POSCO 청년 AI/Big Data 아카데미

성적 관리 프로그램

16 기 B1 조

장선영

목차

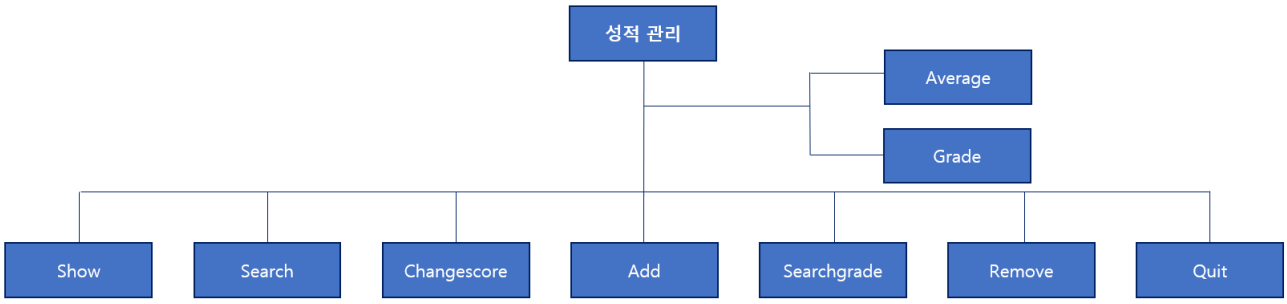
Problem: 성적 관리 프로그램

1. 문제의 개요

본 프로그램을 간략히 설명하면 다음과 같다.

- 사용자가 파일로부터 데이터(학번, 이름, 중간고사 점수, 기말고사 점수)를 읽어 와서 학생들의 성적 목록을 작성한다.
- 데이터를 기반으로 각 학번별 평균과 학점을 입력 받아 출력한다.
- 성적 관리와 관련된 7 개의 명령어를 입력 받아 각 기능을 수행한다.

이 때 사용되는 구상 가능한 구조 차트(structure chart)는 아래와 같이 표현될 수 있다.



- 입력부: 파일 형태의 데이터를 부르고 연다.
- 처리부: 사용자가 입력하는 7 개의 명령어에 따라 실행한다.
- 출력부: 명령어를 실행 후 결과 값을 반영한다.

2. 정의

1) 데이터의 내용

- *학번 (Student ID)*
  학생을 구분하는 고유한 식별자이다.

- *이름*
  학생의 이름을 의미한다.

- *중간고사 점수*
  학생의 중간고사 점수를 의미한다.

- *기말고사 점수*
  학생의 기말고사 점수를 의미한다.

- *평균*
  중간고사 점수와 기말고사 점수의 평균을 의미한다.
  각 시험 별 가중치는 동일하다.

- *학점*
  학점 부여 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

2) 7 개의 명령어 (기능)

명령어를 입력하였을 때만 기능이 실행된다. 이 명령어는 사용자가 명령어 입력 시, 대소문자를 구분하지 않고 동일한 명령어의 기능을 수행하도록 인식한다. 7 개의 명령어(show, search, changescore, add, searchgrade, remove, quit) 이외의 잘못된 명령어 입력 시, 에러 메시지 없이 다시 명령어를 입력 받을 준비를 한다.

7 개의 명령어에 대한 상세 내용은 '4. 내용, 프로그램 구조 및 설명' 부분에 상세히 설명하고자 한다.

3) **대기 모드**

'# ' 은 대기모드로 명령어 입력을 대기하는 것을 의미한다.

3. 알고리즘

1) Pseudo Code

본 프로그램 작성을 위한 알고리즘을 Pseudo Code 형태로 나타내면 다음과 같다.

(1) Main

```
# MAIN

import sys
arg = sys.argv
if (arg의 len이 1보다 크다면) then (해당 파일을 읽는다) : 'python project.py '
else (기본값으로 'students.txt'를 읽는다)

stu_list란 빈 리스트를 작성한다

for 각각의 라인들을 불러온다
    공백에 의해 split 한다
    성과 이름을 같이 합친다
    새로운 인자들을 new_L에 넣고 더 큰 리스트인 stu_list에 넣는다


def average(m, f) 함수 생성
    m + f / 2

def grade(x) 함수 생성
    if 90 <= x 'A'
    elif 80<= x < 90 'B'
    elif 70<= x < 80 'C'
    elif 60<= x < 70 'D'
    else 'F'


stu_list에 average함수와 grade함수로 평균과 학점 추가

평균 점수를 기준으로 내림차순

while (참)
'# ' 소문자 변경
if show 함수 실행
elif searh 함수 실행
elif changescore 함수 실행
elif add 함수 실행
elif searchgrade 함수 실행
elif remove 함수 실행
elif quit 함수 실행
else continue
```

(2) Show 함수

```
# SHOW 함수

stu_list 입력 받으면
평균 점수을 기준으로 내림차순

for stu in stu_list
    stu_list에 있는 값들을 temp에 임시 저장한 후 리스트 형태로 출력한다

(리스트에 리스트 형식)
```

(4) Search 함수

```
# SEARCH 함수

s_id(학번)으 입력 받으면

for i in stu_list
    stu_list[i][0]가 s_id와 같다면
    stu_list[i] 출력
return

else
    continue

s_id가 존재하지 않으면 "NO SUCH PERSON." 출력
```

(5) Changescore 함수

```
# CHANGESCORE 함수

s_id(학번)을 입력 받으면

for i in stu_list의 len
    if s_id와 같다면
        mid / final 구분
        if mid
            new score 입력 받으면

            if new score <= 100
            기존의 값 출력
            "Score changed" 출력
            새로운 값 출력 (new score 반영: new score, 평균, 학점)

            else
                "0~100 사이의 숫자를 입력하시오."
            return

        elif final
            new score 입력 받으면

            if new score <= 100
            기존의 값 출력
            "Score changed" 출력
            새로운 값 출력 (new score 반영: new score, 평균, 학점)

            else
                "0~100 사이의 숫자를 입력하시오."
                return

    else
        "오타를 확인하시오."

        return

"NO SUCH PERSON."
```

(6) Add 함수

```
# ADD 함수

s_id(학번)을 입력 받으면

for stu_list에
    if s_id가 있다면
    "ALREADY EXISTS." 출력
    return

이름을 입력 받는다
중간고사 성적을 입력 받는다
기말고사 성적을 입력 받는다
임의의 리스트(a)에 새로 저장된 정보드를 입력받는다 (average, grade함수로 평균과 성적 반영)
stu_list에 추가
"Student added." 출력
```

(7) Searchgrade 함수

```
# SEARCHGRADE 함수

특정 grade를 입력 받으면

if grade not in 'A', 'B', 'C', 'D', 'F'
    "오타를 확인하시오." 출력
    return

result = False로 고정

빈 리스트 생성 (_L)

for stu_list에
    if grade와 일치하면
        result = True로 변경
        해당 stu_list의 값을 빈 리스트 _L에 추가 (리스트 내 리스트 저장)

if result가 True와 같다면
    grade와 일치하는 리스트들 출력

if result가 False와 같다면
    "NO RESULTS." 출력
```

(8) Remove 함수

```
# REMOVE 함수

if stu_list에 값이 없다면
    "List is empty." 출력

s_id를 입력 받으면

for stu_list
    if s_id가 있다면
        해당 stu_list 제거
        "Student removed." 출력
        return

"NO SUCH PERSON." 출력
```

(9) Quit 함수

```
# QUIT 함수

yes 혹은 no 값을 입력 받으면

if yes
    파일 이름 입력
    평균 점수를 기준으로 내림차순 정렬 진행
    for stu_list
        내용 저장
    입력받은 파일 이름으로 write

else no
    return
```

2) Flow chart

위의 Pseudo Code 알고리즘을 Flowchart 를 통해 표현하면 다음과 같다.

(공동 전제)

- Format 출력

  각 열의 header (Student, Name, Midterm, Final, Average Grade) 출력을 의미한다.

```
  Student        Name       Midterm    Final    Average Grade
--------------------------------------------------------------------
```

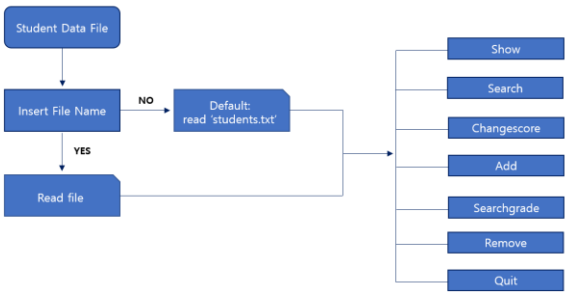
- stu_list

  리스트 내 리스트 구조

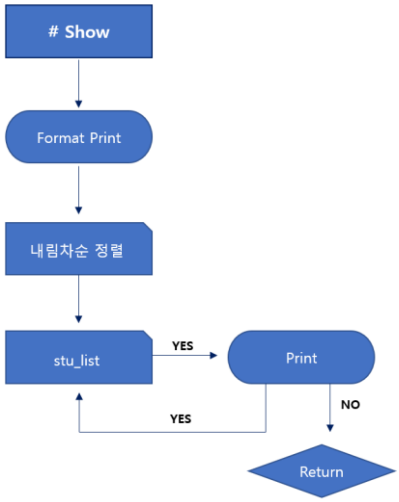
- stu_id

  학번을 의미한다. (student ID)

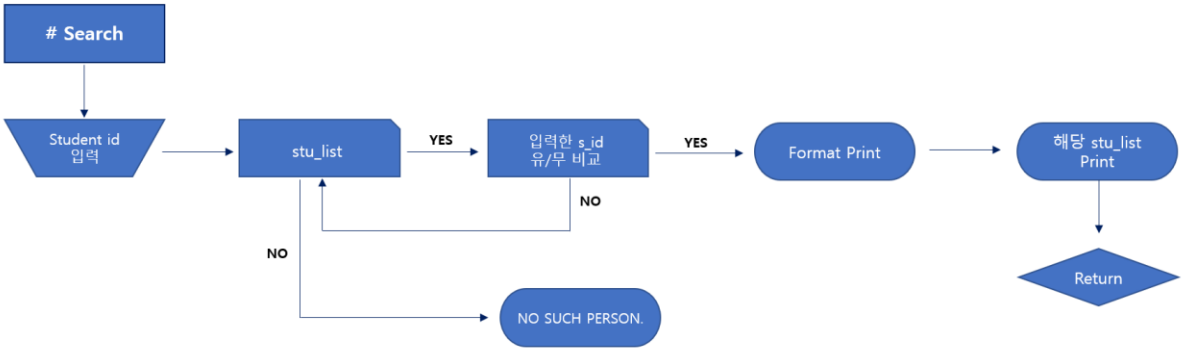
(1) Main



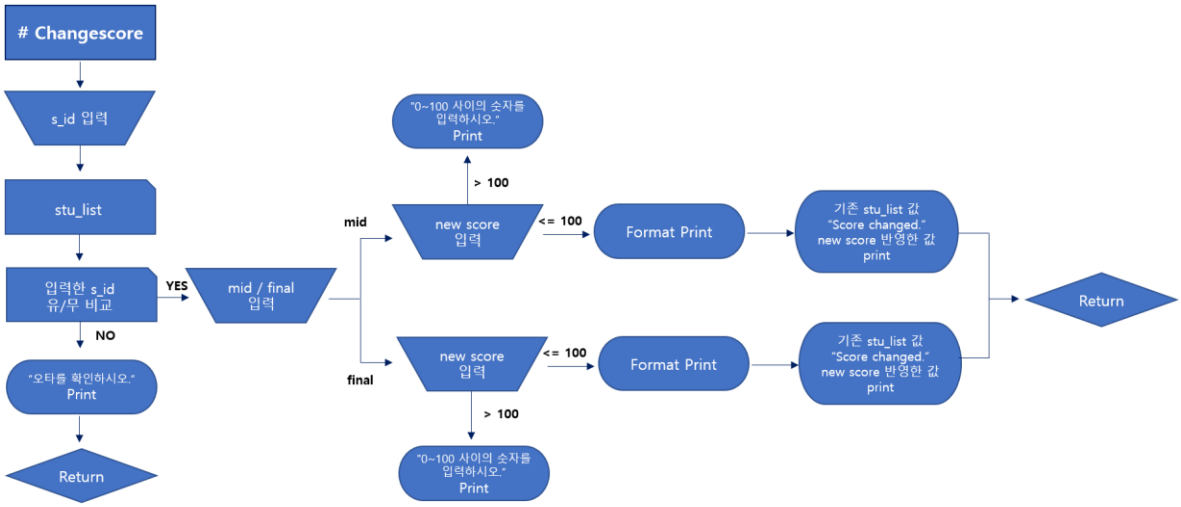
(2) Show



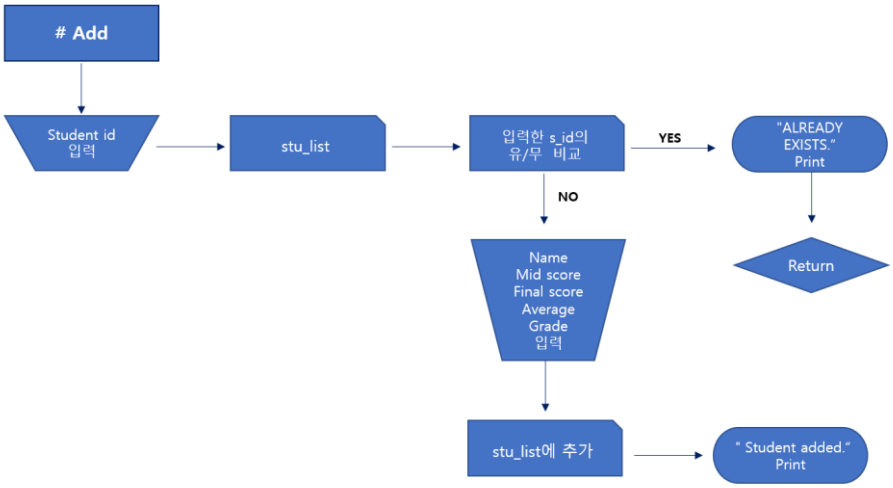
(3) Search



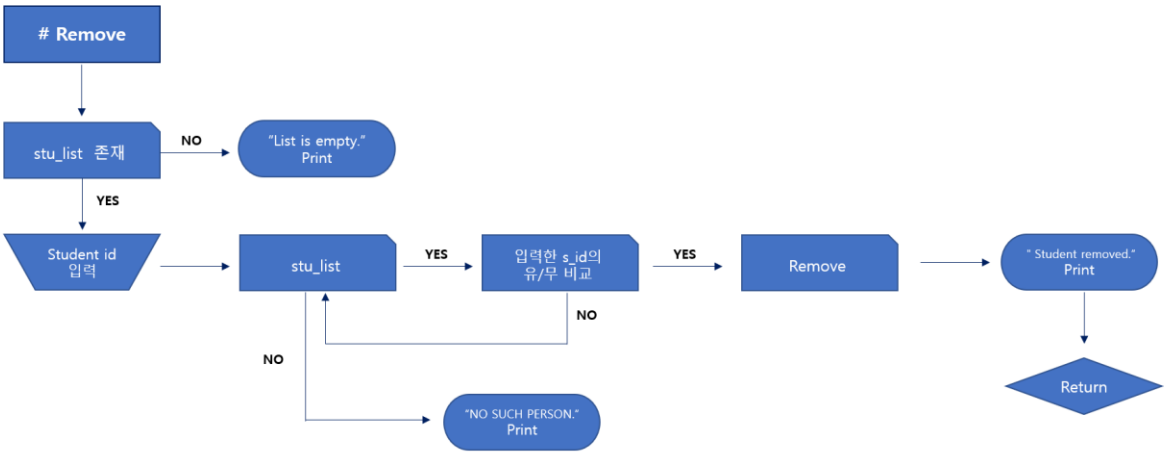
(4) Chagnescore



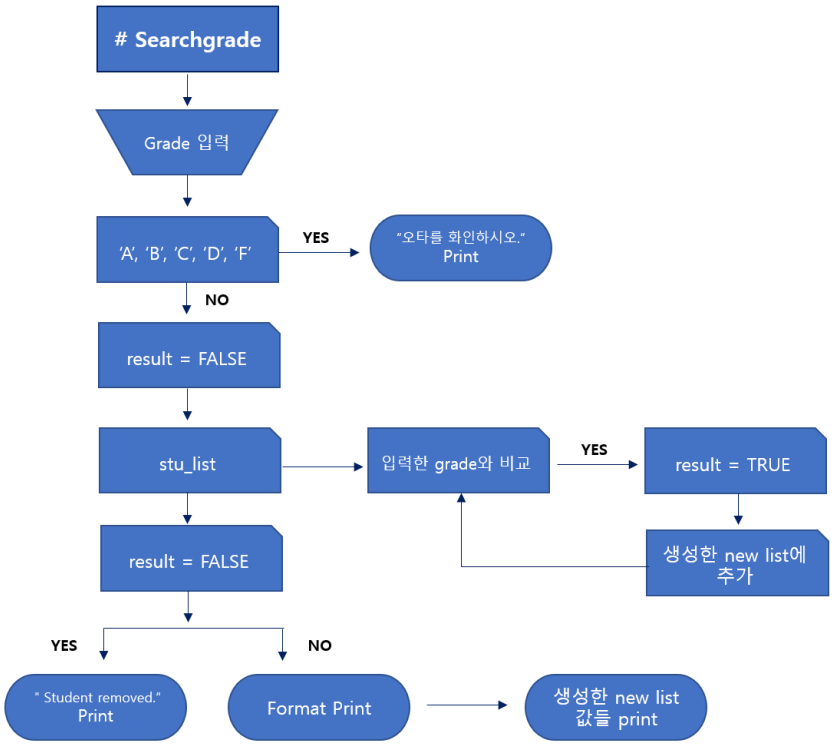
(5) Add



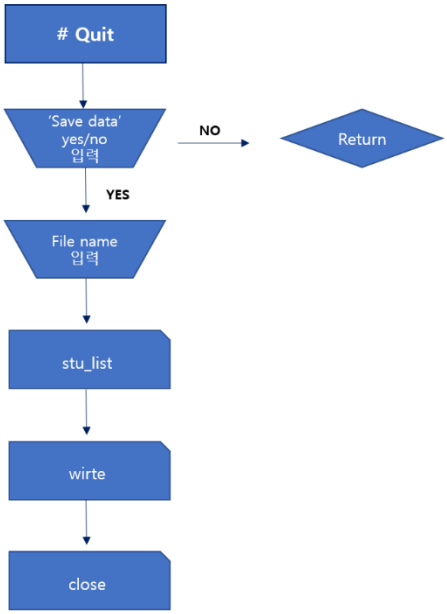
(6) Remove



(7) Searchgrade



(8) Quit



4. 내용 / 프로그램 구조 및 설명

■ *Show*

평균 점수를 기준으로 내림차순으로 전체 학생 정보 출력한다.

■ *Search*

검색하고자 하는 학생의 학번을 입력 받아 특정 학생의 학번, 이름, 중간고사 점수, 기말고사 점수, 평균, 학점을 출력한다.

* 예외

입력 받은 학번이 학생 목록에 없는 경우에는 "NO SUCH PERSON."이라는 에러 메시지를 출력한다.

■ *Changescore*

목록에 저장된 학생 중 1 명의 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 학점도 다시 계산하여 수정한다.

* 예외

학번이 목록에 없는 경우에는 "NO SUCH PERSON."이라는 에러 메시지를 출력한다.

'mid' 또는 'final' 외의 값이 입력된 경우에는 실행되지 않으며 '오타를 확인하시오.'라는 에러 메시지를 출력한다.

점수에 0~100 외의 값이 입력된 경우에는 실행되지 않으며 "0~100 사이의 숫자를 입력하시오."라는 에러 메시지를 출력한다.

■ *Add*

Add 입력 시, 학생의 학번, 이름, 중간고사 점수, 기말고사 점수를 차례대로 요구해 입력 받는다. 추가되면, "Student added"를 같이 출력한다. 평균과 학점은 중간고사 점수와 기말고사 점수를 사용하여 계산하여 저장한다.

학생 추가 후 Show 명령어를 사용하면 평균을 기준으로 내림차순으로 출력한다.

* 예외

목록에 있는 학생의 학번을 입력 시, "ALREADY EXISTS."라는 에러 메시지를 출력한다.

■ *Searchgrade*

Searchgrade 입력 시, 특정 학점을 입력 받아 그 학점에 해당하는 학생을 모두 출력한다.

* 예외

A, B, C, D, F 외의 값이 입력된 경우 실행되지 않는다.

해당 학점의 학생이 없는 경우, "NO RESULTS."를 출력한다.

■ *Remove*

Remove 입력 시, 삭제하고자 하는 학생의 학번을 입력 받은 후, 해당 학생이 목록에 있는 경우 삭제한다. 삭제하면 "Student removed."의 메시지도 같이 출력된다.

* 예외

목록에 아무도 없을 경우 "List is empty." 메시지를 출력한다.

학생이 목록에 없는 경우에는 "NO SUCH PERSON."이라는 에러 메시지를 출력한다.

■ *Quit*

Quit 입력 시, 프로그램을 종료한다.
Quit 이란 명령어를 실행할 경우, 현재까지 편집한 내용의 저장 여부를 묻고, 저장을 선택(yes 입력)할 경우 파일명을 입력 받아서 저장하도록 한다.
저장할 때 목록의 순서는 평균을 기준으로 내림차순으로 한다.

5. 프로그램 실행방법 및 예제

```
PS C:\Users\seony> python project.py
# 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
```

```
PS C:\Users\seony> python project.py students.txt
# 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
```

- text 파일을읽어오며, 아무 값도 입력하지 않더라도 default 로 'students.txt'를 읽어온다.
- 'show' (대소문자 상관 없음, 이하 동일) 명령어 입력 시, 리스트가 출력된다.

```
# 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
```

- 'search' 명령어 입력 후 추가적으로 학번(Student ID)를 입력 시, 해당 학번이 없다면, 'NO SUCH PERSON.'을, 있다면 해당 학번과 일치하는 값을 출력한다.

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

# changescore
Student ID: 20180007
Mid/Final? miid
오타를 확인하시오.

# changescore
Student ID: 20180007
Mid/Final? mid
input new score: 147
0~100 사이의 숫자를 입력하시오.
```

- 'changescore' 명령어 입력 입력 후 추가적으로 학번(Student ID)를 입력 시, 해당 학번이 없다면 'NO SUCH PERSON.'을, 있다면 추가적으로 Mid/Final 여부 및 점수를 입력해야 한다. 이때, mid/final 이외의 값을 입력 시, '오타를 확인하시오.'라는 오류 메시지가 출력된다. 또한 100 을 초과하는 값을 입력 시, '0~100 사이의 숫자를 입력하시오.'란 오류 메시지가 출력된다.

```
# 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
```

- 'changescore' 명령어에 적절한 값들을 모두 입력하면 점수 변경 전 값과 변경 후 평균과 학점이 반영된 값이 함께 출력된다. 그 중간에는 'Score chagned.' 메시지도 함께 뜬다. 'changescore' 명령어 실행 후, 'show' 명령어를 실행해 보면, 변경된 점수가 잘 반영되었음을 확인할 수 있다.

```
# changescore
Student ID: 20180007
Mid/Final? final
input new score: 70
  Student        Name       Midterm    Final   Average Grade
--------------------------------------------------------------
20180007        Kim Cheolsu     75       62       68.5    D
Score changed.
20180007        Kim Cheolsu     75       70       66.0    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       70       66.0    D
20180011        Ha Donghun      58       68       63.0    D
```

```
# 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       70       66.0    D
20180011        Ha Donghun      58       68       63.0    D
```

- 'add' 명령어 수행시, 추가적으로 학번(Student ID) 입력이 필요하다. 기존에 추가되는 학번이 존재 할 경우, 'ALREADY EXISTS.'란 메시지가 출력된다. 입력되는 학번이 존재하지 않을 경우, 추가적으로 이름, 중간고사 점수, 기말고사 점수를 입력하면 반영이 된다.

```
# searchgrade
Grade to search: E
오타를 확인하시오.

# searchgrade
Grade to search: F
NO RESULTS.

# searchgrade
Grade to search: D
  Student        Name       Midterm    Final   Average Grade
----------------------------------------------------------------
20180007        Kim Cheolsu     75       70       66.0    D
20180011        Ha Donghun      58       68       63.0    D
```

- 'searchgrade' 명령어의 경우, 'A, B, C, D, F' 이외의 값을 입력하게 되면 '오타를 확인하시오.'란 값을 출력한다. 리스트에 존재하지 않는 학점을 입력하게 되면 'NO RESULTS.'란 메시지를 출력한다. 리스트에 존재하는 학점을 입력하게 되면 해당 학점을 갖고 있는 모든 값들을 출력한다.

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

# remove
Student ID: 20180011
Student removed.
```

- 'remove' 명령어의 경우 존재하지 않는 학번(Student ID)를 입력 시, 'NO SUCH PERSON.'이란 메시지가 출력한다. 존재하는 학번을 입력 시, 'Student removed.'란 메시지와 함께 반영된다.

```
# 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       70       66.0    D
```

- ‘show’라는 명령어를 통해 ‘remove’ 명령어를 통해 20180011 이란 학번을 가진 값이 모두 제외된 것을 확인할 수 있다.

```
# remove
Student ID: 20180021
Student removed.

# remove
Student ID: 20180002
Student removed.

# remove
Student ID: 20180001
Student removed.

# remove
Student ID: 20180006
Student removed.

# remove
Student ID: 20180007
Student removed.

# remove
Student ID: 20180009
Student removed.

# remove
List is empty.
```

- 리스트에 존재하는 모든 내용들을 지운 후, ‘remove’ 명령어를 실행하면 ‘List is empty.’란 메시지를 출력한다.

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

- ‘quit’ 명령어 입력 시, ‘Save data’ 여부를 물어보며, 이 때 ‘yes’를 입력하게 되면 추가적으로 저장할 파일명을 입력해야 한다. 파일명을 입력하면 새로운 파일로 저장이 된다.

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

- ‘quit’ 명령어 입력 시, ‘Save data’ 여부를 물어보며, 이 때 ‘no’를 입력하게 되면 저장하지 않고 해당 프로그램은 종료가 된다.

6. 토론 및 결론

본 과제는 주어진 명령에만 국한되어 실행할 수 있는 프로그램이기 때문에 실제 활용하기 위해서는 다른 명령어 추가 등 고려를 해야 될 것이다.

본 과제를 수행할 때, list 내의 list 형태로 작성하였으나, dict 형태로 작성될 수 있음을 고려하여 dict 형태로 프로그래밍을 할 수 있도록 노력할 것이다.

Python 의 for 문, if 문, return 입력 위치에 따라 실행 결과가 달라지는데 코드를 짤 때 많은 부분 입력 위치를 적절하게 하지 않아 발생되는 문제들이 많았다.

개인적으로 이런 많은 양의 코드를 짜서 프로그램을 구성하는 경험은 처음이라, 처음부터 체계적으로 하지 못한 점에 대한 아쉬움이 있다. 따라서 다음에는 이런 아쉬움을 보완하고 반영하여 유사코드, 알고리즘 등을 사전에 큰 틀로 짜서 더욱더 체계적인 절차를 갖도록 노력할 것이다.