파이썬 프로그래밍 프로젝트

<성적 관리 프로그램>

이름 : 조현서
이메일 : theeastside@naver.com
날짜 : 2024년 5월 13일

명예서약(Honor code)

나는 이 프로그래밍 과제를 다른 사람의 부적절한 도움 없이 완수하였습니다.

1. 문제의 개요

이 프로그램은 학생 성적 관리 시스템으로 간략히 설명하면 다음과 같다.

- 주어진 텍스트 파일에서 학생 데이터를 읽어와 학생 학번, 이름, 중간고사 점수, 기말고사 점수를 기반으로 평균 점수와 학점을 계산하고 관리한다.
- 사용자에게 학생들의 성적 데이터를 조회, 수정, 추가, 제거 및 파일로 저장하는 기능을 제공한다.

이때 사용되는 주요 기능을 가진 함수들은 아래와 같다.

(1) 데이터 로드 및 딕셔너리 저장 : 텍스트 파일에서 학생 데이터를 읽어와 저장한다.
(2) 전체 학생 정보 출력 : 저장된 학생 정보 전체 목록을 화면에 출력한다.
(3) 학생 검색 : 학번을 입력 받아 해당 학생이 목록에 있는지 검색한다.
(4) 성적 수정 : 학번을 입력 받아 해당 학생의 점수를 수정한다.
(5) 학생 추가 : 새로운 학생을 목록에 추가한다.
(6) 성적 검색 : 성적을 입력 받아 해당 성적을 가진 학생이 목록에 있는지 검색한다.
(7) 학생 삭제 : 학번을 입력 받아 해당 학생을 목록에서 삭제한다.
(8) 종료 : 편집한 내용을 저장하고 프로그램을 종료한다.

2. 알고리즘 의사코드

본 프로그램에서 쓰이는 주요 함수에 대한 의사 코드는 다음과 같다.

Load student data
1 Read data from file: 2 Open file 3 Read data from file: 4 Repeat for each line: 5 Split data by tab 6 Extract student ID, name, midterm score, final score 7 Calculate average score 8 Determine grade 9 Store student information in dictionary 10 Close file

Show all the students
1 View all student information: 2 Sort student info by average score in descending order

```
3     Print table header
4     Print sorted student information
```

Show specific student

```
1  View specific student information:
2     Get student ID input
3     Print student information for given ID
```

Search a student

```
1  Search for a student:
2     Get student ID input
3     Check if entered ID exists in the student dictionary
4     If exists, print the student's information
5     If not, print "NO SUCH PERSON"
```

Change score

```
1  Change score:
2     Get student ID input
3     Check if the student exists
4     If exists:
5        Get input for exam type (midterm or final)
6        Get input for new score
7        Check if new score is within valid range (A,B,C,D,F)
8        If valid:
9           Update existing score
10          Calculate new average
11          Determine new grade
12          Print updated information
13       If not valid, print warning message
14    If student does not exist, print "NO SUCH PERSON"
```

Search grade

```
1  Search by grade:
2     Get grade input
3     Check if the input grade is valid
4     If valid:
5        Find all students with the given grade
6        If students found, print their information
7        If no students found, print "NO RESULTS"
8     If grade not valid, print warning message
```

Add student

```
1 Add a student:
2     Get new student ID input
3     Check if ID already exists
4     If not existing:
5         Get name, midterm score, final score input
6         Calculate average score and grade
7         Add new student info to the dictionary
8         Print "Student added."
9     If ID exists, print "ALREADY EXISTS."
```

Remove student

```
1 Remove a student:
2     Get student ID input
3     Check if the student exists
4     If exists:
5         Remove student info
6         Print "Student removed."
7     If not exists, print "NO SUCH PERSON."
```

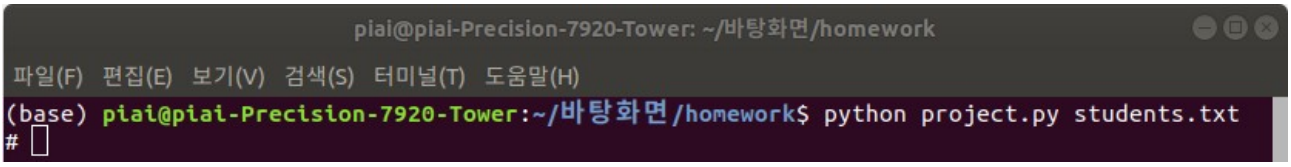
Quit the program

```
1 Exit program and save data:
2     Print "Save Data?[yes/no]" and get input
3     If input is "yes":
4         Get filename input
5         Open file
6         Sort student info by average score and save to file
```

3. 예제

(1) 프로그램 실행

파이썬 코드를 실행시키면 students.txt 텍스트 파일을 읽어와 사용자의 명령어를 입력받을 준비를 한다.



(2) 전체 학생 정보 출력

show 명령 입력 시, 저장된 학생 전체 목록을 평균 점수를 기준으로 내림차순 정렬하여 출력한다.

```
# 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
#
```

(3) 특정 학생 검색

search 명령 입력 시, 검색하고자 하는 학생의 학번을 입력받아 학생 정보를 출력한다. 단, 찾고자 하는 학생이 목록에 없을 경우 NO SUCH PERSON 이라는 에러 메시지를 출력한다.

```
# 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
#
```

(4) 점수 수정

chagescore 명령 입력 시, 학번을 입력받아 해당 학생의 중간고사 점수 혹은 기말고사 점수를 수정한다. 학번이 없을 경우 NO SUCH PERSON 이라고 에러 메시지를 출력한다. Mid, final 이외의 값이 입력된 경우 실행하지 않는다. 점수에 0~100 외의 값이 입력된 경우 실행하지 않는다.

```
# changescore
Student ID: 20180050
NO SUCH PERSON
# changescore
Student ID: 20180007
Mid/Final? miid
# changescore
Student ID: 20180007
Mid/Final? mid
Input new score: 147
# 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
#
```

(5) 학생 추가

add 명령 입력 시, 학생 정보를 입력받아 새로운 학생을 추가한다. 목록에 있는 학생의 학번을 입력 시 ALREADY EXISTS 라는 에러 메시지를 출력한다.

```
# add
Student ID: 20180021
Name: Jo Hyunseo
Midterm Score: 100
Final Score: 100
Student added.
# show
Student               Name  Midterm     Final     Average     Grade
-------------------------------------------------------------------
20180021        Jo Hyunseo    100        100       100.0         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
20180007       Kim Cheolsu     75         62        68.5         D
20180011        Ha Donghun     58         68        63.0         D
#
```

(6) grade 검색

searchgrade 명령 입력 시, 특정 grade를 입력받아 그 grade에 해당하는 학생을 모두 출력한다. A, B,C,D,F 외의 값이 입력된 경우 실행되지 않는다. 해당 grade 학생이 없는 경우 NO RESULT라 출력한다.

```
# searchgrade
Grade to search: E
# searchgrade
Grade to search: F
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
#
```

(7) 특정 학생 삭제

remove 명령 입력 시, 삭제하고자 하는 학생의 학번을 입력받아 삭제 후 Student removed.라 출력한다. 해당 학생이 목록에 없을 경우 NO SUCH PERSON이라 출력한다. 목록에 아무도 없을 경우 List is empty라 출력한다.

```
# remove
Student ID: 20180030
NO SUCH PERSON.
# remove
Student ID: 20180011
Student removed.
# show
Student               Name  Midterm     Final     Average     Grade
-------------------------------------------------------------------
20180021        Jo Hyunseo    100        100       100.0         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
20180007       Kim Cheolsu     75         62        68.5         D
```

```
# remove
Student ID: 20180021
Student removed.
# remove
Student ID: 20180002
Student removed.
# remove
Student ID: 20180009
Student removed.
# remove
Student ID: 20180001
Student removed.
# remove
Student ID: 20180007
Student removed.
# remove
List is empty.
#
```

(8) 프로그램 종료

quit 명령 입력 시, 프로그램을 종료한다. 해당 명령어를 실행할 경우 현재까지 편집한 내용의 저장 여부를 묻고 저장을 선택할 경우 파일명을 입력 받아 저장한다. 저장할 때의 목록의 순서는 평균을 기준으로 내림차순으로 한다.

```
# quit
Save Data?[yes/no] yes
File name: newStudents.txt
(base) piai@piai-Precision-7920-Tower:~/바탕화면/homework$
```

열기(O) | newStudent.txt ~/바탕화면/homework | 저장(S)

```
20180021        Lee Hyori       93      95
20180002        Lee Jieun       92      89
20180009      Lee Yeonghee      81      84
20180001      Hong Gildong      84      73
20180006       Lee Sangsun      77      66
20180007       Kim Cheolsu      75      62
```

4. 추가된 함수

- 학점 계산 함수 : 평균을 입력받아 학점을 계산하여 반환하는 함수이다. 학점을 계산하는 명령은 데이터 로드 및 딕셔너리 저장, 학점 수정 함수, 학생 추가 함수 총 3군데에서 쓰이므로 학점 계산을 하는 get_grade 함수를 따로 구현하였다.

- 출력 함수 : 학생 정보를 출력하는 함수는 한 명의 학생 출력, 전체 학생 출력, 상단 메뉴 출력 함수 총 3개의 함수로 나누어서 구현하였다. search, changescore, searchgrade 명령은 실행 이후 해당 학생이 한 번 출력되므로 한 명의 학생을 출력하는 show_stu 함수로 빼내어 구현하였다. 또한 show 명령으로는 전체 학생을 출력하기 때문에 전체 함수를 출력하는 show_all_stu 함수로 나누어 구현하였다. 딕셔너리를 가져와 화면에 출력할 때 찾은 학생이 여럿일 경우 상단의 메뉴가 여러 번 출력되는 이슈가 있었다. 찾은 학생을 출력할 때 상단의 메뉴도 매번 같이 출력이 되도록 구현했기 때문이었다. 따라서 이 부분을 show_table_header라는 이름의 함수로 빼내었다. 그리고 학생 정보를 출력해야 할 경우 상단 메뉴를 한번 출력한 다음 아래에 계속해서 출력되도록 수정하여 해결하였다.

5. 토론

- 주어진 데이터는 학생들의 정보가 주어지고 학생들의 ID인 학번이 있기 때문에 학번으로 구성된 딕셔너리를 만드는 것이 리스트로 만드는 것보다 검색하기도 편리하고 프로그램 가독성이 더 좋다고 판단하였다. 하지만 구현을 마치고 보니 이 방법이 리스트보다 “확실히” 더 나은지는 잘 모르겠다. 리스트를 썼어도 프로그램 가독성 측면에서는 무리가 없었으리라 생각한다.

6. 결론

이번 과제에서 나의 목표는 어느 누가 보아도 이 프로그램의 목적이 명료하고 누구나 쓰기 좋은 코드로 만들겠다는 것이었다. 그 과정에서 크게 2가지의 고민이 있었다. 첫째는 이 프로그램에 변경이 생겼을 때 (가령 새로운 기능이 추가된다거나, 기존의 기능을 수정해야 한다거나) 이 코드가 수정, 변경이 용이한 코드인가에 대한 고민이었다. 둘째는 함수를 어디까지 세분화해서 나눠야 할지에 대한 고민이었다. 이 코드는 최대한 각 함수의 기능이 독립적이며 다른 코드에 크게 영향을 미치지 않게 하려고 노력하였다. 하지만 어디까지가 그 함수의 역할인지, 몇 회의 반복이 이루어지면 이를 함수로 빼내어 구현해야 할 지에 대한 확신이 부족했다. 이 프로그램에서는 일련의 명령이 3회 이상 반복된다 싶으면 함수로 빼내어 구현하였다. 이 밖에도 함수명과 변수명을 선정하는데 많은 시간과 고민을 할애하였다. 함수명, 변수명만 보고도 명료하게 이해가 되는 이름을 고르기 위해 노력하였다. 결론적으로 지금 이 코드가 가장 좋은 코드인지는 잘 모르겠으나 어떤 코드가 좋은 코드 인가에 대한 고민에 빠지게 되었고 그 부분을 최대한 녹여내어 코드를 여러 번 수정했고 최적화를 하였다.