

파이썬프로그래밍 - 프로젝트

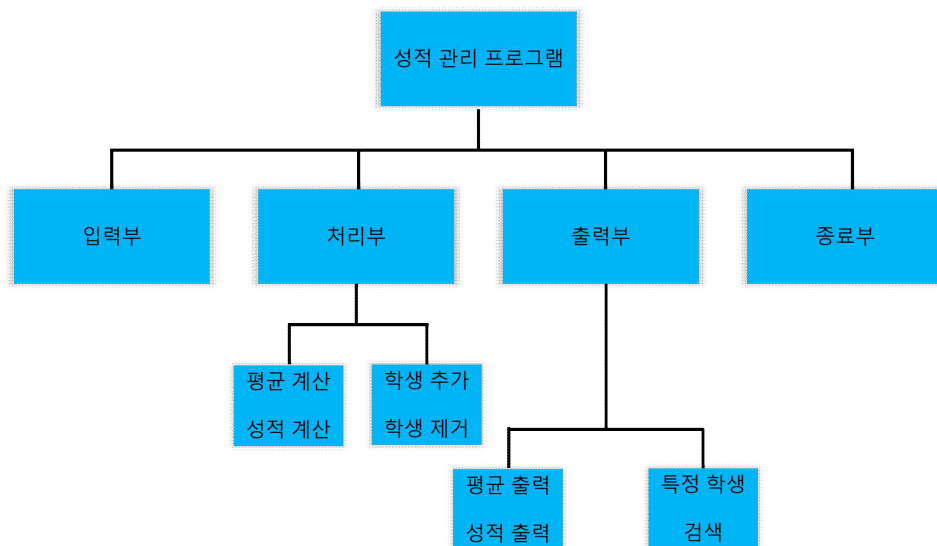
Assignment #1

1. 문제의 개요

본 프로그램을 간략히 설명하면 다음과 같다.

- 사용자로부터 7개의 명령어를 입력 받아 각 기능을 수행하게 된다.
- 각 줄은 각 학생의 학번, 이름, 중간고사 점수, 기말고사 점수로 구성되어 있다.
- 프로그램 실행 시 txt 파일로부터 학생들의 성적 목록 작성을 위한 데이터를 읽을 수 있다.
- 프로그램 실행 시 평균 점수를 기준으로 내림차순으로 정렬하여 출력할 수 있다.

이때 사용되는 구상 가능한 구조 차트는 아래와 같이 표현될 수 있다.



입력부 : 사용자가 입력하는 txt를 리스트에 저장한다.

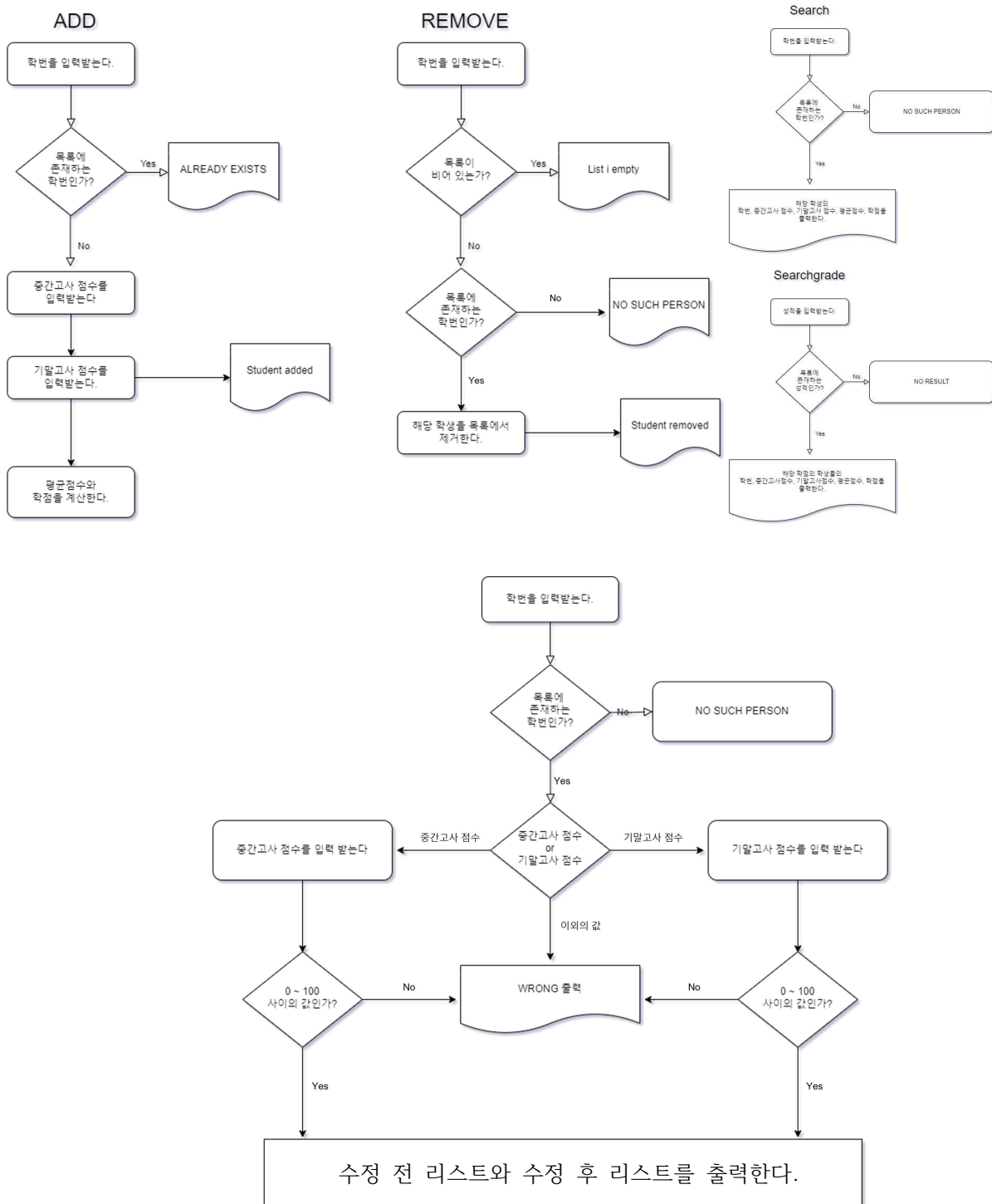
처리부 : 저장된 값을 사용하여 평균 및 성적을 계산하고, 학생을 추가 또는 제거한다.

출력부 : 평균 및 성적을 출력하고 특정 학생을 검색한다.

종료부 : 편집한 내용을 저장하고 프로그램을 종료한다.

2. 알고리즘(Flowchart)

본 프로그램 작성을 위한 알고리즘을 Flowchart를 통해 표현하면 아래와 같다.



3. 프로그램 구조 및 설명

1. 확인을 하고 싶은 학생들의 성적 목록 입력

프로그램을 실행하면 txt 파일로부터 데이터를 읽어 목록을 리스트 자료형을 사용하여 저장하고

전체 목록을 평균 점수를 기준으로 내림차순으로 정렬하여 출력한다.

1) 평균 계산 : 중간고사 점수와 기말고사 점수의 평균을 계산하여 저장한다.

2) 성적 계산 : 성적의 기준은 다음과 같다.

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

2. 프로그램의 기능

프로그램은 아래와 같은 기능을 가진다.

1) 전체 학생 정보 출력(show) : 저장되어 있는 목록을 평균 점수 기준으로 내림차순으로 출력한다.

2) 특정 학생 검색(search) : 검색하고자 하는 학생의 학번을 입력받아 학번, 이름, 중간고사 점수,

기말고사 점수, 평균, 학점을 출력한다.

예외처리 : 찾고자 하는 학생이 목록에 없는 경우에는 에러메세지(NO SUCH PERSON)을 출력한다.

3) 점수 수정(changescore) : 목록에 저장된 학생 중 1명의 중간고사 혹은 기말고사 점수를 수정한다.

수정하고자 하는 학생의 학번, 수정하고자 하는 점수가 중간고사인지 기말고사인지와 수정하고자 하는

점수를 순서대로 입력 받아 해당 학생의 점수를 수정한다.

점수가 바깥에 따라 성적도 다시 계산하여 수정한다.

예외처리 : 학번이 목록에 없는 경우에는 에러메세지(NO SUCH PERSON)을 출력한다.

mid 또는 final 외의 값이 입력된 경우에는 실행되지 않는다.

점수에 0~100 이외의 값이 입력된 경우에는 실행되지 않는다.

4) 특정 학생 추가(add) : 학생의 학번, 이름, 중간고사, 기말고사 점수를 차례로 입력한다.

추가되면, 메시지(student added)를 출력한다.

예외처리 : 목록에 있는 학생의 학번을 입력 시, 에러메세지(ALREADY EXISTS)를 출력한다.

5) 성적 검색(searchgrade) : 특정 학점을 입력 받아 해당하는 학생을 모두 출력한다.

예외처리 : A, B, C, D, F 이외의 값이 입력된 경우 실행되지 않는다.

해당 성적의 학생이 없을 경우, 에러메세지(NO RESULT)를 출력한다.

6) 특정 학생 제거(remove) : 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, 에러메세지(List is empty)를 출력한다.

학생이 목록에 없는 경우에는 에러메세지(NO SUCH PERSON)를 출력한다.

7) 종료(quit) : 입력 시, 프로그램을 종료한다. 명령어 실행 시 현재까지 편집한 내용의 저장 여부를 묻고,

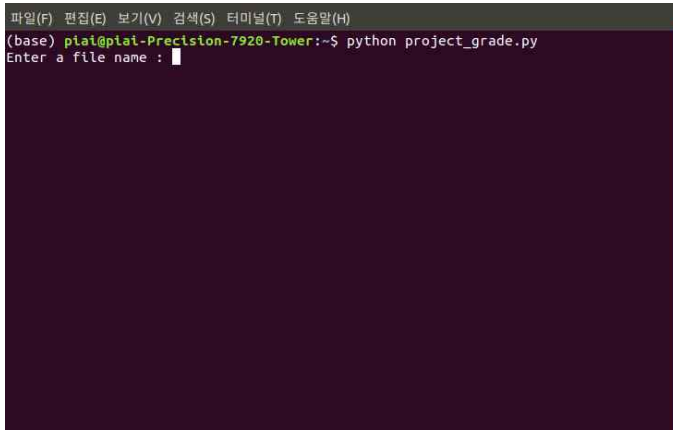
저장을 선택한 경우, 파일명을 입력 받아서 저장한다.

저장 시 목록의 순서는 평균을 기준으로 내림차순으로 한다.

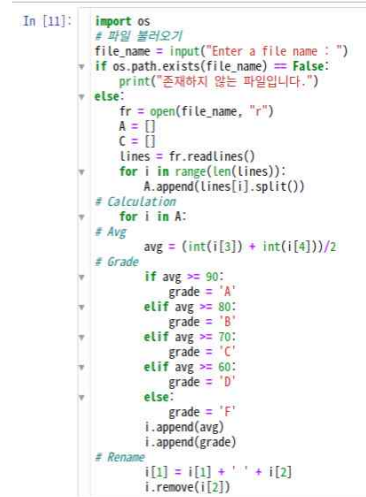
4. 프로그램 실행방법 및 예제

1. 리눅스 환경에서 위 명령어를 통해 작성한 프로그램의 소스코드가 있는지 확인하고 컴파일을 수행한다.

그리고 컴파일이 완료된 파일을 실행한다.

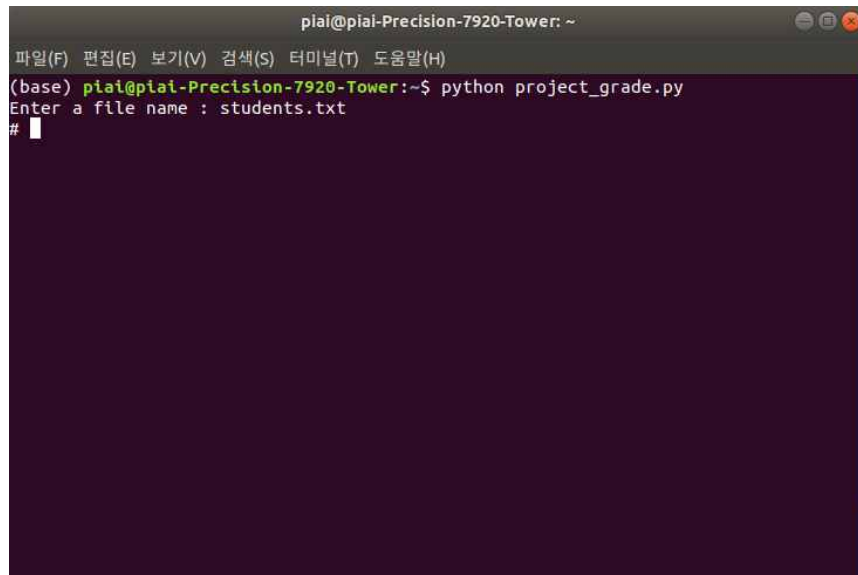


```
파일(F) 편집(E) 보기(V) 검색(S) 터미널(T) 도움말(H)
(base) piat@piat-Precision-7920-Tower:~$ python project_grade.py
Enter a file name : █
```



```
In [11]: import os
# 파일 불러오기
file_name = input("Enter a file name : ")
if os.path.exists(file_name) == False:
    print("존재하지 않는 파일입니다.")
else:
    fr = open(file_name, "r")
    A = []
    C = []
    lines = fr.readlines()
    for i in range(len(lines)):
        A.append(lines[i].split())
    # Calculation
    for i in A:
        # Avg
        avg = (int(i[3]) + int(i[4]))/2
    # Grade
    if avg >= 90:
        grade = 'A'
    elif avg >= 80:
        grade = 'B'
    elif avg >= 70:
        grade = 'C'
    elif avg >= 60:
        grade = 'D'
    else:
        grade = 'F'
    i.append(avg)
    i.append(grade)
# Rename
i[1] = i[1] + ' ' + i[2]
i.remove(i[2])
```

2. 프로그램 동작 시, 확인하고 싶은 학생들의 성적이 입력된 txt 파일 입력을 요청한다.



```
piat@piat-Precision-7920-Tower: ~
파일(F) 편집(E) 보기(V) 검색(S) 터미널(T) 도움말(H)
(base) piat@piat-Precision-7920-Tower:~$ python project_grade.py
Enter a file name : students.txt
# █
```

3. 입력을 마친 후, show 명령어 실행 시 입력되어 있는 학생들의 학번, 이름, 중간, 기말, 평균, 학점이

평균 점수를 기준으로 내림차순으로 정렬되어 출력된다.

출력되는 값들은 가운데 정렬되어 출력되는 것을 확인할 수 있다.

```

파일(F) 편집(E) 보기(V) 검색(S) 터미널(T) 도움말(H)
(base) plat@plat-Precision-7920-Tower:~$ python project_grade.py
Enter a file name : 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
#

```

```

index = ['Student', 'Name', 'Midterm', 'Final', 'Average', 'Grade']

def show(A):
    # Index 출력
    for i in range(len(index)):
        print("{0:^13}".format(index[i]), end=' ')
    print()
    # 구분선 출력
    print('--'*40)
    # 등급 기준 오름차순 정렬
    I_sorted = sorted(A, key = lambda x: x[5])
    # 학생들의 성적 출력
    for I in I_sorted:
        for i in range(len(I)):
            print("{0:^13}".format(I[i]), end=' ')
        print()

```

4. seach 명령어는, 확인하고자 하는 학생의 학번을 입력하면 해당 학생의 학번, 이름, 중간, 기말, 평균, 학점을 출력한다.

잘못된 학번을 입력할 경우 예러 메시지(NO SUCH PERSON)이 출력된다.

```

파일(F) 편집(E) 보기(V) 검색(S) 터미널(T) 도움말(H)
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
# search
Student ID: 20190001
NO SUCH PERSON
# search
Student ID: 20180001
Student      Name      Midterm    Final      Average    Grade
-----
20180001     Hong Gildong 84         73         78.5       C
#

```

```

def search(ID):
    global A
    ID_list = []
    for I in A:
        ID_list.append(I[0])
    if ID not in ID_list:
        print('NO SUCH PERSON')
    else:
        # Index 출력
        for i in range(len(index)):
            print("{0:^13}".format(index[i]), end=' ')
        print()
        # 구분선 출력
        print('--'*40)
        # 해당 학생 성적 출력
        a = ID_list.index(ID)
        for i in range(len(A[a])):
            print("{0:^13}".format(A[a][i]), end=' ')
        print()

```

searchgrade는 특정 학점을 입력받아 해당 학점의 학생들을 모두 출력한다.

```

파일(F) 편집(E) 보기(V) 검색(S) 터미널(T) 도움말(H)
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
# searchgrade
Grade to search: I
# searchgrade
Grade to search: A
Student      Name      Midterm    Final      Average    Grade
-----
20180002     Lee Jieun    92         89         90.5       A
#

```

```

def searchgrade(grade):
    default = ['A', 'B', 'C', 'D', 'F']
    grade_list = []
    count_list = []
    for I in A:
        grade_list.append(str(I[-1]))
        if grade == I[-1]:
            count_list.append(I)
    count = grade_list.count(str(grade))
    if grade in default:
        if count == 0:
            print('NO RESULTS')
        else:
            # index 출력
            for i in range(len(index)):
                print("{0:^13}".format(index[i]), end=' ')
            print()
            # 구분선 출력
            print('--'*40)
            # 해당 성적의 학생 출력
            for I in count_list:
                for i in range(len(I)):
                    print("{0:^13}".format(I[i]), end=' ')
                print()
        else:
            return print('')

```

해당 학점이 목록에 존재하지 않으면 예러메세지(NO RESULT)가 출력된다.

5. 특정 학생의 점수를 수정하고 싶다면, changescore 명령어를 입력하여 수정할 수 있다.

목록에 저장된 학생 1명의 중간고사 혹은 기말고사 점수를 수정한다.

학번이 없다면 에러 메세지(NO SUCH PERSON)이 출력되고,

mid나 final 외의 값이 입력되는 경우 에러메세지(WRONG)이 출력된다.

또한 점수가 0~100 사이의 값이 아닌 경우, 에러메세지(WRONG)이 출력된다.

점수가 수정된다면 수정 전과 후의 평균점수와 학점을 계산하여 출력한다.

```
파일(F) 편집(E) 보기(V) 검색(S) 터미널(T) 도움말(H)

20180007 Kim Cheolsu 57 62 59.5 F
# changescore
Student ID: 20190001
NO SUCH PERSON
# changescore
Student ID: 20180001
Mid/Final? mid
Wrong
# changescore
Student ID: 20180001
Mid/Final? mid
# Input new score: 150
Wrong
Student Name Midterm Final Average Grade
-----
20180001 Hong Gildong 84 73 78.5 C
Score changed.
20180001 Hong Gildong 84 73 78.5 C
#
```

```
파일(F) 편집(E) 보기(V) 검색(S) 터미널(T) 도움말(H)

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
# changescore
Student ID: 20180007
Mid/Final? mid
# Input new score: 85
Student Name Midterm Final Average Grade
-----
20180007 Kim Cheolsu 85 62 59.5 F
Score changed.
20180007 Kim Cheolsu 85 62 73.5 C
#
```

```
def changescore(ID):
    global A
    ID_list = []
    for I in A:
        ID_list.append(I[0])
    if ID not in ID_list:
        return print('NO SUCH PERSON')
    else:
        a = ID_list.index(ID)
        B = A
        mid_final = input("Mid/Final? ").lower()
        if mid_final == 'mid':
            new_score1 = int(input("# Input new score: "))
            if (new_score1 > 100) or (new_score1 < 0):
                print('Wrong')
            else:
                B[a][2] = new_score1
        elif mid_final == 'final':
            new_score2 = int(input())
            if (new_score2 > 100) or (new_score2 < 0):
                print('Wrong')
            else:
                B[a][3] = new_score2
        else:
            return print('Wrong')
    # index 출력
    for i in range(len(index)):
        print("{0:^13}".format(index[i]), end=' ')

    # index 출력
    for i in range(len(index)):
        print("{0:^13}".format(index[i]), end=' ')

    # Score changed
    print('Score changed.')
    # Avg
    avg = (int(A[a][2]) + int(A[a][3]))/2
    # grade
    if avg > 90:
        grade = 'A'
    elif avg > 80:
        grade = 'B'
    elif avg > 70:
        grade = 'C'
    elif avg > 60:
        grade = 'D'
    else:
        grade = 'F'
    A[a][4:6] = [avg, grade]
    # 수정 후 출력
    for i in range(len(B[a])):
        print("{0:^13}".format(B[a][i]), end=' ')
    print()
    A = B
```

6. 학생의 추가 및 제거

add 입력 시, 학생의 학번, 이름, 중간, 기말 점수를 차례로 입력하여 학생을 추가할 수 있다.

```
파일(F) 편집(E) 보기(V) 검색(S) 터미널(T) 도움말(H)

# add
Student ID: 20190001
Name: Song junhee
Name : Midterm Score: 95
95
Final Score: 100
100
Student added
# show
Student Name Midterm Final Average Grade
-----
20180002 Lee Jieun 92 89 90.5 A
20190001 Song junhee 95 100 97.5 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
```

```
파일(F) 편집(E) 보기(V) 검색(S) 터미널(T) 도움말(H)

Enter a file name : show
존재하지 않는 파일입니다.
# quit
Save data?[yes/no] no
(base) pta@ptai-Precision-7920-Tower:~$ python project_grade.py
Enter a file name : 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
# add
Student ID: 20180007
ALREADY EXISTS
#
```

추가가 완료되면 완료메세지(Student added)가 출력되며, 목록에 있는 학생의 학번을 입력하면

에러메세지(ALREADY EXISTS)가 출력된다.

```
def add(ID):
    ID_list = []
    for I in A:
        ID_list.append(str(I[0]))
    if ID not in ID_list:
        Name = input("Name: ")
        print("Name: ", end = ' ')
        Mid_score = int(input("Midterm Score: "))
        print('{0}'.format(Mid_score, end = '\n'))
        Final_score = int(input("Final Score: "))
        print('{0}'.format(Final_score, end = '\n'))
        print('{0}'.format(Final_score, end = '\n'))
        # average
        avg = (Mid_score + Final_score)/2
        # grade
        if avg > 90:
            grade = 'A'
        elif avg > 80:
            grade = 'B'
        elif avg > 70:
            grade = 'C'
        elif avg > 60:
            grade = 'D'
        else:
            grade = 'F'
        new_student = [ID, Name, Mid_score, Final_score, avg, grade]
        A.append(new_student)
    else:
        print('ALREADY EXISTS')
```

특정 학생의 제거의 경우 remove 명령어를 입력하면 제거할 수 있다.

```
# remove
Student ID: 20190002
NO SUCH PERSON
# remove
Student ID: 20180007
Student removed.
# show
Student      Name      Midterm      Final      Average      Grade
-----
20180002     Lee Jieun      92           89          90.5         A
20190001     Song junhee    95           100         97.5         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
# remove
Student ID: 20180002
Student removed.
#
```

```
def remove(ID):
    if len(A) == 0:
        print('List is empty')
    else:
        ID_list = []
        for I in A:
            ID_list.append(str(I[0]))
        if ID not in ID_list:
            print('NO SUCH PERSON')
        else:
            del A[ID_list.index(ID)]
            print('Student removed.')
```

목록에 아무도 없을 경우, 메세지(List is empty)가 출력되고

목록에 존재하지 않는 학번을 입력한 경우, 에러메세지(NO SUCH PERSON)이 출력된다.

목록에 존재하는 학생이 제거가 완료되면 메세지(Student removed)가 출력된다.

7. 프로그램의 종료를 원하는 경우, quit 명령어를 입력하여 종료한다.

```
파일(F) 편집(E) 보기(V) 검색(S) 터미널(T) 도움말(H)
Student      Name      Midterm      Final      Average      Grade
-----
20180002     Lee Jieun      92           89          90.5         A
20190001     Song junhee    95           100         97.5         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
# remove
Student ID: 20180002
Student removed.
# quit
Save data?[yes/no] no
(base) plat@plat-Precision-7920-Tower:~$ python project_grade.py
Enter a file name : students.txt
# quit
Save data?[yes/no] yes
File name: students.txt
(base) plat@plat-Precision-7920-Tower:~$
```

```
def quit():
    default = ['yes', 'no']
    save = input('Save data?[yes/no] ')
    if save in default:
        if save == 'yes':
            print('File name: ', end = ' ')
            name = input()
            Final_list = []
            A_sorted = sorted(A, key = lambda x: x[4])
            for I in A:
                Final_list.append(str(I[3]))
            fs = open(name, "w")
            for I in A_sorted:
                I_cut = I[0:3]
                d1 = "{0}".format(I[0])
                d2 = "{0}".format(I[1])
                d3 = "{0}".format(I[2])
                d4 = "{0}".format(I[3])
                data = "{0}\t{1}\t{2}\t{3}\n".format(d1, d2, d3, d4)
                fs.write(data)
            fs.close()
        else:
            pass
    else:
        return print('Wrong input!')
```

해당 명령어를 실행할 경우, 현재까지 편집한 내용의 저장 여부를 묻는다.(yes/no)

저장(yes)을 선택하면 저장할 파일명을 입력받아 저장한다.

저장 시 목록의 순서는 평균 기준 내림차순이다.

결론

해당과제를 진행하면서 사용자 정의 함수 내에서 조건문과 반복문을 많이 사용하였고, 원하는 값을 출력하기 위한 조건을 알고리즘으로 도식화하고 코드로 옮기는 과정에서 사용자 정의 함수의 사용 방법에 익숙해질 수 있었다. 또한 원하는 값이 입력되지 않았을 때, 메시지를 출력하는 예외처리방법에 대하여 학습할 수 있었다. 또한 이전에는 리스트를 정렬하여 출력하는 방법을 알지 못했지만, 이번 과제를 위해서 문자열 출력의 정렬에 대한 공부를 좀 더 하였고, 반복문을 통해서 리스트의 값을 정렬되게 출력하는 방법을 알게 되었다.