string_three = string_one + string_two
      >>>Expo2020 tookplace in 2021
```

```
string_three = string_three + " " + "in Dubai"
      >>>Expo2020 tookplace in 2021 in Dubai
```

```
string_three +=  "!"
      >>>Expo2020 tookplace in 2021 in Dubai!
```

String methods

dir(my_string)

>>>[‘__add__’, ‘__class__’, ‘__**contains**__’, ‘__delattr__’, ‘__doc__’, ‘__eq__’, ‘__format__’, ‘__ge__’, ‘__getattribute__’, ‘__getitem__’, ‘__getnewargs__’, ‘__getslice__’, ‘__gt__’, ‘__hash__’, ‘__init__’, ‘__le__’, ‘__**len**__’, ‘__lt__’, ‘__mod__’, ‘__mul__’, ‘__ne__’, ‘__new__’, ‘__reduce__’, ‘__reduce_ex__’, ‘__repr__’, ‘__rmod__’, ‘__rmul__’, ‘__setattr__’, ‘__sizeof__’, ‘__str__’, ‘__subclasshook__’, ‘_formatter_field_name_split’, ‘_formatter_parser’, ‘capitalize’, ‘center’, ‘**count**’, ‘decode’, ‘encode’, ‘**endswith**’, ‘expandtabs’, ‘**find**’, ‘**format**’, ‘index’, ‘isalnum’, ‘isalpha’, ‘isdigit’, ‘islower’, ‘isspace’, ‘istitle’, ‘isupper’, ‘**join**’, ‘ljust’, ‘**lower**’, ‘lstrip’, ‘partition’, ‘**replace**’, ‘rfind’, ‘rindex’, ‘rjust’, ‘rpartition’, ‘rsplit’, ‘rstrip’, ‘**split**’, ‘splitlines’, ‘startswith’, ‘**strip**’, ‘swapcase’, ‘title’, ‘translate’, ‘upper’, ‘zfill’]

Data Type: Integers and Floats

Integer: Whole numbers

```
my_number = 5
type(my_number)
    >>><class 'int'>
int(5.0)
    >>><class 'int'>
my_number  += 1
    >>>6
```

Float: Fraction numbers

```
my_number = 5.0
type(my_number)
    >>><class 'float'>
float(5)
    >>><class 'float'>
del my_number
    >>>
```

Data Type: Boolean

Boolean values represent **True/False** statements.
Often used to check for None, empty strings, list, objects, etc.

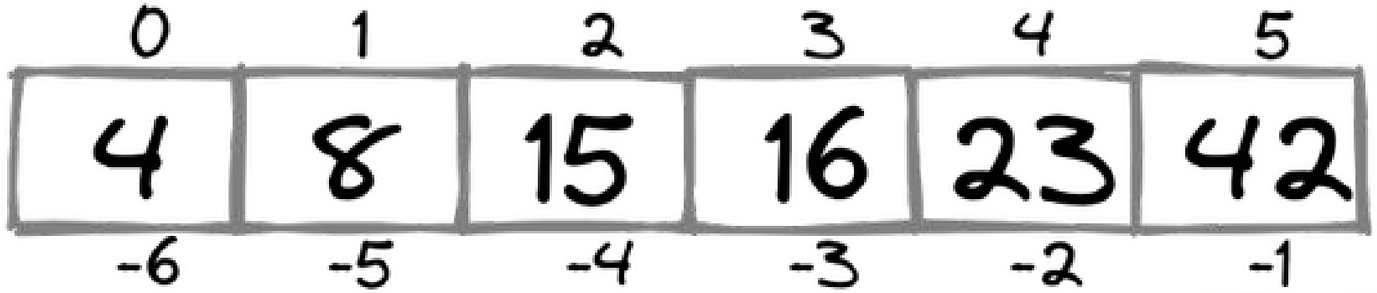
```
my_bool = 2 + 2 == 5
my_bool, type(my_bool)
    >>>False, <class 'bool'>
```

```
bool(False), bool(None), bool(0), bool(""), bool(()), bool([]), bool({})
    >>>False, False, False, False, False, False, False
```

```
bool(True), bool(1), bool("a"), bool((b)), bool([c]), bool({"d":"e"})
    >>>True, True, True, True, True, True
```

Data Type: List

my_string = [4,8,15,16,23,42]



my_list[x_{start}:x_{end}:step]

```
my_list[0]      >>>4
my_list[-1]     >>>42
my_list[1:4]    >>>[8, 15, 16]
my_list[3:]     >>>[16, 23, 42]
my_list[::-1]   >>>[42, 23, 16, 15, 8, 4]
my_list[:]      >>>[4, 8, 15, 16, 23, 42]
```

```
my_list[0] = 1
    >>>[1, 8, 15, 16, 23, 42]

my_list = my_list + [100]
    >>>[1, 8, 15, 16, 23, 42, 100]

del my_list[-2]
    >>>[1, 8, 15, 16, 23, 100]
```

List methods

dir(my_list)

>>>['__add__', '__class__', '__**contains**__', '__delattr__', '__delitem__', '__dir__', '__doc__', '__eq__', '__format__', '__ge__', '__getattribute__', '__getitem__', '__gt__', '__hash__', '__iadd__', '__imul__', '__init__', '__init_subclass__', '__iter__', '__le__', '__**len**__', '__lt__', '__mul__', '__ne__', '__new__', '__reduce__', '__reduce_ex__', '__repr__', '__reversed__', '__rmul__', '__setattr__', '__setitem__', '__sizeof__', '__str__', '__subclasshook__', '**append**', 'clear', '**copy**', '**count**', '**extend**', 'index', 'insert', '**pop**', '**remove**', 'reverse', '**sort**']

Data Type: Tuple

my_tuple = **(16,2,4,8)**

my_tuple[0] = 0
>>>TypeError: 'tuple' object does not support item assignment

my_tuple.append(0)
>>> AttributeError: 'tuple' object has no attribute 'append'

my_tuple[-1] >>>8
my_tuple[1:3] >>>(2, 4)

Tuples are **immutable**. Neither they, nor their elements can be changed.

Tuples are **ordered**. Their order won't change. They maintain the order of data insertion.

Data Type: Set

my_set = **{1, 2, 4, 8, 16, 32}**

```
my_set   = {16,1,2,32,2,2,4,8,16,1,1}
      >>>{1, 2, 4, 8, 16, 32}

my_set.add(5.0)
my_set.add(False)
      >>>{False, 1, 2, 4, 5.0, 8, 16}

my_list = [8, 2, 2, 3, 1, 1, 4, 5, 5, 7, 8]
my_set = set(my_list)
      >>> {1, 2, 3, 4, 5, 7, 8}
```

Sets contain **unique** values. Duplicate values are not allowed.

Sets are **immutable**. Their elements cannot be changed.

Sets are **unordered**. They don't maintain the order of data insertion.

Data Type: Dictionary

```
my_dictionary = {
        "name" : "Burj Khalifa",
        "height_m" : 830,
        "completed" : True
}
```

```
my_dictionary["name"]
        >>>”Burj Khalifa”

my_dictionary.keys()
        >>>dict_keys(['name', 'height_m',
'completed'])

my_dictionary.values()
        >>>dict_values(['Burj Khalifa', 830, True])
```

```
{
        "key": "value"
}
```

Keys are **unique** within a dictionary

Dictionaries are **ordered** by key, not maintaining initial order

Data Type: Dictionary

```
my_dictionary['height_m'] = 829.8

my_dictionary['hashtags'] = ["#burjkhalifa",
      "#dubai", "#uae", "#dubaimall"]

del my_dictionary['completed']

      >>>{
            "name" : "Burj Khalifa",
            "height_m" : 829.8,
            "hashtags" : ["#burjkhalifa",
                  "#dubai", "#uae", "#dubaimall"]
            }
```

```
my_dictionary = {
      "name" : "Burj Khalifa",
      "height_m" : 830,
      "completed" : True
}
```

Dictionary methods

dir(my_dictionary)

>>>['__class__', '__contains__', '__delattr__', '__delitem__', '__dir__', '__doc__', '__eq__', '__format__', '__ge__', '__getattribute__', '__getitem__', '__gt__', '__hash__', '__init__', '__init_subclass__', '__iter__', '__le__', '__len__', '__lt__', '__ne__', '__new__', '__reduce__', '__reduce_ex__', '__repr__', '__reversed__', '__setattr__', '__setitem__', '__sizeof__', '__str__', '__subclasshook__', 'clear', 'copy', 'fromkeys', 'get', '**items**', '**keys**', 'pop', 'popitem', 'setdefault', 'update', '**values**']

Data Type comparison