포항공대 AI Bigdata 아카데미 프로젝트

파이썬 1 주차 프로젝트

성적 관리 프로그램

담당교수: 윤은영

제출일자: 2020.01.21

학번: s_0520

학과: 통계학과

이름: 최 영용

명예서약(Honor code)

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

1.프로그램 개요

텍스트 파일로 부터 학생들의 데이터를 읽어서 성적 목록을 만들어 관리하는 성적 관리 프로그램을 작성 하고자 한다. 중간, 기말고사 점수를 읽으면 평균과 점수 등급을 매기며 학생정보를 출력한다. 그리고 조회, 특정 학생 검색, 학점 검색, 점수 수정, 학생 추가, 특정 학생 삭제, 종료 기능을 추가하여 프로그램을 완성한다.

2.성적 관리 프로그램 기능

(학생 정보 출력) 조회 기능 - show 입력 시 저장되어 있는 전체 학생 내림차순으로 조회 된다.

(특정 학생 검색) 검색 기능 - search 입력 시 , 학생의 학번을 요구 받아 학번,이름, 중간고사 점수,기말고사 점수 , 평균, 학점을 출력한다.

(학점 검색) 검색 기능 - searchgrade 입력 시, 특정 학점을 입력 받아 그 학점에 해당하는 학생을 모두 출력한다.

(점수 수정) 수정 기능 - changescore 입력 시, 수정하고자 하는 학생의 학번, 수정하고자 하는 시험을 입력 받아 해당 학생의 점수를 수정한다.

(학생 추가) 추가 기능 - add 입력 시, 학번, 이름, 중간고사 점수, 기말고사 점수를 차례로 입력 받는다. 평균과 학점은 중간고사, 기말고사 점수를 사용하여 계산 후 저장된다 .

(특정 학생 삭제) 삭제 기능 - remove 입력 시, 삭제하고자 하는 학생의 학번을 입력 받은 후 , 학생이 목록에 있는 경우 삭제한다.

(종료) 종료 기능 - quit 입력 시 , 프로그램을 종료한다.

※입력 시 모든 명령어는 대소문자 구분 없이도 정상적으로 기능을 수행 한다.

3. 알고리즘

- 본 프로그램 작성을 위한 알고리즘을 Pseudo 코드 형태로 나타내면 다음과 같다.

1)main 함수 알고리즘

Pseudo-algorithm for main fuction

// 프로그램에 필요한 변수들은 미리 선언해놓은 것으로 가정한다.

```
1 Read python.py from terminal
2 Open the ‘students.txt’ file
3 Save file aruments in dictionary and Calculate the average and grade
4 Print out all student information  and Prepare to receive input
5 *1)Function is executed when input is *2)normally received
6 When input the quit, the program exits and the Save as txt file
```

*1) 기능 관련 알고리즘은 다음 페이지 참조
*2) 정상적으로 input 을 입력 못받을시엔 다시 입력 받는다.

2) Show 함수 알고리즘 - (학생 정보 출력) 조회 기능

Pseudo-algorithm for show fuction

// 프로그램에 필요한 변수들은 미리 선언해놓은 것으로 가정한다.

1 'Show' is entered for input
2 Take student information as argument
3 Sort in descending order by average score
4 Print all student information in the output window
5 Exit the function and get ready for input again

3) Search 함수 알고리즘 - (특정 학생 검색) 검색 기능

Pseudo-algorithm for Search fuction

// 프로그램에 필요한 변수들은 미리 선언해놓은 것으로 가정한다.

1 'Search' is entered for input
2 Take student information as argument
3 Enter the student_ID to find
4 Check student information for input
5 *3) If True, print out the student's information.
6 Exit the function and get ready for input again

*3) False 라면 "NO SUCH PERSON" 출력 후 함수 탈출

4) Search_grade 함수 알고리즘 - (학점 검색) 검색 기능

Pseudo-algorithm for Search_grade fuction

// 프로그램에 필요한 변수들은 미리 선언해놓은 것으로 가정한다.

1 'Grade' is entered for input
2 Take student information as argument
3 Enter the grade to find
4 Check grade information for input
5 *4)if Grade information exists, Print all of the students with that score
6 Exit the function and get ready for input again

*4) 학생정보가 존재하지 않거나 입력 등급의 학점이 존재하지 않는다면 함수 탈출

5) Changescore 함수 알고리즘 - (점수 수정) 수정 기능

Pseudo-algorithm for changescore fuction

// 프로그램에 필요한 변수들은 미리 선언해놓은 것으로 가정한다.

1 'Changescore' is entered for input
2 Take student information as argument
3 Enter the Student_ID to change the score
4 Check if student_ID input is in student information
5 *5) If it exists, select the type of test you want to change (mid or final)
6 Enter the score you want to change
7 Print student information before changing and after
8 Exit the function and get ready for input again

*5) 해당 등급의 학점이 존재하지 않는다면 , "NO SUCH PERSON" 출력 후 함수 탈출

6) Add 함수 알고리즘 - (학생 추가) 추가 기능

Pseudo-algorithm for add fuction

// 프로그램에 필요한 변수들은 미리 선언해놓은 것으로 가정한다.

1 'Add' is entered for input
2 Take student information as argument
3 Enter the Student_ID to add the student
4 Check if student_ID input is in student information
5 *6) If it not exists , Enter name, middle and final scores and print "Student added"
6 Calculate average and grade and add to student information
7 Exit the function and get ready for input again

*6) 해당 학번의 학생이 존재하면 , "ALREADY EXISTS" 출력 후 함수 탈출

7) remove 함수 알고리즘 - (특정 학생 삭제) 삭제 기능

Pseudo-algorithm for remove fuction

// 프로그램에 필요한 변수들은 미리 선언해놓은 것으로 가정한다.

1 'Remove' is entered for input
2 Take student information as argument
3 Enter the remove_ID to remove the student
4 Check if remove_ID input is in student information
5 *7) If it exists, delete student[reove_ID] and print "student removed"
6 Exit the function and get ready for input again

*7) 해당 학번의 학생이 존재하지 않으면 , "NO SUCH PERSON" 출력 후 함수 탈출

8) Quit 함수 알고리즘 - (종료) 종료 기능

Pseudo-algorithm for quit fuction

// 프로그램에 필요한 변수들은 미리 선언해놓은 것으로 가정한다.

1 'Quit' is entered for input
2 Take student information as argument
3 Prompt whether to save and enter yes / no
4 *8) if 'yes' , Save all student information in a new file
5 Exit the function and End main()

***7) 'no' 입력시 함수 탈출 후 min() 종료**

3. 코드 설명 및 구현

1) 보조 함수 정의 (정렬, 평균 및 등급계산, 출력 바 , 학생 정보 출력)

i . re_sort()

정렬 함수 평균값을 기준으로 내림차순으로 정렬한다.

```
def re_sort(Student):
    sorted_sl = sorted(Student.items(), key = lambda a: a[1][3],reverse=True)
    return sorted_sl
```

ii . calc_score()

중간,기말 점수를 인자로 받아 average(평균), grade(성적 등급)를 리턴한다.

```
def calc_score(mid, final):
    average = (float(mid)+float(final))/2
    grade=''
    if(average>=90):
        grade = 'A'
    elif(average>=80):
        grade = 'B'
    elif(average>=70):
        grade = 'C'
    elif(average>=60):
        grade = 'D'
    elif(average<60):
        grade = 'F'
    return average , grade
```

iii . bar() , per_show()

bar()는 학생 정보를 출력하기 열이름을 출력 해준다
per_show()는 학생 번호가 적힌 ID 와 그것을 제외한 stu 의 정보를 받아서 하나의 학생의 정보를 출력해준다.

```
def bar():
    print('Student \t   Name    Mideterm    final     Average    Grade')
    print('-----------------------------------------------------------------')


def per_show(stu, ID):
        print(ID+'  ',end='')
        print('%13s' %(stu[0])+'         ',end='')
        print(str(stu[1])+'          ',end='')
        print(str(stu[2])+'        ',end='')
        print(str(stu[3])+'       ',end='')
        print(stu[4], end='\n')
```

2) Main() 함수 정의

Student 라는 모든 학생의 정보를 받을 딕셔너리 변수를 설정한다. 그리고 현재 학생의 정보인 'student.txt'를 열어서 Student 변수에 추가해주며 모든 학생을 출력한다. 그리고 while 문 loop 를 시행하여 프로그램 기능의 값을 입력 받는다.

```
def main():
    Student=dict()
    f=open('students.txt','r')
    bar()

    for line in f:
        stu = line.strip().split('\t')
        name = stu[1]
        average, grade= calc_score(stu[2],stu[3])
        Student[stu[0]]=[name,stu[2],stu[3],str(average),grade]
    f.close()


    for stu in re_sort(Student):
        per_show(stu[1],stu[0])


    while(1):
        order = input().upper()
        if(order=='SHOW'): all_show(Student)
        elif(order=='SEARCH'): search_num(Student)
        elif(order=='CHANGESCORE'): changescore(Student)
        elif(order=='SEARCHGRADE'): search_grade(Student)
        elif(order=='ADD'): input_stu(Student)
        elif(order=='REMOVE'): stu_remove(Student)
        elif(order=='QUIT'): quit(Student); break

if __name__=='__main__':
    main()
```

예외처리 : upper()을 이용하여, 기능 입력값의 대소문자를 구분하지 않고 동일한 명령어의 기능을 수행하도록 한다. 7 개의 명령어 이외의 명령어를 입력 시 에러 메시지 없이 다시 명령어를 받을 준비를 한다.

main 함수를 호출 했을때의 현재 학생들의 현황과 입력값을 받는다.

```
(base) pirl@pirl-Precision-7920-Tower:~$ python project1.py
Student             Name    Mideterm    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
```

3) all_show() 함수 정의

학생정보가 담긴 Student 를 인자로 받은뒤 bar(), re_sort() 함수를 호출한뒤 per_show()함수를 이용하여 한사람씩 출력창에 학생들을 출력한다.

```
#조회함수
def all_show(Student):
    bar()
    for stu in re_sort(Student):
        per_show(stu[1],stu[0])
```

show 를 입력시에 나오는 출력창이다.

```
show
Student            Name    Mideterm    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
```

4) search_num() 함수 정의

학생정보가 담긴 Student 를 인자로 받은뒤 검색할 학생의 학번을 stu_id 로 받는다.

예외처리:

찾고자 하는 학생이 목록에 없는 경우에는 “NO SUCH PERSON.” 이라는 에러 메시지를 출력

```
def search_num(Student):
    stu_id= input("Student ID: ")
    if(stu_id not in Student):
        print("NO SUCH PERSON.")
    else:
        bar()
        per_show(Student[stu_id],stu_id)
```

```
search
Student ID: 20180002
Student            Name    Mideterm    final Average    Grade
-----------------------------------------------------------------
20180002       Lee Jieun         92         89        90.5       A
search
Student ID: 12345678
NO SUCH PERSON.
```

5) changescore() 함수 정의

학생정보가 담긴 Student 를 인자로 받은뒤 점수를 수정할 학생의 학번을 stu_id 로 받는다. Mid, final 중 수정할 시험을 결정하고 점수를 입력한다. 그리고 Student 의 인자에 수정한 시험의 성적을 입력 시킨다. 그리고 수정전 정보와 수정후 정보를 출력한다.

예외처리:
찾고자 하는 학생이 목록에 없는 경우에는 “NO SUCH PERSON.” 이라는 에러 메시지를 출력.
mid/fial 의 대소문자 구분 없이 입력을 받음.
수정할 점수가 0~100 점 사이만 가능.

```
#점수 수정 함수
def changescore(Student):
    stu_id=input("Student ID:")
    modify_score=0

    if(stu_id not in Student.keys()):
        print("NO SUCH PERSON")
    else :
        mid_final = input("Mid/Final?").upper()
        if(mid_final == 'MID' or mid_final == 'FINAL'):

            new_score = int(input("Input new score: "))
            if(new_score<0 or new_score>=100):
                return
            bar()
            per_show(Student[stu_id],stu_id)

            if (mid_final=='MID'):
                    Student[stu_id][1] = new_score

            elif (mid_final=='FINAL'):
                    Student[stu_id][2] = new_score

        average, grade= calc_score(Student[stu_id][1],Student[stu_id][2])
        Student[stu_id][3] = average
        Student[stu_id][4] = grade
        per_show(Student[stu_id],stu_id)
    else: return
```

찾는 학번이 없는 경우

```
changescore
Student ID:2018002
NO SUCH PERSON
```

수정한 점수 0~100 점이 아닌경우

```
changescore
Student ID:20180002
Mid/Final?mid
Input new score: 110
```

점수 수정후 출력 되는 과정

```
changescore
Student ID:20180002
Mid/Final?mid
Input new score: 90
Student         Name    Mideterm    final Average    Grade
-----------------------------------------------------------
20180002      Lee Jieun        92          89       90.5      A
20180002      Lee Jieun        90          89       89.5      B
```

6) search_grade() 함수 정의

학생정보가 담긴 Student 를 인자로 받은뒤 조회하고 싶은 등급을 grade_arg 에 받는다. Student 의 정보를 이용하여 같은 등급의 학생을 조회한다.

예외처리:
등급을 잘못 입력시에는 함수를 빠져나간다.
조회할 등급이 없는경우는 공백을 출력한다.

```python
#같은 등급의 학점 학생 검색 기능
def search_grade(Student):
    grade_arg= input("Grade to search: ").upper()
    stu_id_list = []
    grade_list=['A','B','C','D','F']
    stu_id_list = list(Student.keys())
    same_grades = []

    if(grade_arg not in grade_list):
        return
    else:
        for stu_id in stu_id_list:
            if Student[stu_id][4] == grade_arg:
                same_grades.append(stu_id)
        if len(same_grades) == 0:
            print("")
        else:
            bar()
            for ID in same_grades:
                per_show(Student[ID],ID)
```

조회한 등급의 학생이 없는경우 공백 출력

```
searchgrade
Grade to search: A
```

등급 입력을 잘못한 경우

```
searchgrade
Grade to search: qwe
```

올바르게 등급을 조회 한경우

```
searchgrade
Grade to search: b
Student         Name    Mideterm    final Average    Grade
-----------------------------------------------------------------
20180002      Lee Jieun        90          89        89.5       B
20180009   Lee Yeonghee        81          84        82.5       B
```

7) input_stu 함수 정의

학생정보가 담긴 Student 를 인자로 받은뒤 추가할 학번을 stu_num 에 받는다. 그리고 이름과 중간,기말 점수를 받고 “Student added”를 출력한다. 평균과 등급을 매긴후 Student 에 추가한다.

예외처리:
학번의 길이를 8 로 고정하기 위해 길이가 8 인지 확인.
입력할 학번이 이미 존재한다면 “ALREADY EXISTS” 출력.

```
def input_stu(Student):

    stu_num = input("Student ID: ")
    if(len(stu_num)!=8):
        print("Length only 8")
        return
    elif(stu_num in Student):
        print("ALREADY EXISTS.")
        return
    name = input("Name: ")
    mid_score = input("Midterm Score : ")
    final_score = input("final Score: ")

    print("Student added")


    average, grade= calc_score(mid_score,final_score)

    Student[stu_num]=[name,mid_score,final_score,str(average),grade]
```

정상적으로 학생을 입력후 출력한 모습

```
Student ID: 21410781
Name: c YY
Midterm Score : 70
final Score: 80
Student added
show
Student            Name    Mideterm    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
21410781           c YY          70          80        75.0       C
20180011      Ha Donghun          58          68        63.0       D
20180007     Kim Cheolsu          57          62        59.5       F
```

학번 길이가 8 이 아닌경우

```
add
Student ID: 123456
Length only 8
```

이미 존재하는 학번을 입력한 경우

```
add
Student ID: 21410781
ALREADY EXISTS.
```

8) input_stu 함수 정의

학생정보가 담긴 Student 를 인자로 받은뒤 제거할 학번을 remover_id 에 받는다. 그리고 학번을 찾으면 학생의 정보를 del 을 이용하여 지운뒤 “student removed” 출력

예외처리:
학생 정보가 없을시에는 “List is empty” 출력
잘못된 학번을 입력할시에는 “NO SUCH PERSON.” 출력

```
#학생 정보 제거 함수
def stu_remove(Student):
    if(len(Student)==0):
        print('List is empty')
        return
    remover_id = input("Sudent ID: ")
    stu_id_list = list(Student.keys())

    if(remover_id not in stu_id_list):
        print("NO SUCH PERSON.")
    else :
        del Student[remover_id]
        print("student removed")
```

학번을 입력하고 학생의 정보를 지운다.

```
REMOVE
Sudent ID: 20180002
student removed
show
Student            Name    Mideterm    final Average    Grade
-----------------------------------------------------------------
20180009    Lee Yeonghee        81          84        82.5       B
20180001    Hong Gildong        84          73        78.5       C
21410781           c yy         70          80        75.0       C
20180011      Ha Donghun        58          68        63.0       D
20180007     Kim Cheolsu        57          62        59.5       F
```

학번을 잘못 입력 한 경우

```
remove
Sudent ID: 12345678
NO SUCH PERSON.
```

모든 학생을 지운뒤 remove 입력 값을 준경우

```
remove
List is empty
```

9) input_stu 함수 정의

학생정보가 담긴 Student 를 인자로 받은뒤 파일 세이브 여부를 save_bool 에 입력시킨다. 그리고 'yes' 입력시에는 저장할 텍스트 파일명을 save_txt 에 입력 받는다. 현재 Student 에 저장되어있는 학생들의 정보를 save_txt 에 담긴 이름으로 저장한다.

예외처리:
세이브를 원하지않는 'NO' 입력시에는 파일 저장을 하지않고 프로그램을 종료 한다.

```
#프로그램 종료 함수
def quit(Student):
    save_bool= input("Save data?[yes/no]: ").upper()
    if(save_bool == "YES"):
        save_txt = input("File name: ")
        f= open(save_txt , 'w')

        for v in re_sort(Student):
            num=v[0]+'\t'
            name=v[1][0]+'\t'
            mid=v[1][1]+'\t'
            final=v[1][2]+'\n'
            data = (num + name + mid + final)
            f.write(data)
    elif(save_bool == 'NO'): return
```

new.txt 로 파일을 저장한 경우

```
Student             Name    Mideterm    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
quit
Save data?[yes/no]: yes
File name: new.txt
```

jupyter new.txt 2 minutes ago

File Edit View Language

```
1  20180002→Lee Jieun→92→89
2  20180009→Lee Yeonghee→81→84
3  20180001→Hong Gildong→84→73
4  20180011→Ha Donghun→58→68
5  20180007→Kim Cheolsu→57→62
6
```

5. 토론

- 모든 학생의 정보를 받는 Student 변수를 list 가 아닌 딕셔너리로 저장한것이 더욱 수훨하게 프로그램을 제작 할 수 있었다.
- 자주 사용되는 코드들을 함수를 선언하여 코드의 길이를 줄일수 있게 되었다.

6. 결론

- 프로그램을 제작하면서 input 을 이용하여 학생의 정보를 입력 받아 현재의 학생 정보와 비교하며 예외처리를 하는 방법을 학습할 수 있게 되었다.

7. 개선방향

- 본 과제에서 제안하는 예외 처리에 집중하여 프로그램을 만들었지만, 중간 고사, 기말 고사 성적을 입력 받을때에 정수만을 입력 받게 만들면 오류가 없는 프로그램이 될 것으로 예상한다.