Conda, Typechecking and Pytest

Exercises on Typechecking

Consider the file `exercise.py` with the following content:

```
SURNAME_INDEX = 0
FIRSTNAME_INDEX = 1
ADULT_AGE = 18


def create_person(surname, firstname, age):
    return surname, firstname, age


def get_names_of_adult_persons(persons):
    return [f'{person[SURNAME_INDEX]} {person[FIRSTNAME_INDEX]}'
            for person in persons if person[2] >= ADULT_AGE]


if __name__ == '__main__':
    persons_list = []

    mike = create_person('Davis', 'Mike', '25')
    john = create_person('Roberts', 'John', 16)
    lee = create_person('Willams', 46, 'Lee')

    persons_list.append(mike)
    persons_list.append(john)
    persons_list.append('lee')

    print(get_names_of_adult_persons(persons_list))
```

There are two functions:

- `create_person` takes three arguments, the first two of them are strings, while the last is an integer. This function returns a tuple of the three arguments passed to it. The returned tuple represents the data `(surname, firstname, age)` of a person.
- `get_names_of_adult_persons` takes a list of tuples that represents persons (those that are created by the first function.) The function returns a list of the names of the persons who are adults (their age is at least 18).

You should update the code in this file by doing the following:

- add type hints (for more info about mypy check this link).
- fix the errors.

Exercises on Pytest

Write a pytest file with the name `test_exercises.py` to test the functionality of the functions in `exercises.py`. Both files should be **in the same directory**. In `test_exercises.py`, you should write two test functions:

- `test_create_person` with at least two test cases.
- `test_get_names_of_adult_persons` with at least two test cases.

As an additional exercise, you should do the following:

- add a new function `count_adult_persons` that returns the count of adults given a list of tuples that represents persons.
- add a test for the newly created function with at least two test cases.

To run your tests:

1. In VSCode, click on the `Testing` tab on the left.
2. Select `Configure Python Tests` and click on the `pytest` option.
3. Select the directory where `exercises.py` and `test_exercises.py` are in.
4. You should be able to run your tests in VSCode by pressing the *play* button on the left.

Note that, alternatively, in the terminal you can run `$ pytest test_exercises.py`.

Clean Coding Style/Conventions Resources

In this course, you have to write well-documented clean code. You also have to follow python conventions. You can consult the following links to know more about python conventions and clean code principles:

- Style Guide for Python Code
- Documentation Strings Conventions
- Google Python Style Guide
- Blogs on Clean Code and Code Quality: link 1, link 2, link 3, and link 4

Also, try to write your code in a more pythonic way (i.e., write idiomatic python). You can check this link.