Python_Basic

September 14, 2023

1 Python Basics

1. Type of variables

1. when you want to assign a value to a variable you just use =

[4]:
```
not_my_int = 5
```

The three types of variables for today 1. int - integer 2. float - non interger **Number** 3. String - Text variable

[6]:
```
my_int = 5
my_float = 5.0
my_string = 'my_string_value'
```

You can use the function: type to see the type

[8]:
```
type(my_int)
```

[8]:
```
int
```

Yet another function: print() print('Yey')

Follow up exercise: 1. Write variables from the three types we saw 2. Use type in 3 different cells to see the types 3. Use print to print your string (we don't know yer how) 4. Use print to print the types

[11]:
```
print('Yey!')
```

```
Yey!
```

[12]:
```
type(my_int)
type(my_float)
type(my_string)
```

[12]:
```
str
```

[15]:
```
Yey = 'YEY!!!!'
```

[16]:
```
print('Yey')
print(Yey)
```

```
Yey
YEY!!!!
```

How to assign a value to a function? - Just put it in the () parentheses How do you call that? That is an **argument**

[ ]:
```
print('Hello World!')
```

Followup: use the print function to print Hello World!

2 Operators:

1. Basic math functions: + , - , / , *

[17]:
```
# int to int --> int
5 + 4
```

[17]:
```
9
```

[18]:
```
# float to int --> float
5.3 - 3
```

[18]:
```
2.3
```

[19]:
```
# int / int --> float
# float / float --> float
6 / 3
```

[19]:
```
2.0
```

[20]:
```
# int * int --> int
6 * 3
```

[20]:
```
18
```

3 Operators on string

Basic operators: +, - , / , *

[21]:
```
my_string + my_string
```

[21]:
```
'my_string_valuemy_string_value'
```

[22]:
```
my_string - my_string
```

```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
<ipython-input-22-46262ea83a9d> in <cell line: 1>()
----> 1 my_string - my_string
```

```
TypeError: unsupported operand type(s) for -: 'str' and 'str'
```

[23]:
```
my_string / my_string
```

```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
<ipython-input-23-164568a00c39> in <cell line: 1>()
----> 1 my_string / my_string

TypeError: unsupported operand type(s) for /: 'str' and 'str'
```

[24]:
```
my_string * my_string
```

```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
<ipython-input-24-f4df06c60ca6> in <cell line: 1>()
----> 1 my_string * my_string

TypeError: can't multiply sequence by non-int of type 'str'
```

[25]:
```
my_string * 3
```

[25]:
```
'my_string_valuemy_string_valuemy_string_value'
```

4 What is a function?

A code part that was **defined** and can operate on an **argument**

[42]:
```
def my_function(my_number):
  return my_number
```

1. **def** –> I'm going to start a function, key words
2. after that we ** must have ** the function name in our case "my_function" can be any name
3. **must** have parentheses –> they will be the argument def
4. In the parentheses you can have an **argument** or not –> (argument), ()
5. **must** have ":"
6. New line
7. Must have at least one space or tab
8. Optional have the **return** key word

5 Bad Functions got to catch them all

1. No name
2. No parentheses

3. No colon ":" at the end of the line
4. No space on or tab on the second line
5. No function block

[2]:
```
# 1. No name
def (my_number):
  return my_number
```

```
  Cell In[2], line 1
    def (my_number):
        ^
SyntaxError: invalid syntax
```

[3]:
```
# 2. No parentheses
# def my_function:
return my_number
```

```
  Cell In[3], line 1
    def my_function:
                    ^
SyntaxError: invalid syntax
```

[31]:
```
# 3. No colon
def my_function
  return my_number
```

```
  File "<ipython-input-31-cae28636d2a3>", line 1
    def my_function:
                    ^
SyntaxError: invalid syntax
```

[4]:
```
# 4. No indentation
def my_function():
return my_number
```

```
  Cell In[4], line 3
    return my_number
    ^
IndentationError: expected an indented block
```

[6]:
```
#5 No function body
def my_function():
```

```
  Cell In[6], line 2
    def my_function():
                      ^
SyntaxError: unexpected EOF while parsing
```

[7]:
```
# 6. No return No problem!!
def my_function():
    print('Hello World!')
```

6 Good functions

[ ]:
```
def my_add_function(a, b):
    return a + b

def my_print_function():
    print('Hello World!')

def my_string_manipulation_function(my_string):
    return my_string + '!!!!'
```

7 Fore-shadowing

How to define a sting 1. “Hello world!” 1. ‘Hello world!’ 1. ”””Hello world”””

Why? The python creator chose it

[37]:
```
'same'
```

[37]: 'same'

[38]:
```
"same"
```

[38]: 'same'

[41]:
```
"""same"""
```

[41]: 'same'

[36]:
```
'''same'''
```

[36]: 'same'

[43]:
```
def my_function(my_number):

  return my_number
```

[44]:
```
my_function(2)
```

[44]:
```
2
```

[47]:
```
my_char = 's'
```

[48]:
```
my_char2 = 'S'
```

[45]:
```
my_string = 'YaY WoW'
```

[46]:
```
my_string
```

[46]:
```
'YaY WoW'
```

[49]:
```
my_string
```

[49]:
```
'YaY WoW'
```

[50]:
```
# New Operator []
my_string[0]
```

[50]:
```
'Y'
```

[51]:
```
print(my_string)
```

```
YaY WoW
```

[55]:
```
my_string[3]
```

[55]:
```
' '
```

[59]:
```
print(my_string[0:4])
print(my_string[4])
```

```
YaY
W
```

[66]:
```
my_string[9]
```

```
---------------------------------------------------------------------------
IndexError                                Traceback (most recent call last)
<ipython-input-66-a8aa16518d4b> in <cell line: 1>()
----> 1 my_string[9]
```

```
IndexError: string index out of range
```

[60]:
```
print(my_string[:4])
print(my_string[0:4])
```

```
YaY
YaY
```

[61]:
```
my_string[-1]
```

[61]:
```
'W'
```

[67]:
```
my_string[-9]
```

```
---------------------------------------------------------------------------
IndexError                                Traceback (most recent call last)
<ipython-input-67-93719b2ddbf7> in <cell line: 1>()
----> 1 my_string[-9]

IndexError: string index out of range
```

[72]:
```
my_string[-3:7]
```

[72]:
```
'WoW'
```

[80]:
```
# Write code that show the string without the space in between YaY and WoW
new_string = my_string[:3] + my_string[-3:]
new_string
```

[80]:
```
'YaYWoW'
```

[84]:
```
my_string[:3]
```

[84]:
```
'YaY'
```

[83]:
```
my_string[3:]
```

[83]:
```
' WoW'
```