(/)

Curriculum
SE Foundations
Average: **58.48%**

You have a captain's log due before 2024-05-12 (in 3 days)! Log it now!
(/captain_logs/5661831/edit)

0x04. Python - More Data Structures: Set, Dictionary

Python

Weight: 1

Project over - took place from Dec 6, 2023 6:00 AM to Dec 7, 2023 6:00 AM

An auto review will be launched at the deadline

In a nutshell…

- **Auto QA review:** 7.15/139 mandatory & 0.0/48 optional
- **Altogether: 5.14%**
  - Mandatory: 5.14%
  - Optional: 0.0%
  - Calculation: 5.14% + (5.14% * 0.0%) == **5.14%**

Resources

Read or watch:

- Data structures (/rltoken/GmgoSUtBbHBW8suWkws51g)
- Lambda, filter, reduce and map (/rltoken/53f4kKVT0-jyzrJstOSJWg)
- Learn to Program 12 Lambda Map Filter Reduce (/rltoken/v9eyFryhkYmxDI13iTx2VA)

man or help:

- `python3`

Learning Objectives

At the end of this project, you are expected to be able to explain to anyone (/rltoken/nbatZmfQyeB03w9ipyFhSw), **without the help of Google**:

General

- Why Python programming is awesome
- What are sets and how to use them
- What are the most common methods of set and how to use them
- When to use sets versus lists
- How to iterate into a set
- What are dictionaries and how to use them
- When to use dictionaries versus lists or sets
- What is a key in a dictionary
- How to iterate over a dictionary
- What is a lambda function
- What are the map, reduce and filter functions

Copyright - Plagiarism

- You are tasked to come up with solutions for the tasks below yourself to meet with the above learning objectives.
- You will not be able to meet the objectives of this or any following project by copying and pasting someone else's work.
- You are not allowed to publish any content of this project.
- Any form of plagiarism is strictly forbidden and will result in removal from the program.

Requirements

General

- Allowed editors: `vi`, `vim`, `emacs`
- All your files will be interpreted/compiled on Ubuntu 20.04 LTS using python3 (version 3.8.5)
- All your files should end with a new line
- The first line of all your files should be exactly `#!/usr/bin/python3`
- A `README.md` file, at the root of the folder of the project, is mandatory
- Your code should use the pycodestyle (version `2.8.*`)
- All your files must be executable
- The length of your files will be tested using `wc`

Quiz questions

Great! You've completed the quiz successfully! Keep going! (Show quiz)

Tasks

0. Squared simple

mandatory

Score: 65.0% (*Checks completed: 100.0%*)

Write a function that computes the square value of all integers of a matrix.

- Prototype: `def square_matrix_simple(matrix=[]):`
- `matrix` is a 2 dimensional array
- Returns a new matrix:
  - Same size as `matrix`
  - Each value should be the square of the value of the input
- Initial matrix should not be modified
- You are not allowed to import any module
- You are allowed to use regular loops, `map` , etc.

```
guillaume@ubuntu:~/0x04$ cat 0-main.py
#!/usr/bin/python3
square_matrix_simple = __import__('0-square_matrix_simple').square_matrix_simple

matrix = [
    [1, 2, 3],
    [4, 5, 6],
    [7, 8, 9]
]

new_matrix = square_matrix_simple(matrix)
print(new_matrix)
print(matrix)

guillaume@ubuntu:~/0x04$ ./0-main.py
[[1, 4, 9], [16, 25, 36], [49, 64, 81]]
[[1, 2, 3], [4, 5, 6], [7, 8, 9]]
guillaume@ubuntu:~/0x04$
```

Repo:

- GitHub repository: `alx-higher_level_programming`
- Directory: `0x04-python-more_data_structures`
- File: `0-square_matrix_simple.py`

Done! | Check your code | Get a sandbox | QA Review

1. Search and replace

mandatory

Score: 0.0% (*Checks completed: 0.0%*)

Write a function that replaces all occurrences of an element by another in a new list.

- Prototype: `def search_replace(my_list, search, replace):`
- `my_list` is the initial list
- `search` is the element to replace in the list
- `replace` is the new element
- You are not allowed to import any module

```
guillaume@ubuntu:~/0x04$ cat 1-main.py
#!/usr/bin/python3
search_replace = __import__('1-search_replace').search_replace

my_list = [1, 2, 3, 4, 5, 4, 2, 1, 1, 4, 89]
new_list = search_replace(my_list, 2, 89)

print(new_list)
print(my_list)

guillaume@ubuntu:~/0x04$ ./1-main.py
[1, 89, 3, 4, 5, 4, 89, 1, 1, 4, 89]
[1, 2, 3, 4, 5, 4, 2, 1, 1, 4, 89]
guillaume@ubuntu:~/0x04$
```

Repo:

- GitHub repository: `alx-higher_level_programming`
- Directory: `0x04-python-more_data_structures`
- File: `1-search_replace.py`

Done? | Check your code | Ask for a new correction | Get a sandbox | QA Review

2. Unique addition

mandatory

Score: 0.0% (*Checks completed: 0.0%*)

Write a function that adds all unique integers in a list (only once for each integer).

- Prototype: `def uniq_add(my_list=[]):`
- You are not allowed to import any module

(7)

```
guillaume@ubuntu:~/0x04$ cat 2-main.py
#!/usr/bin/python3
uniq_add = __import__('2-uniq_add').uniq_add

my_list = [1, 2, 3, 1, 4, 2, 5]
result = uniq_add(my_list)
print("Result: {:d}".format(result))

guillaume@ubuntu:~/0x04$ ./2-main.py
Result: 15
guillaume@ubuntu:~/0x04$
```

Repo:

- GitHub repository: `alx-higher_level_programming`
- Directory: `0x04-python-more_data_structures`
- File: `2-uniq_add.py`

Done? | Check your code | Ask for a new correction | Get a sandbox | QA Review

3. Present in both

mandatory

Score: 0.0% (*Checks completed: 0.0%*)

Write a function that returns a set of common elements in two sets.

- Prototype: `def common_elements(set_1, set_2):`
- You are not allowed to import any module

```
guillaume@ubuntu:~/0x04$ cat 3-main.py
#!/usr/bin/python3
common_elements = __import__('3-common_elements').common_elements

set_1 = { "Python", "C", "Javascript" }
set_2 = { "Bash", "C", "Ruby", "Perl" }
c_set = common_elements(set_1, set_2)
print(sorted(list(c_set)))

guillaume@ubuntu:~/0x04$ ./3-main.py
['C']
guillaume@ubuntu:~/0x04$
```

Repo:

- GitHub repository: `alx-higher_level_programming`
- Directory: `0x04-python-more_data_structures`
- File: `3-common_elements.py`

(/) Done? | Check your code | Ask for a new correction | >_ Get a sandbox | QA Review

4. Only differents

mandatory

Score: 0.0% (*Checks completed: 0.0%*)

Write a function that returns a set of all elements present in only one set.

- Prototype: `def only_diff_elements(set_1, set_2):`
- You are not allowed to import any module

```
guillaume@ubuntu:~/0x04$ cat 4-main.py
#!/usr/bin/python3
only_diff_elements = __import__('4-only_diff_elements').only_diff_elements

set_1 = { "Python", "C", "Javascript" }
set_2 = { "Bash", "C", "Ruby", "Perl" }
od_set = only_diff_elements(set_1, set_2)
print(sorted(list(od_set)))

guillaume@ubuntu:~/0x04$ ./4-main.py
['Bash', 'Javascript', 'Perl', 'Python', 'Ruby']
guillaume@ubuntu:~/0x04$
```

Repo:

- GitHub repository: `alx-higher_level_programming`
- Directory: `0x04-python-more_data_structures`
- File: `4-only_diff_elements.py`

☐ Done? | Check your code | Ask for a new correction | >_ Get a sandbox | QA Review

5. Number of keys

mandatory

Score: 0.0% (*Checks completed: 0.0%*)

Write a function that returns the number of keys in a dictionary.

- Prototype: `def number_keys(a_dictionary):`
- You are not allowed to import any module

```
guillaume@ubuntu:~/0x04$ cat 5-main.py
#!/usr/bin/python3
number_keys = __import__('5-number_keys').number_keys

a_dictionary = { 'language': "C", 'number': 13, 'track': "Low level" }
nb_keys = number_keys(a_dictionary)
print("Number of keys: {:d}".format(nb_keys))

guillaume@ubuntu:~/0x04$ ./5-main.py
Number of keys: 3
guillaume@ubuntu:~/0x04$
```

Repo:

- GitHub repository: `alx-higher_level_programming`
- Directory: `0x04-python-more_data_structures`
- File: `5-number_keys.py`

Done? | Check your code | Ask for a new correction | Get a sandbox | QA Review

6. Print sorted dictionary

mandatory

Score: 0.0% (*Checks completed: 0.0%*)

Write a function that prints a dictionary by ordered keys.

- Prototype: `def print_sorted_dictionary(a_dictionary):`
- You can assume that all keys are strings
- Keys should be sorted by alphabetic order
- Only sort keys of the first level (don't sort keys of a dictionary inside the main dictionary)
- Dictionary values can have any type
- You are not allowed to import any module

```
guillaume@ubuntu:~/0x04$ cat 6-main.py
#!/usr/bin/python3
print_sorted_dictionary = __import__('6-print_sorted_dictionary').print_sorted_di
ctionary

a_dictionary = { 'language': "C", 'Number': 89, 'track': "Low level", 'ids': [1,
2, 3] }
print_sorted_dictionary(a_dictionary)

guillaume@ubuntu:~/0x04$ ./6-main.py
Number: 89
ids: [1, 2, 3]
language: C
track: Low level
guillaume@ubuntu:~/0x04$
```

Repo:

- GitHub repository: `alx-higher_level_programming`
- Directory: `0x04-python-more_data_structures`
- File: `6-print_sorted_dictionary.py`

Done? | Check your code | Ask for a new correction | Get a sandbox | QA Review

7. Update dictionary

mandatory

Score: 0.0% (*Checks completed: 0.0%*)

Write a function that replaces or adds key/value in a dictionary.

- Prototype: `def update_dictionary(a_dictionary, key, value):`
- `key` argument will be always a string
- `value` argument will be any type
- If a key exists in the dictionary, the value will be replaced
- If a key doesn't exist in the dictionary, it will be created
- You are not allowed to import any module

(7)

```
guillaume@ubuntu:~/0x04$ cat 7-main.py
#!/usr/bin/python3
update_dictionary = __import__('7-update_dictionary').update_dictionary
print_sorted_dictionary = __import__('6-print_sorted_dictionary').print_sorted_di
ctionary

a_dictionary = { 'language': "C", 'number': 89, 'track': "Low level" }
new_dict = update_dictionary(a_dictionary, 'language', "Python")
print_sorted_dictionary(new_dict)
print("--")
print_sorted_dictionary(a_dictionary)

print("--")
print("--")

new_dict = update_dictionary(a_dictionary, 'city', "San Francisco")
print_sorted_dictionary(new_dict)
print("--")
print_sorted_dictionary(a_dictionary)

guillaume@ubuntu:~/0x04$ ./7-main.py
language: Python
number: 89
track: Low level
--
language: Python
number: 89
track: Low level
--
--
city: San Francisco
language: Python
number: 89
track: Low level
--
city: San Francisco
language: Python
number: 89
track: Low level
guillaume@ubuntu:~/0x04$
```

Repo:

- GitHub repository: `alx-higher_level_programming`
- Directory: `0x04-python-more_data_structures`
- File: `7-update_dictionary.py`

Done? | Check your code | Ask for a new correction | Get a sandbox | QA Review

8. Simple delete by key

mandatory

Score: 0.0% (*Checks completed: 0.0%*)

(/)

Write a function that deletes a key in a dictionary.

- Prototype: `def simple_delete(a_dictionary, key=""):`
- `key` argument will be always a string
- If a key doesn't exist, the dictionary won't change
- You are not allowed to import any module

```
guillaume@ubuntu:~/0x04$ cat 8-main.py
#!/usr/bin/python3
simple_delete = __import__('8-simple_delete').simple_delete
print_sorted_dictionary = \
    __import__('6-print_sorted_dictionary').print_sorted_dictionary

a_dictionary = { 'language': "C", 'Number': 89, 'track': "Low", 'ids': [1, 2, 3]
}
new_dict = simple_delete(a_dictionary, 'track')
print_sorted_dictionary(a_dictionary)
print("--")
print_sorted_dictionary(new_dict)

print("--")
print("--")
new_dict = simple_delete(a_dictionary, 'c_is_fun')
print_sorted_dictionary(a_dictionary)
print("--")
print_sorted_dictionary(new_dict)

guillaume@ubuntu:~/0x04$ ./8-main.py
Number: 89
ids: [1, 2, 3]
language: C
--
Number: 89
ids: [1, 2, 3]
language: C
--
--
Number: 89
ids: [1, 2, 3]
language: C
--
Number: 89
ids: [1, 2, 3]
language: C
guillaume@ubuntu:~/0x04$
```

Repo:

- GitHub repository: `alx-higher_level_programming`
- Directory: `0x04-python-more_data_structures`
- File: `8-simple_delete.py`

Done? | Check your code | Ask for a new correction | Get a sandbox | QA Review

9. Multiply by 2

mandatory

Score: 0.0% (*Checks completed: 0.0%*)

Write a function that returns a new dictionary with all values multiplied by 2

- Prototype: `def multiply_by_2(a_dictionary):`
- You can assume that all values are only integers
- Returns a new dictionary
- You are not allowed to import any module

```
guillaume@ubuntu:~/0x04$ cat 9-main.py
#!/usr/bin/python3
multiply_by_2 = __import__('9-multiply_by_2').multiply_by_2
print_sorted_dictionary = \
    __import__('6-print_sorted_dictionary').print_sorted_dictionary

a_dictionary = {'John': 12, 'Alex': 8, 'Bob': 14, 'Mike': 14, 'Molly': 16}
new_dict = multiply_by_2(a_dictionary)
print_sorted_dictionary(a_dictionary)
print("--")
print_sorted_dictionary(new_dict)

guillaume@ubuntu:~/0x04$ ./9-main.py
Alex: 8
Bob: 14
John: 12
Mike: 14
Molly: 16
--
Alex: 16
Bob: 28
John: 24
Mike: 28
Molly: 32
guillaume@ubuntu:~/0x04$
```

Repo:

- GitHub repository: `alx-higher_level_programming`
- Directory: `0x04-python-more_data_structures`
- File: `9-multiply_by_2.py`

Done? | Check your code | Ask for a new correction | Get a sandbox | QA Review

10. Best score (7)

mandatory

Score: 0.0% (*Checks completed: 0.0%*)

Write a function that returns a key with the biggest integer value.

- Prototype: `def best_score(a_dictionary):`
- You can assume that all values are only integers
- If no score found, return `None`
- You can assume all students have a different score
- You are not allowed to import any module

```
guillaume@ubuntu:~/0x04$ cat 10-main.py
#!/usr/bin/python3
best_score = __import__('10-best_score').best_score

a_dictionary = {'John': 12, 'Bob': 14, 'Mike': 14, 'Molly': 16, 'Adam': 10}
best_key = best_score(a_dictionary)
print("Best score: {}".format(best_key))

best_key = best_score(None)
print("Best score: {}".format(best_key))

guillaume@ubuntu:~/0x04$ ./10-main.py
Best score: Molly
Best score: None
guillaume@ubuntu:~/0x04$
```

Repo:

- GitHub repository: `alx-higher_level_programming`
- Directory: `0x04-python-more_data_structures`
- File: `10-best_score.py`

Done? | Check your code | Ask for a new correction | Get a sandbox | QA Review

11. Multiply by using map

mandatory

Score: 0.0% (*Checks completed: 0.0%*)

Write a function that returns a list with all values multiplied by a number without using any loops.

- Prototype: `def multiply_list_map(my_list=[], number=0):`
- Returns a new list:
  - Same length as `my_list`
  - Each value should be multiplied by `number`
- Initial list should not be modified
- You are not allowed to import any module
- You have to use `map`

- Your file should be max 3 lines

(/)

```
guillaume@ubuntu:~/0x04$ cat 11-main.py
#!/usr/bin/python3
multiply_list_map = __import__('11-multiply_list_map').multiply_list_map

my_list = [1, 2, 3, 4, 6]
new_list = multiply_list_map(my_list, 4)
print(new_list)
print(my_list)

guillaume@ubuntu:~/0x04$ ./11-main.py
[4, 8, 12, 16, 24]
[1, 2, 3, 4, 6]
guillaume@ubuntu:~/0x04$
```

Repo:

- GitHub repository: `alx-higher_level_programming`
- Directory: `0x04-python-more_data_structures`
- File: `11-multiply_list_map.py`

Done? | Check your code | Ask for a new correction | Get a sandbox | QA Review

12. Roman to Integer

mandatory

Score: 0.0% (*Checks completed: 0.0%*)

Technical interview preparation:

- You are not allowed to google anything
- Whiteboard first

Create a function `def roman_to_int(roman_string):` that converts a Roman numeral (/rltoken/oSuwqUrL0BL_hi4VqVvs_g) to an integer.

- You can assume the number will be between 1 to 3999.
- `def roman_to_int(roman_string)` must return an integer
- If the `roman_string` is not a string or `None` , return 0

(7)

```
guillaume@ubuntu:~/0x04$ cat 12-main.py
#!/usr/bin/python3
""" Roman to Integer test file
"""
roman_to_int = __import__('12-roman_to_int').roman_to_int

roman_number = "X"
print("{} = {}".format(roman_number, roman_to_int(roman_number)))

roman_number = "VII"
print("{} = {}".format(roman_number, roman_to_int(roman_number)))

roman_number = "IX"
print("{} = {}".format(roman_number, roman_to_int(roman_number)))

roman_number = "LXXXVII"
print("{} = {}".format(roman_number, roman_to_int(roman_number)))

roman_number = "DCCVII"
print("{} = {}".format(roman_number, roman_to_int(roman_number)))

guillaume@ubuntu:~/0x04$ ./12-main.py
X = 10
VII = 7
IX = 9
LXXXVII = 87
DCCVII = 707
guillaume@ubuntu:~/0x04$
```

Repo:

- GitHub repository: `alx-higher_level_programming`
- Directory: `0x04-python-more_data_structures`
- File: `12-roman_to_int.py`

Done? | Check your code | Ask for a new correction | Get a sandbox | QA Review

Done with the mandatory tasks? Unlock 4 advanced tasks now!