


Python Checkpoint

Due 22 Jun by 12:30 **Points** 1

Available 22 Jun at 10:28 - 22 Jun at 13:00 about 3 hours

Checkpoint Python Module

(download of questions: [Python_Checkpoint-1.pdf](https://canvas.academy.se/courses/343/files/75189/download?download_frd=1) 
(https://canvas.academy.se/courses/343/files/75189/download?download_frd=1)
[Minimise file preview](#)
)

Create a new GitHub repository called “checkpoint_python_NAME” where NAME is your name and clone it to your computer. All answers should be answered here and pushed to this repository when done.

When you're done with the checkpoint, provide the link to the repo with your answers as your “response”. You can also chose to make the repository private and invite my user (<https://github.com/hahnjas> [_https://github.com/hahnjas_](https://github.com/hahnjas)) .

Question 1a (Level 1):

We are working at AwesomeBank3000, where the manager has an idea for fraud detection. He thinks everyone with a salary over 555000 who pays less than 30% tax-rate, is worth investigating (tax-rate= taxes/salary). The customers database is huge, so you only take a small sample of the data into python and try to see if you can make the logic for a program that prints the name of those who fit this category.

Create a new file called “question_1.py”.

At the top of the file set the following lists:

```
“  
  
customers = ['James', 'John', 'Robert', 'Mary', 'Patricia', 'Jennifer']  
  
salary = [155000, 755000, 455000, 1255000, 635000, 575000]  
  
taxes = [55800, 317100, 182000, 451800, 171450, 71400]  
“
```

the first customer ('James') has the first salary (155000) with the first taxes (55800), the second person ('John') has the second salary and taxes values etc...

add logic to the file such that your script prints the name of anyone in the fraud category as defined by you manager.

HINT: Remember the lists are ordered and have the same length.

Question 1b (Level 2):

Adapt your previous reply to instead load customers, salary and taxes from file(s). Write a small comment in your code about how you decided to set up the file(s). You may want to use dictionaries or classes for this.

Question 2a (Level 1):

Create a new file called "question_2.py".

In the file, write a function called

"to_camel_case(text)" that takes "text" as input/parameter and returns the camel case version of that text. For camel case every word starts with a capital letter, all other letters are lower case and all spaces are removed.

Examples of what your "to_camel_case" function should return for different inputs:

to_camel_case('hello world') -> 'HelloWorld'

to_camel_case('My stRriNg') -> 'MyString'

to_camel_case('tHis Is A tesT string') -> 'ThisIsATestString'

Question 2b (Level 2):

Make a function that takes a text in camel case format input/parameter and returns snake case

The function should raise a value error if the input text has spaces saying

"wrong format dude! Try again". Also place this function in the "question_2.py" file

Question 3 (Level 1):

The company SuperSale3000 wants to rank its sales team at the end of every week. Each call gives the seller 10 points, each meeting 30 points and each sale 100 points. There is also a 100-point bonus for each, if you make more than 150 calls, or 20 meetings or 5 sales.

Create a new file called "question_3.py".

They already have the data for each employee saved in a dictionary.

At the top of the file set the following example dictionary for Jordan Belfort:

```
jordan_belfort = {'calls': 178, 'meetings':17, 'sales':6}
```

Create a function using the above logic to calculate the total score from the dictionary and add it to the same dictionary using the key 'score'.

Call the function with your dictionary as a parameter and print the result.

Create at least one more example dictionary for a team member of your choice, also call your scoring function for that team member and print the result.

Checkpoint Python Module

Create a new GitHub repository called "checkpoint_python_NAME" where NAME is your name and clone it to your computer. All answers should be answered here and pushed to this repository when done.

When you're done with the checkpoint, provide the link to the repo with your answers as your "response". You can also chose to make the repository private and invite my user (<https://github.com/hahnjas>) .

Question 1a (Level 1):

We are working at AwesomeBank3000, where the manager has an idea for fraud detection. He thinks everyone with a salary over 555000 who pays less than 30% tax-rate, is worth investigating (tax-rate= taxes/salary). The customers database is huge, so you only take a small sample of the data into python and try to see if you can make the logic for a program that prints the name of those who fit this category.

Create a new file called "question_1.py".

At the top of the file set the following lists:

```
customers = ['James', 'John', 'Robert', 'Mary', 'Patricia', 'Jennifer']
salary = [155000, 755000, 455000, 1255000, 635000, 575000]
taxes = [55800, 317100, 182000, 451800, 171450, 71400]
```

the first customer ('James') has the first salary (155000) with the first taxes (55800), the second person ('John') has the second salary and taxes values etc...

add logic to the file such that your script prints the name of anyone in the fraud category as defined by you manager.

HINT: Remember the lists are ordered and have the same length.

Question 1b (Level 2):

Adapt your previous reply to instead load customers, salary and taxes from file(s). Write a small comment in your code about how you decided to set up the file(s). You may want to use dictionaries or classes for this.

Question 2a (Level 1):

Create a new file called "question_2.py".

In the file, write a function called

"to_camel_case(text)" that takes "text" as input/parameter and returns the camel case version of that text. For camel case every word starts with a capital letter, all other letters are lower case and all spaces are removed.

Examples of what your "to_camel_case" function should return for different inputs:

to_camel_case('hello world') -> 'HelloWorld'

to_camel_case('My stRriNg') -> 'MyString'

to_camel_case('tHis Is A tesT string') -> 'ThisIsATestString'