청년 AI·Bigdata 아카데미 25기

파이썬 프로그래밍 프로젝트

성적 관리 프로그램

담당교수: 윤은영

소　　속: B반 2조

학　　과: 정보통계학과 / 빅데이터과

이　　름: 김한탁

이 메 일: gksxkrdl @ naver.com

명예서약(Honor code)

"나는 이 프로그래밍 과제를 다른 사람의 부적절한 도움 없이 완수하였습니다."

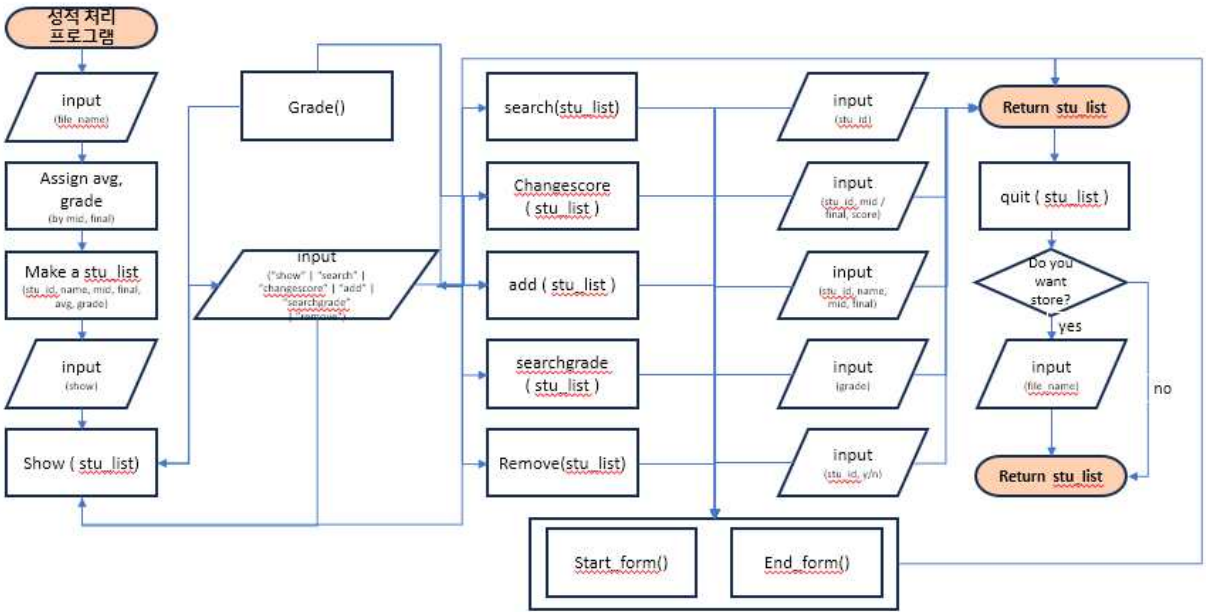
Problem: 성적 관리 프로그램

1. 문제의 개요

본 문제는 학생들의 성적을 입력 받아 처리하는 프로그램을 만드는 것으로, 문제에서 해결하고자 하는 주요한 내용들은 다음과 같다.

ⓐ 텍스트 파일의 중간고사 점수, 기말고사 점수를 이용해, 평균, 학점 계산
ⓑ 프로그램 실행 시, 전체 목록을 평균 점수를 기준으로 내림차순 정렬하여 출력
ⓒ 7개의 명령어(show, search, changescore, searchgrade, add, remove, quit)의 입력을 통해, 사용자는 기능을 사용할 수 있다.
ⓓ show 함수를 통해 저장된 목록을 평균 점수를 기준으로 내림차순 정렬 뒤, 소수 첫째 자리까지 출력
ⓔ search 함수를 통해, 특정 학번을 입력받아 이름, 중간시험 점수, 기말시험 점수, 평균, 학점 출력
ⓕ changescore 함수를 통해, 학번, 중간고사인지 기말고사인지, 수정하고자 하는 점수를 입력받아 해당 학생의 점수와 학점 다시 계산
ⓖ add 함수를 통해, 학번, 이름, 중간고사 점수, 기말고사 점수를 차례로 입력받아 새로운 학생을 추가
ⓗ searchgrade 함수를 통해, 특정 학점을 입력받아 해당 학점에 해당되는 학생을 모두 출력
ⓘ remove 함수를 통해, 특정 학번을 입력받아 해당 학생의 데이터를 제거 여부를 결정
ⓙ quit 함수를 통해, 편집된 데이터 저장 여부를 결정하고 프로그램을 종료
ⓚ 명령어를 통해 함수를 거치는 학생 데이터들은 기존 상태에서 편집되어 리스트가 업데이트되어야 하며, quit 함수를 만나기 전까지는 무한히 명령어를 통한 명령을 내릴 수 있어야 한다.

2. 알고리즘(순서도)



3. 프로그램 구조 및 설명

1) students.txt 파일 불러오기

- 프로그램 파일명을 입력받고, 불러온 파일의 학번(stu_id), 이름(name), 중간고사 점수(mid), 기말고사 점수(final)을 변수로 저장하고 mid와 final을 이용하여 평균(avg)과 학점(grade)을 구한다. 그 뒤 학생별 데이터를 리스트에 저장하고, for문을 통해 중첩리스트인 student_list에 저장한다.
- 또한 파일명을 입력받지 않으면, 기본적으로 "students.txt"로부터 데이터를 리스트에 저장된다.

2) 초기화면 출력

- 중첩리스트인 student_list를 이용하여, 평균을 기준으로 내림차순으로 정렬한 학생 데이터를 출력한다. 이 과정은 사용자 정의 함수인 show 함수를 이용한다.

3) 명령문 반복

- while / True 문을 이용하여, 사용자로부터 입력받은 명령어를 무한히 반복할 수 있도록 한다. break 문은 quit 함수에 존재하여, quit 함수를 통해 프로그램을 종료한다. 더하여, 잘못된 입력에 대해선 오류 알림 문구를 띄우고, 다시 명령어를 입력하도록 안내한다.

4) start_form 함수

- 중첩리스트를 이용해 학생 데이터를 출력할 때, 처음 출력될 수 있도록 만들어진 시작 서식을 나타내는 사용자 정의 함수

5) end_form 함수

- 중첩리스트를 이용해 학생 데이터를 출력할 때, 마지막에 출력될 수 있도록 만들어진 종료 서식을 나타내는 사용자 정의 함수

6) Grade 함수

- 평균(avg)를 토대로 학점(grade)를 산출하는 사용자 정의 함수로, 다음의 조건을 가진다.

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

7) show 함수

- 중첩리스트인 student_list를 이용하여, 평균을 기준으로 내림차순으로 정렬한 학생 데이터를 출력한다.

8) search 함수

- 학번을 입력받아, if 문을 통해 중첩리스트 내 학생 데이터에 일치하는 학번이 있는 경우 해당 학생의 리스트를 출력한다. 없는 경우에는 학생이 존재하지 않음을 알린다.

9) changescore 함수

- "changescore"를 입력함에 따라, 학번(Student), 시험 종류(Midterm/Final), 점수가 모두 잘 입력받고, 기존 학생의 데이터와 변경된 데이터를 출력하여 성공적으로 변경되었는지 확인한다. 만약 시험 종류가 잘못 입력되었다면 재확인을 요구하는 알림을 주고, 점수가 잘못 입력되었다면 점수의 정상적인 입력범위를 출력한다.

10) add 함수

- 학번(Student)을 입력받게 되고 입력된 학번이 기존의 데이터에 존재하지 않을 경우, 새로운 학생으로 생각되어 이름과 Midterm score, Final score를 입력받게 된다. 입력받는 각 시험의 점수가 정상 범위일 경우 중첩리스트 내 리스트로 학생의 정보들이 추가된다. 만약 기존에 존재하는 학번을 입력할 경우, 이미 존재하는 학번임을 알리고 Midterm score나 Final score가 잘못 입력되었을 때는 점수의 정상 입력범위를 출력한다.

11) remove 함수

- 입력받은 학점이 존재하면 해당 학생의 데이터를 출력하고 삭제할 것인지 재확인한다. 이때 대소문자를 구분하지 않는 "y" 혹은 "n"의 입력을 통해 데이터를 삭제할 것인지 취소할 것인지를 결정하게 된다. 만약 리스트가 완전히 비어있는 경우에는 리스트가 비어있음을 알리고, 학생 데이터에 없는 학번을 입력하게 된다면 데이터에 없는 학생임을 알리는 문구를 출력한다.

11) quit 함수

- 데이터를 저장할 것인지에 대한 대소문자 구분이 없는 "y" 혹은 "n"의 입력받는다. 이를 통해 편집한 데이터를 새로운 파일로 저장하고 종료할 것인지 저장하지 않고 종료할 것인지를 결정하게 된다. 또한 편집될 이름이 저장되는 파일의 이름에는 공백이 없다고 가정되므로, 띄어쓰기가 포함된 파일명이 입력된 경우에도 replace 함수를 이용해 공백이 제거된 파일명으로 저장되도록 한다.

4. 프로그램 실행방법 및 예제

1. 초기 출력화면

1) 파일명이 입력된 경우

```
(base) piai@piai-Precision-7920-Tower:~$ cd AI·BigData_25/
(base) piai@piai-Precision-7920-Tower:~/AI·BigData_25$ python project.py studen
s.txt

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) 파일명이 입력되지 않은 경우

```
(base) piai@piai-Precision-7920-Tower:~/AI·BigData_25$ python project.py

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
-------------------------------------------------------------
```

- 프로그램이 파일명을 읽은 경우, students.txt 파일로부터 정상적으로 데이터를 읽은 것을 확인할 수 있다.
- 파일명을 작성하지 않은 경우에도, default로 students.txt 파일이 열리도록 소스코드가 작성되어 students.txt 의 데이터를 읽어왔음을 알 수 있다.
- 두 경우 모두 읽은 데이터를 통해, 평균(Average)와 학점(Grade)을 생성했고 모든 데이터가 잘 출력되었다.

2. show 함수 실행 결과

```
#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
-------------------------------------------------------------
```

- 사용자가 "show"를 입력함에 따라, 평균(Average)을 key의 인수로 하는 리스트의 sorted() 함수가 사용되어, 파일에서 읽어온 데이터가 평균 기준 내림차순으로 정렬되어 출력되었다.
- 추가로 평균(Average)은 round 함수를 통해 소수점 이하 첫째 자리까지만 출력된 것을 확인할 수 있다.

3. search 함수 실행 결과

1) 학번(Student이 정상적으로 입력된 경우

```
#search
Student ID: 20180002

Student         Name            Midterm Final   Average Grade
-------------------------------------------------------------
20180002         Lee Jieun       92      89      90.5    A
-------------------------------------------------------------
```

2) 학번을 잘못 입력했거나, 없는 학번을 입력한 경우

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

- 사용자가 "search"를 입력함에 따라, 학번(Student)를 입력받게 되고 찾고자 하는 학생이 목록에 있는 경우에 해당 학생의 학번(Student), 이름(Name), 중간고사 점수(Midterm), 기말고사 점수(FInal), 평균(Average), 학점(Grade)를 출력하였다.
- 반면, 사용자가 입력한 학번의 학생이 없는 경우에는 해당 학생이 없다는 의미로 "NO SUCH PERSON."이라는 문구가 출력되었음을 확인할 수 있다.

4. changescore 실행 결과

1) 학번(Student), 시험 종류, 점수 범위가 정상적으로 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
```

2) 학번을 잘못 입력했거나, 없는 학번을 입력한 경우 / 시험 종류를 잘못 입력된 경우 / 점수의 범위가 잘못 입력된 경우

```
#changescore
Student ID: 2018007
NO SUCH PERSON.
```

```
#changescore
Student ID: 20180007
Mid/Final? miid
**********Please Check Again...**********
```

```
#changescore
Student ID: 20180007
Mid/Final? mid
Input new score: 147
**********1부터 100까지의 값을 입력해주세요...**********
```

- 사용자가 "changescore"를 입력함에 따라, 학번(Student), 시험 종류(Midterm/Final), 점수가 모두 잘 입력되었다면 기존 학생의 데이터와 변경된 데이터를 출력하여 성공적으로 변경되었는지 확인할 수 있다.

- 사용자가 입력한 학번의 학생이 없는 경우에는 해당 학생이 없다는 의미로 "NO SUCH PERSON."이라는 문구를 출력하였다.
- 시험의 종류를 입력하는 input 함수를 이용한 "Mid/Final ?"문에서 대소문자를 구별하지 않는 "mid"와 "final" 이외의 문자열에 대해서, "Please Check Again"이라는 문구를 출력함으로써 다시 입력할 수 있도록 하였다
- 이외에도, 수정하고자 하는 점수를 입력받을 때, 0 이상 100 이하의 점수만을 취급하고 있으므로, 그 외의 점수에 대해서는 "1부터 100까지의 값을 입력해주세요..."라는 문구를 출력됨을 확인할 수 있다.

5. add 함수 실행 결과

1) 학번(Student)이 기존에 존재하지 않으며, 중간고사 점수(Midtem), 기말고사 점수(Final)의 점수 범위가 정상적인 경우

```
#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
-------------------------------------------------------------
```

2) 존재하는 학번을 입력한 경우 / 중간고사 점수가 잘못 입력된 경우./ 기말고사 점수가 잘못 입력된 경우

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

```
#add
Student ID: 20180000
Name: Kim Hantag
Midterm Score: 111
************1부터 100까지의 값을 입력해주세요...************
```

```
#add
Student ID: 20180000
Name: Kim hantag
Midterm Score: 100
Final Score: 111
************1부터 100까지의 값을 입력해주세요...************
```

- 사용자가 "add"를 입력함에 따라, 학번(Student)을 입력받게 되고 입력된 학번이 기존의 데이터에 존재하지 않을 경우 새로운 학생으로 생각되어 이름과 Midterm score, Final score를 입력받게 된다. 입력받는 각 시험의 점수가 정상 범위일 경우 중첩리스트 내 리스트로 학생의 정보들이 저장되며, show 함수를 통해 반영 결과를 확인할 수 있다.
- 입력된 학번이 이미 존재하면 "ALREADY EXISTS"를 출력하여 중복됨을 알려주고, Midterm score과 Final score가 1 이상 100 이하의 범위라면 "1부터 100까지의 값을 입력해주세요..." 문구를 출력하여 다시 입력할 수 있도록 했다.

6. searchgrade 함수 실행 결과

1) 입력한 학점을 받은 학생들이 존재하는 경우

```
#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
```

2) 입력한 학점이 존재하지 않는 학점이 아닌 경우

```
#searchgrade
Grade to search: E
해당 학점을 받은 학생이 존재하지 않습니다.
```

3) 입력한 학점이 존재하는 학점이 맞으나, 해당 학점을 받은 학생이 없는 경우

```
#searchgrade
Grade to search: F
NO RESULTS.
```

- 사용자가 "searchgrade"를 입력함에 따라 학점(Grade)를 입력받게 되고, 입력받은 값이 존재하는 학점이며 해당 학점을 받은 학생이 있는 경우 해당 학생들의 데이터를 출력한다.
- 존재하는 학점이나 해당 학점을 받은 학생이 없는 경우 "해당 학점을 받은 학생이 존재하지 않습니다."의 문구를 출력하며, 입력받은 학점이 존재하지 않는 학점이라면, "NO RESULT"라는 문구를 출력하여 결과가 없음을 알린다.

7. remove 함수 실행 결과

1. 입력한 학생 데이터를 지우고자 하는 경우

```
#remove
Student ID: 20180011

Student         Name            Midterm Final   Average Grade
-------------------------------------------------------------
20180011         Ha Donghun      58      68      63.0    D

위의 학생 정보가 삭제됩니다.
정말 삭제하시겠습니까?(Y/N):Y

Ha Donghun 의 학생 정보가 삭제되었습니다.
```

2) 잘못 입력한 학생 데이터를 복구하고자 하는 경우

```
#remove
Student ID: 20180011

Student         Name            Midterm Final   Average Grade
-------------------------------------------------------------
20180011         Ha Donghun      58      68      63.0    D

위의 학생 정보가 삭제됩니다.
정말 삭제하시겠습니까?(Y/N):N
취소되었습니다.
```

3) 학생 데이터를 담은 중첩 리스트가 비어있는 경우

```
#remove
Student ID: 20180001

Student         Name            Midterm Final   Average Grade
-------------------------------------------------------------
20180001         Hong Gildong    84      73      78.5    C

위의 학생 정보가 삭제됩니다.
정말 삭제하시겠습니까?(Y/N):y

Hong Gildong 의 학생 정보가 삭제되었습니다.

#remove
Student ID: 20180007

Student         Name            Midterm Final   Average Grade
-------------------------------------------------------------
20180007         Kim Cheolsu     57      62      59.5    F

위의 학생 정보가 삭제됩니다.
정말 삭제하시겠습니까?(Y/N):y

Kim Cheolsu 의 학생 정보가 삭제되었습니다.

#remove
List is empty.
```

4. 학번을 잘못 입력했거나, 없는 학번을 입력한 경우

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

- 사용자가 "remove"를 입력함에 따라 학점(Grade)를 입력받게 되고, 입력받은 학점이 존재하면 해당 학생의 데이터를 출력하고 삭제할 것인지 재확인한다. 이때 대소문자를 구분하지 않는 "y" 혹은 "n"의 입력을 통해 데이터를 삭제할 것인지 취소할 것인지를 결정하게 된다.
- 이미 학생 데이터가 모두 제거된 빈 리스트가 remove함수의 파라미터로 입력되게 되면, 삭제될 데이터가 없다는 의미로 "List is empty"라는 문구가 출력된다.
- 입력된 학번(Student)이 존재하지 않는 학번이라면 "NO SUCH PERSON" 문구를 출력하여, 학생이 존재하지 않음을 알려준다.

8. quit 함수 실행 결과

1) 편집한 데이터를 저장하고 종료하는 경우

```
#quit
Save data?[yes/no] yes
다음의 데이터를 저장합니다.

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
-------------------------------------------------------------

File name: newStudents.txt
저장이 완료되었습니다!
```

2) 편집한 데이터를 저장하지 않고 종료하는 경우

```
#quit
Save data?[yes/no] no
저장하지 않고 종료합니다.
```

3) 편집한 데이터를 띄어쓰기로 입력된 파일이름에 저장하고 종료하는 경우

```
#quit
Save data?[yes/no] yes
다음의 데이터를 저장합니다.

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
-------------------------------------------------------------

File name: new Students - 띄어쓰기 없애기 . txt
저장이 완료되었습니다!
```

4. 1, 2, 3의 결과로 생성된 파일

- 사용자가 "quit"을 입력함에 따라 데이터를 저장할 것인지에 대한 대소문자 구분이 없는 "y" 혹은 "n"의 입력을 받는다. 이를 통해 편집한 데이터를 새로운 파일로 저장하고 종료할 것인지 저장하지 않고 종료할 것인지를 결정하게 된다.
- 편집될 이름이 저장되는 파일의 이름에는 공백이 없다고 가정되므로, 띄어쓰기가 포함된 파일명이 입력된 경우에도 replace 함수를 이용해 공백이 제거된 파일명으로 저장되도록 하였다.

9. 이외의 프로그램의 전체적 구조

1) 명령어를 잘못 입력하거나, 없는 명령어를 입력한 경우

```
#안녕하세요
******************************wrong input!******************************
****************************다시 입력해 주세요.****************************
```

2) 명령어의 연속적인 입력이 있는 경우

```
#안녕하세요
******************************wrong input!******************************
****************************다시 입력해 주세요.****************************

#안녕히계세요
******************************wrong input!******************************
****************************다시 입력해 주세요.****************************

#또 왔어요
******************************wrong input!******************************
****************************다시 입력해 주세요.****************************

#
```

- 성적 처리 프로그램은 7개의 명령어(show, search, changescore, add, searchgrade, remove, quit)을 제외한 나머지 문자열에 대해 명령이 유효하지 않다는 의미로 "wrong input"이라는 문구를 출력하게 된다.
- 또한 명령어를 받는 input 함수는 while True 문에 속해 있으므로, break 문이 만나지 않는 이상 명령어를 계속 작성할 수 있으며, quit 함수를 통과해야만 편집된 데이터의 저장 여부를 결정하고 프로그램을 종료할 수 있다.

5. 토론

- 명령어로 받으면 실행되는 7개의 명령어에 대해 함수로 구현하였다(show, search, changescore, add, searchgrade, remove, quit). 알고리즘을 구현하는 과정에서 소스 코드 길이가 상당히 길어서 알고리즘이 불필요하게 작성되지 않았는지에 대한 고민이 있었고, 그 결과로 사용자 정의 함수를 통해 소스 코드 길이를 줄였고(start_form, end_form, Grade 함수), 불필요한 반복문이나, 조건문을 최대한 사용하지 않을 수 있도록 다시 알고리즘을 작성하였다.

6. 결론

- 기존에 프로그램이 실행되도록 소스 코드를 작성한 것과 다르게, 최대한 구체적이고 자세한 수행을 실행하면서 최소한의 소스 코드를 이용해 프로그램을 작성하였다. 이를 통해 알고리즘을 설계(순서도, 의사코드)하는 단계에서 자세하고, 꼼꼼하게 작성하는 것이 중요함을 다시금 깨닫게 되었다.