PROJECT #2 & GITHUB PULL REQUESTS

petrsebek1

LAST LESSON

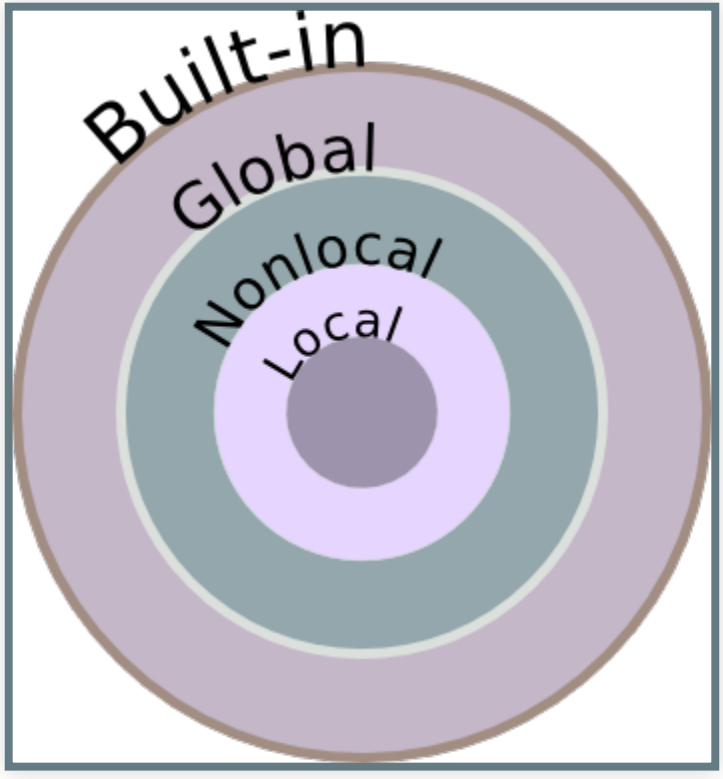
SCOPES

```
>>> name = 'John'
>>> def func():
...     print(name)
>>> func()
John
```

SCOPES

```
>>> def func():
...     name = 'John'
...     print(name)
>>> print(name, 'Smith')
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
NameError: name 'name' is not defined
```

SCOPES



BUILT-IN SCOPE

- Built-in types
- Built-in functions

```
>>> import pprint
>>> pprint.pprint(__builtins__.__dict__)
{'ArithmeticError': <class 'ArithmeticError'>,
 'AssertionError': <class 'AssertionError'>,
 'AttributeError': <class 'AttributeError'>,
...
 'abs': <built-in function abs>,
 'all': <built-in function all>,
 'any': <built-in function any>,
...
 'tuple': <class 'tuple'>,
 'type': <class 'type'>,
 'vars': <built-in function vars>,
 'zip': <class 'zip'>}
```

GLOBAL

```
name = 'John'

def print_name():
        print(name)

print_name()
```

```
John
```

GLOBAL

```
name = 'John'

def print_name():
        print(name)

print_name()
name = 'Bob'
print_name()
```

```
John
Bob
```

GLOBAL

```
name = 'John'

def print_name():
        name = 'Bob'
        print(name)

print_name()
print(name)
```

```
Bob
John
```

GLOBAL

```
name = 'John'

def print_name():
        name = 'Bob'
        print('Locals:', locals())
        print('Globals:', globals())
        print(name)

print_name()
print(name)
```

```
Locals: {'name': 'Bob'}
Globals {
        '__name__': '__main__',
        '__builtins__': <module 'builtins' (built-in)>,
        '__file__': '/home/psebek/projects/engeto/07/global.py',
        'name': 'John',
        'print_name': <function print_name at 0x7f8751945ef0>
        ...
}
Bob
John
```

GLOBAL

```
name = 'John'

def print_name():
        global name # <---- global keyword
        name = 'Bob'
        print(name)

print_name()
print(name)
```

```
Bob
Bob
```

NESTED FUNCTIONS

```
import random

def wrapper():
    start = random.randint(1, 5)
    end = random.randint(5, 10)
    print('start = ', start, ', end = ', end)
    def inner():
        return random.randint(start, end)
    return inner

random_limits = wrapper()
print('#' * 20)
print(random_limits())
print('-' * 20)
print(random_limits())
```

```
start =  3, end = 8
####################
6
--------------------
3
```

NESTED FUNCTIONS

```
import random

def wrapper():
    start = random.randint(1, 5)
    end = random.randint(5, 10)
    print('start = ', start)
    print('end = ', end)

    def inner():
        nonlocal start
        nonlocal end
        start = end = 12
        return random.randint(start, end)

    return inner
```

```
start =  2
end =  5
####################
12
--------------------
12
```

FUNCTION ARGUMENTS

```
>>> def function(argument_1, argument_2):
...     print(argument_1, argument_2)
>>> function(1, 2)
1 2
```

KEYWORD ARGUMENTS

```
>>> def function(argument, default_argument = 2):
...     print(argument, default_argument)
```

```
>>> function(1)
1 2
```

```
>>> function(1, 3)
1 3
```

```
>>> function(1, default_argument = 4)
1 4
```

```
>>> function(argument = 2, default_argument = 5)
2 5
```

```
>>> function(default_argument = 5, argument = 3)
3 5
```

VARIABLE NUMBER OF ARGUMENTS *

```
>>> def function(*args):
...     print(args)
```

```
>>> function(1, 2, 3)
(1, 2, 3)
```

```
>>> function()
()
```

```
>>> l = [1, 2, 3]
>>> function(*l)
(1, 2, 3)
```

VARIABLE NUMBER OF KEYWORD ARGUMENTS **

```
>>> def function(**kwargs):
...     print(kwargs)
```

```
>>> function(a = 1, b = 2)
{'a': 1, 'b': 2}
```

```
>>> function()
{}
```

```
>>> arguments = {'c': 3, 'd': 4}
>>> function(**arguments)
{'c': 3, 'd': 4}
```

ULTIMATE FORM OF ARGUMENTS

```
>>> def function(a, b, *args, default = None, **kwargs):
...     print(a, b, args, default, kwargs)
```

```
>>> function(1, 2, 3, 4, 5, key = 'KEY')
1 2 (3, 4, 5) None {'key': 'KEY'}
```

NEVER USE MUTABLES AS DEFAULTS

```
>>> def append(item, l = []):
...     l.append(item)
...     return l

>>> i = append(1)
>>> print(i)
[1]
>>> j = append(2)
>>> print(j)
[1, 2]
>>> print(i)
[1, 2]
```

NEVER USE MUTABLES AS DEFAULTS

```
>>> def append(item, l = None):
...     if l is None:
...         l = []
...     l.append(item)
...     return l

>>> i = append(1)
>>> print(i)
[1]
>>> j = append(2)
>>> print(j)
[2]
>>> print(i)
[1]
```

TODAY'S LESSON

STRING FORMATTING

```
>>> name = 'Guido'
>>> surname = 'van Rossum'
>>> age = 63
>>> print('Name:', name, ', surname:', surname, ', age:', age, '!
Name: Guido , surname: van Rossum , age: 63 !
```

```
>>> print('Name: ' + name + ', surname: ' + surname + ', age: ' +
Name: Guido, surname: van Rossum, age: 63!
```

```
>>> print(f'Name: {name}, surname: {surname}, age: {age}!')
Name: Guido, surname: van Rossum, age: 63!
```

F-STRINGS – WIDTH, ALIGNMENT AND FILL

https://realpython.com/python-f-strings/

```
>>>> f'{name:10}'
'Guido     '
```

```
>>>> f'{name:>10}'
'     Guido'
```

```
>>>> f'{name:^10}'
'  Guido   '
```

```
>>>> f'{name:.^10}'
'..Guido...'
```

```
>>> width = 10
>>>> f'{name:.^{width}}'
'..Guido...'
```

F-STRINGS – NUMBER FORMAT

```
>>> price = 123.4567890
>>> f'{price}'
'123.456789'
```

```
>>> f'{price:.2}'
'1.2e+02'
```

```
>>> f'{price:.2f}'
'123.46'
```

```
>>> f'{price:e}'
'1.234568e+02'
```

```
>>> f'{price:+.2f}'
'+123.46'
```

```
>>> negative_price = -price
>>> f'{price: .2f} vs. {negative_price: .2f}'
' 123.46 vs. -123.46'
```

F-STRINGS – EXPRESSIONS

```
>>> f'Surname length: {len(surname)}'
'Surname length: 10'
```

```
>>> f'lower full name: {name.lower()} {surname.lower()}'
'lower full name: guido van rossum'
```

Don't complicate f-strings

```
>>> import random
>>> f'Over threshold: {"yes" if random.random() > 0.3 else "no"}'
'Over threshold: yes'
```

```
>>> import random
>>> over_threshold = 'yes' if random.random() > 0.3 else 'no'
>>> f'Over threshold: {over_threshold}'
'Over threshold: no'
```

.format()

```
>>> 'Name: {}, surname: {}'.format(name, surname)
'Name: Guido, surname: van Rossum'
```

```
>>> 'Name: {name}, surname: {surname}'.format(name = name, surnam
'Name: Guido, surname: van Rossum'
```

```
>>> credentials = {'name': 'Guido', 'surname': 'van Rossum'}
>>> f'Name: {credentials["name"]}, surname: {credentials["surname
```

```
>>> 'Name: {name}, surname: {surname}'.format(**credentials)
'Name: Guido, surname: van Rossum'
```

EXERCISE

```
dataset = [
        ['Name', 'Item', 'Amount', 'Price', 'Total'],
        ['Bettison, Elnora', 'Doxycycline Hyclate', 98, 23.43, 22
        ['McShee, Glenn', 'DROXIA', 27, 33.86, 914.22],
        ['Conyard, Phil', 'Nadolol', 44, 12.35, 543.4],
]
```

Output:

```
=============================================================
||          Name          |          Item          | Amount | Price
=============================================================
|| Bettison, Elnora       | Doxycycline Hyclate    |     98 |  23.43
|| McShee, Glenn          | DROXIA                 |     27 |  33.86
|| Conyard, Phil          | Nadolol                |     44 |  12.35
=============================================================
```

format_header

```
>>> format_header(dataset[0])
'||          Name          |          Item          | Amount |   Price
```

- Aligned to the center
- Name, Item width 20
- Amount, Price width 6
- Total width 8

format_row

```
>>> format_row(dataset[1])
'|| Bettison, Elnora     | Doxycycline Hyclate  |     98 |  23.43
```

- Name, Item width 20, aligned to the left
- Amount, Price width 6, aligned to the rigth
- Total width 8, aligned to the right

format_table

```
>>> print(format_table(dataset))
=================================================================
||          Name          |          Item          | Amount | Price
=================================================================
|| Bettison, Elnora       | Doxycycline Hyclate    |     98 |  23.43
|| McShee, Glenn          | DROXIA                 |     27 |  33.86
|| Conyard, Phil          | Nadolol                |     44 |  12.35
|| Bettison, Elnora       | Claravis               |     91 |   9.85
|| Idalia, Craig          | Nadolol                |     83 |  12.35
|| Woodison, Annie        | Metolazone             |     46 |  43.06
|| Woodison, Annie        | DROXIA                 |     50 |  33.86
|| Skupinski, Wilbert     | Nadolol                |     60 |  12.35
=================================================================
```

FILES

FILE PATH

Windows

```
C:\Users\Bob\Documents\gifts.txt
```

Linux + Mac

```
/home/Bob/Documents/gifts.txt
```

ABSOLUTE VS. RELATIVE PATH

```
/home/Bob $ pwd
/home/Bob
/home/Bob $ cd Documents
/home/Bob/Documents $ cat gifts.txt # relative path
gifts file content
/home/Bob/Documents $ cat /home/Bob/Documents/gitfs.txt # absolut
gifts file content
/home/Bob/Documents $ cd .. # parent directory
/home/Bob $ cat Documents/gifts.txt # relative path
```

FILE OPEN, READ AND CLOSE

```
file = open('gifts.txt')
content = file.read()
print(content)
file.close()
```

Always use context manager

```
>>> with open('gifts.txt') as file:
...     content = file.read()
... print(content)
VR - 1000 USD
JetPack - 30000 USD
Book - 30 USD
```

OPEN NON-EXISTENT FILE

```
>>> with open('g.txt.') as file:
...     content = file.read()
...
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
FileNotFoundError: [Errno 2] No such file or directory: 'g.txt.'
```

FILE WRITE

```
with open('gifts.txt', mode='w') as file:
        file.write('Overwriting content of the file')
```

OPEN FOR WRITING TO NON-EXISTENG FILE

```
>>> with open('g.txt', 'w') as file:
...     file.write('Hello world')
...
11
```

FILE OPEN MODES

`'r'`
: open for reading text, default

`'w'`
: truncate and open for writing text

`'a'`
: open for writing text, append at the end

`'r+'`
: open for reading and writing text

`'w+'`
: truncate and open for reading and writing text

`'rb'`
: open for reading bytes

`'wb'`
: truncate and open for writing bytes

EXERCISE

INTERACTIVE_COPY.PY

1. Ask user for a path to a file to read.
2. Open the file and read its content.
3. Ask user for a path to a file to write.
4. Write content of the first file to the second.

```
$ python interactive_copy.py
Read file: gifts.txt
Write file: gifts_2.txt
```

READING FILE LINE BY LINE

```
>>> with open('gifts.txt') as file:
...     for line in file:
...         print('#', line, end='')
...
# VR - 1000 USD
# JetPack - 30000 USD
# Book - 30 USD
```

END