Lab #3 - 학점 계산기 (exam_grader)

Introduction

이번 과제는 조금 짧습니다. 그러나 간단하지 않습니다. 처음으로 수강자가 직접 함수를 작성합니다. 나중에 보면 매우 쉬운 함수이지만 지금은 어렵게 느껴질 것입니다. 이미 포기하기는 늦었으니 해봅시다.

backend.ai 설치

숙제를 제출하기 앞서, 레블업의 backend.ai를 여러분의 파이썬에 설치하셔야 합니다. 설치하는 과정은 매우 쉽습니다. 아래처럼 터미널 또는 cmd 창에서 입력을 하시면 됩니다.

```
pip install backend.ai-client
```

숙제 파일(lab_3.zip) 다운로드

먼저 해야 할 일은 숙제 파일을 다운로드 받는 것 입니다. 이미 해보았기 때문에 어렵지 않을 것입니다. Chrome 또는 익스플로러와 같은 웹 브라우저 주소창에 아래 주소를 입력합니다.

https://github.com/TeamLab/introduction_to_python_TEAMLAB_MOOC/blob/master/lab_assignment/lab_3/lab_3.zip

다운로드를 위해 `View Raw` 또는 `Download` 버튼을 클릭합니다. 또는 아래 다운로드 링크를 클릭하면 자동으로 다운로드가 됩니다. Lab 3 - 다운로드 다운로드 된 lab_3.zip 파일을 작업 폴더로 이동한 후 압축해제 후 작업하길 바랍니다. 압축해제 하면 폴더가 `linux_mac`과 `windows`로 나눠져 있습니다. 자신의 OS에 맞는 폴더로 이동해서 코드를 수정해 주시기 바랍니다.

폴더 안에는 `exam_grader.py` 파일이 있을 것 입니다. 받자마자 실험 차원에서 코드를 한번 실행해봅니다. 코드를 실행할 때에는 cmd 또는 터미널 창에서 `python exam_grader.py`를 입력하면 됩니다. 이제 다들 코드를 어떻게 실행하는 지는 알 것입니다. 실행하면 아마 아래와 같은 에러 메세지가 뜰 것입니다.

```
Start of Exam Grader Program
============================
Traceback (most recent call last):
  File "exam_grader.py", line 67, in <module>
    main()
  File "exam_grader.py", line 58, in main
    total_score = sum_of_scores(number_of_subjects)
  File "exam_grader.py", line 29, in sum_of_scores
    for i in range(number_of_subjects):
TypeError: 'NoneType' object cannot be interpreted as an integer
```

기존의 lab과 달리 이번 코드는 각 함수들을 개별로 작동하는 코드가 아닌 함수들이 순차적으로 작동하여 최종적인 목표를 달성하도록 설계되어 있습니다. 코드의 구조를 살펴봅시다.

exam_grader.py 코드 구조

이번 lab을 수행하기 위해서 첫 번째로 `main` 함수부터 들여다 봅시다. `main` 함수는 아래와 같이 구성되어 있습니다. 혹시몰라 또 적지만 밑에 코드는 `atom 에디터`로 확인이 가능하고, 해당 `cmd` 또는 `terminal` 환경에서 해당 디렉토리로 이동 후, `atom exam_grader.py` 명령으로 파일을 열어볼 수 있습니다.

```
def main():
    print("Start of Exam Grader Program")
    print("============================")

    number_of_subjects = get_number_of_subjects()
```

```
    total_score = sum_of_scores(number_of_subjects)
    average_score = get_average_score(
        total_score=total_score, number_of_subjects=number_of_subjects)
    print_exam_grader(average_score)

    print("==========================")
    print("End of Exame Grader Program")
```

처음 두 줄과 마지막 두 줄은 프로그램의 시작과 끝을 알리는 `print` 문으로 실제 프로그램 실행에 영향을 주지 않습니다. `number_of_subjects = get_number_of_subjects()` 코드에는 우리가 첫 번째로 수정해야 할 함수가 나옵니다. 본 코드는 `get_number_of_subjects` 의 함수에서 나온 결과 값을 `number_of_subjects` 변수에 저장하라는 의미입니다. `get_number_of_subjects` 는 사용자에게 입력을 받아 `총 과목의 수` 를 계산해줍니다.

다음 줄인 `total_score = sum_of_scores(number_of_subjects)` 는 실제 이번 숙제에 직접적인 영향을 주지 않는 `helper 함수` 입니다. 전체적인 프로그램의 실행을 도와주는 함수로 과목의 수에 따라 각 과목의 점수를 입력받을 수 있도록 설계되어 있습니다. 숙제 제출이 끝난 후 살펴볼 것을 권장합니다.

다음 줄인 `average_score = get_average_score(total_score=total_score, number_of_subjects=number_of_subjects)` 코드가 이번 lab에서 가장 중요한 코드입니다. 본 코드는 `get_average_score` 라는 함수에 `total_score` 와 `number_of_subjects` 라는 변수를 입력하여 성적의 평균 값을 `average_score` 에 할당합니다. `number_of_subjects` 는 첫 번째 코드에서, `total_score` 는 두 번째 코드에서 각각 값이 결정됩니다. 현재 다운로드 받은 `lab code` 에는 `get_average_score` 가 존재하지 않습니다. 직접 작성해야만 합니다.

마지막으로 `print_exam_grader(average_score)` 는 성적의 평균 값을 바탕으로 최종 평균과 학점을 화면에 출력해주는 함수를 사용하는 코드입니다. 역시 `helper 함수` 로 수강자가 수정할 필요가 없습니다. 그러나 다음 시간에 배울 `if` 문에 대해서 적혀있으므로 숙제 제출후 꼭 살펴볼 것을 권장합니다.

get_number_of_subjects 함수 수정하기

첫 번째 수정내용은 `get_number_of_subjects` 함수입니다. 본 함수의 코드는 아래와 같습니다.

```
# 총 과목 수를 Console를 통해 받기
def get_number_of_subjects():

    # """
    # Input:
    #   - None
    # Output:
    #   - number_of_subjects: Integer Type의 총 과목 수
    # Examples(python shell):
    #   >>> import exam_grader as eg
    #   >>> eg.get_number_of_subjects()
    #   과목수를 입력하세요: 10
    #   10
    # """
    #
    # ===Modify codes below=================
    number_of_subjects = None
    # ======================================
    return number_of_subjects
```

복잡해 보일 수도 있으나, 함수 자체는 간단합니다. 입력 받는 값은 없고 함수를 실행시키면 사용자 console 창에 `과목수를 입력하세요:` 라는 질문이 나오면서 사용자의 입력을 기다립니다. 사용자가 **정수** 를 입력하면 그 값을 integer type으로 변환하여 반환해줍니다. 간단하게 1) 입력 받고, 2) 정수로 변환해주기만 하면 되는 함수이기 때문에 쉽게 할 수 있을 것입니다.

get_average_score 함수 생성하기

이번엔 전혀 주어진 template 코드 없이 직접 함수를 작성해봅시다. 함수에 대한 정보는 아래와 같습니다.

내용	구성
함수명	get_average_score
input 변수(argument)	1. total_score : Integer Type의 성적 총합

	2. number_of_subjects : Integer Type의 과목 갯수
output 값(return)	Float Type의 total_score을 number_of_subjects로 나눈 값

이것만 보고 어떻게 함수를 만들지 난감하겠지만, 이 정도 정보면 모든 정보를 다 준 것입니다. 실제 개발을 할 때는 함수명이나 변수명은 개발자가 임의로 정할 수 있습니다. 그러나 본 숙제에서는 정해진 규칙대로 작성하지 않으면 숙제 검사를 해주지 않습니다. ~~컴퓨터는 거짓말을 하지는 않지만, 그렇다고 융통성이 있지도 *않습니다*~~.

테스트 및 제출

두 함수를 모두 작성했다면 실행을 해봅시다. `console 환경`에서 `python exam_grader.py` 명령어로 실행을 하면 됩니다. 정확히 작성되었다면 아래와 같이 결과를 볼 수 있을 것입니다. 아래 결과 중 셋째 줄인 `과목수를 입력하세요:`는 출력되고 뒤에 `3`은 실습자가 직접 입력하는 것이며, 그 아랫줄에 `85`, `95`, `100` 모두 사용자가 직접 입력하는 항목입니다.

```
Start of Exam Grader Program
============================
과목 수를 입력하세요: 3
1번째 과목의 점수를 입력하세요 : 85
2번째 과목의 점수를 입력하세요 : 95
3번째 과목의 점수를 입력하세요 : 100
평균 점수:  93.33333333333333
학     점:  A
============================
End of Exame Grader Program
```

이제 숙제를 제출합시다.

숙제 template 파일 제출하기 (윈도우의 경우)

1. `windows` + `r`를 누르고 cmd 입력 후 확인을 클릭합니다.
2. 작업을 수행한 폴더로 이동 합니다.
3. 밑에 명령어를 cmd창에 입력합니다.

```
submit.bat [YOUR_HASH_KEY]
```

숙제 template 파일 제출하기 (Mac or Linux)

1. 터미널을 구동합니다.
2. 작업을 수행한 디렉토리로로 이동 합니다.
3. 밑에 bash창을 입력합니다.

```
bash ./submit.sh [YOUR_HASH_KEY]
```

완벽하지 못한 상태에서 숙제 제출을 수행하면 아래와 같은 에러를 볼 수도 있습니다.

```
--------------------------- | ----------- | --------------------
       Function Name        |    Passed?  |             Feedback
--------------------------- | ----------- | --------------------
test_get_number_of_subjects |       PASS  |             Good Job
test_get_average_score      |    Not Yet  |   Check Your Grammar
--------------------------- | ----------- | --------------------
```

이런 경우는 `get_average_score` 함수를 이름에 안맞게 작성했을 경우 주로 나타납니다. 물론 아닌 경우도 있습니다. 다시 위의 내용을 점검하여 오탈자가 없는지 확인한 후 제출해봅시다. 제대로 작성할 경우 아래 메세지를 볼 수 있습니다.

```
----------------------- | ------------ | --------------------
       Function Name    |    Passed?   |             Feedback
----------------------- | ------------ | --------------------
get_number_of_subjects  |       PASS   |             Good Job
   get_average_score    |       PASS   |             Good Job
----------------------- | ------------ | --------------------
```

Next Work

느끼시겠지만, 더 이상의 친절함은 없습니다. 점점 스스로 혼자 헤쳐나가야 합니다. 많은 에러 메세지와 사투를 벌여야 하고, 왠지 모르지만 안되는 이유를 찾아야 합니다. 이제 구글에 익숙해지길 바랍니다. 구글 검색이야말로 최고의 프로그래머가 되는 지름길입니다.

> **Human knowledge belongs to the world** - from movie 'Password' -