1주차 과제

성적 관리 프로그램

(문제)

파일로부터 데이터를 읽어서 성적 목록을 만들어 관리하는 성적 관리 프로그램을 작성한다.

(주의사항)

- 이 프로그램은 사용자로부터 **7개의 명령어**(show, search, changescore, searchgrade, add, remove, quit)를 입력 받아 각 기능을 수행 하게 된다. 최소한 각 명령어 별로 함수를 정의하여 사용한다. 명령어 외에 필요한 함수는 추가로 정의하여 사용할 수 있다.

- 소스코드 제출파일이름은 "**project1.py**"로 한다.

- 보고서 파일은 **project1.doc(x)** 로 한다. 보고서 작성에 대한 설명은 마지막 페이지에 있으니 아래의 문제를 풀기 전, 먼저 확인하도록 한다.

- 기본적으로 문제에서 요구한 사항대로 구현을 하되, 실제로 프로그램을 작성하다 보면 결정해야 할 세부 사항이 많아진다. 명시되어 있지 않은 세부사항에 대해서 처리한 방법, 이유 등을 보고서에 기록하도록 한다.

(설명)

➢ 프로그램 실행은 다음과 같이 한다. (실행예시에 밑줄로 표시된 문자는 사용자 입력에 해당)

- 리눅스의 경우

```
$ python project1.py students.txt
```

- 윈도우의 경우

```
C:₩>python project1.py students.txt
```

- 파일명을 입력하지 않을 경우, default로 "students.txt"로부터 데이터를 읽는다.
- 파일명이 입력될 경우, 입력된 파일로부터 데이터를 읽는다.
- 파일명에는 공백이 없다고 가정한다. (즉, 공백이 있는 파일명의 입력에 대해서는 고려하지 않는다.

➢ 프로그램 실행 시 텍스트 파일로부터 학생들의 성적 목록 작성을 위한 데이터를 읽으며, 텍스트 파일의 내용 및 구성은 아래와 같다.

20180001	Hong Gildong	84	73
20180002	Lee Jieun	92	89
20180007	Kim Cheolsu	57	62
20180009	Lee Yeonghee	81	84
20180011	Ha Donghun	58	68

- 각 줄은 각 학생의 학번, 이름, 중간고사 점수, 기말고사 점수로 구성되어 있으며 각 항목 사이는 **tab(₩t)**으로 구분된다.
- 학생과 학생 사이는 **줄 바꿈 문자(₩n)**으로 구분된다.
  Ex. **[Student number][₩t][Name][₩t][Midterm][₩t][Final][₩n]**
- 프로그램을 실행시키면 텍스트 파일로부터 데이터를 읽어 목록을 리스트(list) 자료형 또는 딕셔너리(dict) 자료형을 사용하여 저장하고, 전체 목록을 평균 점수를 기준으로 내림차순으로 정렬하여 아래의 예제처럼 출력한다. 동일한 평균 점수를 가진 학생들이 있는 경우 순서는 상관없다.

```
   Student            Name    Midterm    Final    Average    Grade
-------------------------------------------------------------------
  20180002        Lee Jieun       92       89       90.5        A
  20180009     Lee Yeonghee       81       84       82.5        B
  20180001     Hong Gildong       84       73       78.5        C
  20180011       Ha Donghun       58       68       63.0        D
  20180007      Kim Cheolsu       57       62       59.5        F

#
```

(힌트) 학생목록 저장과 정렬 관련 힌트는 7쪽에 있음.

- 평균(Average)항목은 중간고사 점수와 기말고사 점수의 평균을 계산하여 저장한다.
- 학점(Grade)항목의 기준은 아래와 같다.

```
A: 평균이 90 점 이상
B: 평균이 80 점 이상, 90 점 미만
C: 평균이 70 점 이상, 80 점 미만
D: 평균이 60 점 이상, 70 점 미만
F: 평균이 60 점 미만
```

- 위와 같이 학생들의 성적 목록이 출력 된 후에는 명령어 입력을 대기하는 #표시가 뜨며, 이 상태에서 사용자는 명령어를 입력할 수 있다.

- 사용자는 **7개의 명령어**(show, search, changescore, searchgrade, add, remove, quit)를 사용할 수 있으며, 명령어를 입력하였을 때만 기능이 실행된다. 이 명령어는 사용자가 명령어 입력 시, 대소문자를 구분하지 않고 동일한 명령어의 기능을 수행하도록 작성한다. 예를 들면, show, SHOW, Show, shoW 는 동일한 동작을 수행한다.

- 7개의 명령어 이외의 잘못된 명령어 입력 시, 에러 메시지 없이 아래와 같이 다시 명령어를 입력 받을 준비를 한다.

```
# find
#
```

(기능)

- 성적 관리 프로그램은 아래와 같은 기능을 가진다.
- 명시된 7가지 명령어 외의 명령어가 입력될 경우 무시하고 다시 명령어 입력을 대기한다.

1. show (전체 학생 정보 출력)

- ***show*** 입력 시, 저장되어 있는 전체 목록을 아래와 같이 평균 점수를 기준으로 내림차순으로 출력한다. 평균 점수는 소수점 이하 첫째 자리까지만 표시한다.

```
# show
   Student            Name    Midterm    Final    Average    Grade
-------------------------------------------------------------------
  20180002        Lee Jieun       92       89       90.5        A
  20180009     Lee Yeonghee       81       84       82.5        B
  20180001     Hong Gildong       84       73       78.5        C
  20180011       Ha Donghun       58       68       63.0        D
  20180007      Kim Cheolsu       57       62       59.5        F


#
```

2. search (특정 학생 검색)

- ***search*** 입력 시, 아래와 같이 검색하고자 하는 학생의 학번을 요구해 입력 받아 학번, 이름, 중간고사 점수, 기말고사 점수, 평균, 학점을 출력한다.
- 예외처리:
  - ✓ 찾고자 하는 학생이 목록에 없는 경우에는 "NO SUCH PERSON." 이라는 에러 메시지를 출력

```
# search
Student ID: 20180050
NO SUCH PERSON.

# search
Student ID: 20180002
   Student            Name    Midterm    Final    Average    Grade
-------------------------------------------------------------------
  20180002        Lee Jieun       92       89       90.5        A


#
```

3. changescore (점수 수정)

- 목록에 저장된 학생 중 1명의 중간고사(*mid*) 혹은 기말고사(*final*)의 점수를 수정한다.
- ***changescore*** 입력 시, 수정하고자 하는 학생의 학번, 수정하고자 하는 점수가 중간고사인지 기말고사인지와 수정하고자 하는 점수를 순서대로 입력 받아 해당 학생의 점수를 수정한다.
- 점수가 바뀜에 따라 Grade도 다시 계산하여 수정한다.
- 예외처리:
  - ✓ 학번이 목록에 없는 경우에는 "NO SUCH PERSON."이라는 에러 메시지를 출력
  - ✓ "mid" 또는 "final" 외의 값이 입력된 경우에는 실행되지 않음
  - ✓ 점수에 0~100 외의 값이 입력된 경우에는 실행되지 않음

```
# changescore
Student ID: 20180050                    ➔ 학번이 없는 경우
NO SUCH PERSON.

# changescore
Student ID: 20180007
Mid/Final? miid                         ➔ mid/final 외의 값이 입력된 경우

# changescore
Student ID: 20180007
Mid/Final? mid
Input new score: 147                    ➔ 0~100 외의 값이 입력된 경우

# changescore
Student ID: 20180007
Mid/Final? mid
Input new score: 75
   Student            Name    Midterm    Final    Average    Grade
-----------------------------------------------------------------
  20180007     Kim Cheolsu       57        62       59.5       F
Score changed.
  20180007     Kim Cheolsu       75        62       68.5       D


# show
   Student            Name    Midterm    Final    Average    Grade
-----------------------------------------------------------------
  20180002       Lee Jieun       92        89       90.5       A
  20180009    Lee Yeonghee       81        84       82.5       B
  20180001    Hong Gildong       84        73       78.5       C
  20180007     Kim Cheolsu       75        62       68.5       D
  20180011      Ha Donghun       58        68       63.0       D


#
```

4. add (학생 추가)

- ***add*** 입력 시, 아래와 같이 학생의 학번, 이름, 중간고사 점수, 기말고사 점수를 차례로 요구해 입력 받는다. 추가되면, 메시지 "Student added."를 아래 예제와 같이 출력한다.
- Average와 Grade는 중간고사 점수와 기말고사 점수를 사용하여 계산하여 저장한다.
- 학생 추가 후 ***show*** 명령어를 사용하면 평균을 기준으로 **내림차순**으로 출력된다.
- 에러처리:
  - ✓ 목록에 있는 학생의 학번을 입력 시, 'ALREADY EXISTS.' 이라는 에러 메시지 출력

```
# add
Student ID: 20180001
ALREADY EXISTS.

# add
Student ID: 20180021
Name: Lee Hyori
Midterm Score: 93
Final Score: 95
Student added.

# add
Student ID: 20180006
Name: Lee Sangsun
Midterm Score: 77
Final Score: 66
Student added.

# show
   Student            Name    Midterm    Final    Average    Grade
-------------------------------------------------------------------
  20180021        Lee Hyori         93       95       94.0        A
  20180002        Lee Jieun         92       89       90.5        A
  20180009     Lee Yeonghee         81       84       82.5        B
  20180001     Hong Gildong         84       73       78.5        C
  20180006      Lee Sangsun         77       66       71.5        C
  20180007      Kim Cheolsu         75       62       68.5        D
  20180011       Ha Donghun         58       68       63.0        D


#
```

5. searchgrade (Grade 검색)

- ***searchgrade***입력 시, 특정 grade를 입력 받아 그 grade에 해당하는 학생을 모두 출력한다.
- 예외처리:
  - ✓ A, B, C, D, F 외의 값이 입력된 경우 실행되지 않음.
  - ✓ 해당 grade 의 학생이 없는 경우 아래와 같이 메시지 "NO RESULTS." 출력

```
# searchgrade
Grade to search: E                    ➔ A/B/C/D/F 외의 값이 입력된 경우

# searchgrade
Grade to search: F                    ➔ 해당 grade의 학생이 없는 경우
NO RESULTS.

# searchgrade
Grade to search: D
   Student            Name    Midterm    Final    Average    Grade
-----------------------------------------------------------------
  20180007      Kim Cheolsu       75       62       68.5        D
  20180011       Ha Donghun       58       68       63.0        D


#
```

6. REMOVE (특정 학생 삭제)

- ***remove*** 입력 시, 아래와 같이 삭제하고자 하는 학생의 학번을 입력 받은 후, 학생이 목록에 있는 경우 삭제한다. 삭제하면, 메시지 "Student removed."를 아래와 같이 출력한다.
- 예외처리:
  - ✓ 목록에 아무도 없을 경우 아래의 예제와 같이 "List is empty." 메시지 출력

```
# remove
List is empty.

#
```

  - ✓ 학생이 목록에 없는 경우에는 "NO SUCH PERSON."이라는 에러 메시지를 출력

```
# remove
Student ID: 20180030
NO SUCH PERSON.

# remove
Student ID: 20180011
Student removed.

# show
   Student            Name    Midterm    Final    Average    Grade
-----------------------------------------------------------------
  20180021       Lee Hyori       93       95       94.0        A
  20180002       Lee Jieun       92       89       90.5        A
  20180009    Lee Yeonghee       81       84       82.5        B
  20180001    Hong Gildong       84       73       78.5        C
  20180006     Lee Sangsun       77       66       71.5        C
  20180007     Kim Cheolsu       75       62       68.5        D


#
```

7. quit (종료)

- ***quit*** 입력 시, 프로그램을 종료한다.
- 해당 명령어를 실행할 경우, 현재까지 편집한 내용의 저장 여부를 묻고, 저장을 선택(yes 입력)할 경우 파일명을 입력 받아서 저장하도록 한다. 앞서 본 "students.txt"와 같이 내용을 구성한다.

  [Student number][₩t][Name][₩t][Midterm][₩t][Final][₩n]
- 저장할 때 목록의 순서는 평균을 기준으로 내림차순으로 한다.
- 파일 이름에는 공백이 없다고 가정한다.

```
# quit
Save data?[yes/no] yes
File name: newStudents.txt
$
```

```
# quit
Save data?[yes/no] no
$
```

(힌트) 학생목록 저장 및 정렬함수

(1) 중첩리스트(list)사용 예시

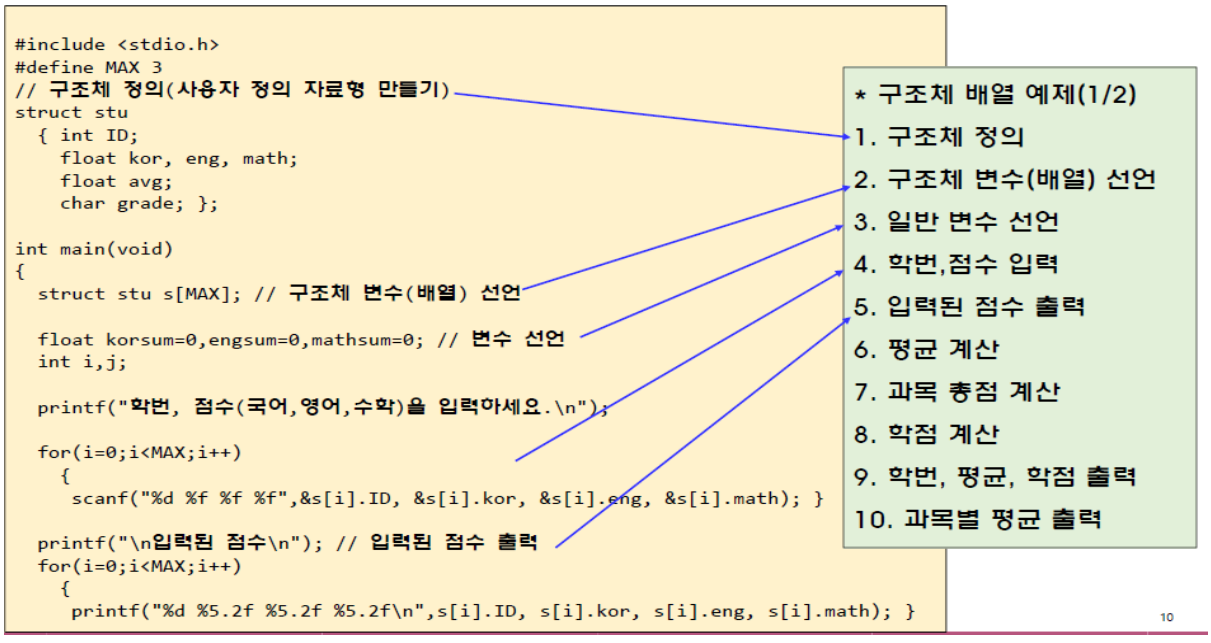
```
# 학생이름에 학번, 중간점수, 기말점수를 순서대로 넣어줍니다.
>>> gildong = [201801, 84, 73]
>>> jieun = [201802, 92, 89]
>>> cheolsu = [201803, 57, 62]
>>> donghun = [201804, 58, 68]
# 위 학생들은 전산과목을 듣는 학생들이며, 모두 stu_list 목록에 넣어서 관리합시다.
>>> stu_list = [gildong, jieun, cheolsu, donghun]
>>> print(stu_list)
[[201801, 84, 73], [201802, 92, 89], [201803, 57, 62], [201804, 58, 68]]
>>> for stu in stu_list:   # 실행해 볼 것
        print(stu)
>>> stu_list[1][2]   # 어떤 값이 나올까요? stu_list[1] 의 접근으로 [201802, 92, 89]가 선택되고, 선택된 리스트의 2번째 index이니, jieun 학생의 기말점수가 선택된 거죠?
# 원소 접근 방법을 알았으니, 기말점수를 기준으로 오름차순으로 정렬하여 출력을 해 봅시다.
>>> stu_list.sort(key=lambda e : e[2])
>>> print(stu_list)
[[201803, 57, 62], [201804, 58, 68], [201801, 84, 73], [201802, 92, 89]]
# 기말점수를 기준으로 내림차순으로 정렬하는 방법은 아시죠?
# lambda가 무엇인지 궁금하시죠? lambda 함수에 대해서 찾아보고, 정의하는 방법 등을 익히시고 보고서에 기록한 후, 사용하면 됩니다.
```

(2) 사전(dict) 사용

```
# 위의 학생목록을 딕셔너리로 구현할 수 있습니다. 학번은 중복되지 않으니 keyword로 사용하면 좋겠죠? 사전은 value로 아래와 같이 리스트를 가질 수 있습니다.
>>> stu_dict = {201801:[84,73], 201802:[92,89], 201803:[57,62], 201804:[58,68]}
# 그러면 201802 학번 학생의 중간점수에 접근하려면 어떻게 해야 할까요?
>>> stu_dict[201802]  # key 값으로 value에 접근할 수 있죠?
[92, 89]
>>> stu_dict[201802][0] # 선택된 리스트 [92, 89] 에서 index가 0인 것이 중간이죠?
92
# 마찬가지로 value 가 리스트일 때, 원소 접근하는 방법을 알았으니 중간점수를 기준으로 정렬을 해 봅시다.
>>> sorted_sl = sorted(stu_dict.items(), key = lambda a: a[1][0])
>>> print(sorted_sl)
 [(201803, [57, 62]), (201804, [58, 68]), (201801, [84, 73]), (201802, [92, 89])]
# 중간고사를 기준으로 정렬된 리스트를 얻을 수 있죠?
```

(보고서 작성)

1. 보고서는 doc(x) 문서로 작성한다. project1.doc(x)
2. 보고서의 첫 페이지에는 제목, 이름, 이메일 등을 적는다.
3. 프로그램 기능에 대한 간단한 개요를 적는다.
4. 프로그램 작성 전, 먼저 문제 해결 방법을 설계(알고리즘 설계)를 한 후, 구현을 하도록 한다. 의사코드(pseudo code) 또는 순서도를 작성한 후, 구현하도록 한다.
   참고)
   아래는 이론 강의자료 "Data Structure"에서 23번 슬라이드에 있는 내용이다.



5. 예제: 프로그램 실행 결과를 각 기능에 대해서 간단히 실행 한 후, 화면 캡쳐를 하여 포함시

키고 간단히 설명한다.

- **리눅스(Ubuntu)에서 화면캡쳐하는 방법** (일부 화면만 캡처하는 방법이 설명되어 있음)
  http://www.morenice.kr/108

6. 토론: 프로그램 개발 과정에서 생겼던 여러 가지 문제 또는 결정 사항에 대한 해결 방법 및 결정 과정에 대하여 설명한다.
7. 결론: 이번 숙제를 통하여 배우거나 깨달은 내용들을 정리한다.