Initial Set – Up Before Giving The Assignment

Step 1: Create a folder by the name `assignment_evaluator`

Step 2 (Optional): In terminal go to the directory `assignment_evaluator` and create a virtual python environment by the name `evaluator_venv`:

- `python -m venv evaluator_venv`

To activate the virtual environment run:

- `source evaluator_venv/bin/activate`

To deactivate the virtual environment run:

- `deactivate`

Note: We only need this virtual environment to separate the dependencies from global python set up on our system. If not necessary, we can skip this.

Initial Set – Up Before Giving The Assignment

Conti. . .

Step 3: Write the master program which contains the correct solution to the problem we are giving as assignment:

- `master_program.py`

Step 4: Write a program to populate the `test_cases.json` file which will import `master_program.py`:

- `correct_program_to_populate_the_test_cases.py`

Note: After this step we will have `test_cases.json` file ready for us.

Step 5: Make a `configs.py` file with all the configuration details required.

Initial Set – Up Before Giving The Assignment

Conti. . .

Step 6: Create a folder with the name `submissions`. This will contain all the submissions of the students.

Step 7: Create a folder with the name `REQUIREMENTS`. This will contain all the submitted `requirements.txt` files of the students.

Step 8: Write a program `prepare_combined_requirements_file.py` to gather all the unique requirements from each of the files inside `REQUIREMENTS` folder and prepare a `requirements.txt` file using which we will install all the dependencies by running the following command in terminal:

- `pip install -r requirements.txt`

Initial Set – Up Before Giving The Assignment

Conti. . .

Step 9: Write the `evaluator.py` program which evaluates all the .py format files in the submissions folder

Note: Run the `evaluator.py` as the main function with all the code inside `__main__` block. Because I have used multiprocessing to speed up the evaluation by exploiting parallel computing. For multiprocessing, each process imports scripts every time it is created. So if we are not running it from `__main__` block it will lead to infinite loop for sure.

Folder Structure After The Initial Set-Up

assignment_evaluator/

- submissions/ (Empty folder)
- REQUIREMENTS/ (Empty folder)
- master_program.py
- prepare_combined_requirements_file.py
- correct_program_to_populate_test_cases.py (Uses master_program.py)
- configs.py
- test_cases.json
- evaluator.py

Instructions For the Students

1. Write the entire code in a single `.py` file and name it as `ID_Firstname_Lastname.py`
2. The main function should have the same name as we specified. (Suppose the assignment is adding two integers and we ask them to use **”add”** as the main function that gives the ultimate result. Then they should use the name **”add”** for their main function.)
3. If importing any modules that are not part of the Python standard library, then submit a `requirements.txt` file with the name `employeeID_Firstname_Lastname.txt`
4. Don’t call the function.
5. For submission, mail both the files to `evaluator@company.com`