파이썬 프로젝트 30기

이름 : 김도형

소속 : C반 4조

e-mail : ehguddlok@naver.com

Problem 성적 관리 프로그램

1. 문제의 개요

본 프로그램에 대해서 간략히 설명하자면 아래와 같다.

- 파일로부터 데이터를 읽어서 성적 목록을 만들어 관리하는 성적 관리프로그램이다.
- 성적 목록.txt 파일을 읽어온다.
- 7개의 명령어(show, search, changescore, searchgrade, add, remove, quit)를 입력 받아 각 기능을 수행한다.

각 기능을 간단히 설명하자면 아래와 같다.

- show : 저장되어 있는 전체 목록을 평균 점수를 기준으로 내림차순으로 출력한다.
- search : 검색하고자 하는 학생의 학번을 받아 출력한다.
- changescore : 목록에 있는 학생 1명에 대해 중간/기말의 점수를 수정한다.
- add : 학생의 학번, 중간고사, 기말고사 점수를 입력 받아 목록에 추가한다.
- searchgrade : 특정 grade를 입력 받아 출력한다.
- remove : 학생의 학번을 입력 받아 목록에 있는 경우 삭제한다.
- quit : 편집한 내용의 저장 여부를 입력하고 프로그램을 종료한다.

이때 학생의 학점은 아래의 기준으로 정한다.

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. 문제 해결 방법

문제 해결을 위한 **main함수**의 pseudo code는 아래와 같다.

```
- main() 함수 : 프로그램의 주요 로직 관리
        - txt 파일을 읽는다. (deafault는 students.txt이다.)
        - 평균점수, 학점을 계산한다.
        - dict(stu_dict)와 list(sorted_sl)에 평균점수, 학점을 추가한 후 평균으로 내림차순 정렬 (이때
        cal_grade()함수를 사용하여 학점 계산)
        - show() 함수 : 학생 목록 출력
        - 입력 반복 루프
                - 명령어 입력받기
                - 'show'면 show() 함수 : 모든 학생 성적 목록 출력
                - 'search'면 search() 함수 : 해당 학생 성적 목록 출력
                - 'changescore'면 changescore() 함수 : 학생의 점수를 수정
                - 'add'면 add() 함수 : 새로운 학생 성적 정보 추가
                - 'searchgrade'면 searchgrade() 함수 : 해당 학점 학생 성적출력
                - 'remove'면 학생 목록 비었는지 확인 후 remove() 함수 : 학생 성적 정보 제거
                - 'quit'면 파일 저장 유무 물어보고 save_file() 함수 : 편집된 학생 성적 정보 저장
                - 이외의 명령어 입력하면 사용가능한 명령어들 출력
```

7가지의 주요 함수에 대한 pseudo code는 아래와 같다.

```
show() :
        -   출력 형식을 맞추어 평균 점수로 정렬된 sorted_ls를 출력한다.
search () :
        -   학생의 학번을 입력 받는다.
        -   check_student() : stu_dict를 사용하여 학생의 유무를 확인한다.
            - 해당 학번의 학생이 없으면 'NO SUCH PERSON' 출력 후 return
        -   찾은 학생을 show()를 통해 출력한다.
changescore() :
        -   학생의 학번을 입력 받는다.
            - check_student() : stu_dict를 사용하여 학생의 유무를 확인한다.
                - 해당 학번의 학생이 없으면 'NO SUCH PERSON' 출력 후 return
        -   수정하고자 하는 점수가 중간고사인지 기말고사인지 입력받는다.
                - 입력받은 값이 mid/final이 아니면 return
        - 입력할 학생의 점수를 입력받는다.
                - 점수가 0~100 사이가 아니라면 return
        - 점수의 평균과 학점을 구한다.
        - changed_show() 함수 : 바뀐 학생의 정보를 형식에 맞추어 출력
add() :
        -   학생의 학번을 입력 받는다.
            - 학생의 학번이 이미 존재하면 'ALREADT EXISTS' 출력
        -   학생의 학번이 8자리가 아니라면 함수 'Student ID must be 9 digits.' 출력 후
            return (학번은 8자리여야 한다고 판단함.)
```

- *학생의 이름, 중간고사 점수, 기말고사 점수를 입력받는다.*
- *평균과 학점을 구한 후 stu_dict에 추가하고, sorted_sl을 평균 내림차순으로 정렬한다.*
- *'Student added' 출력 후 return*

searchgrade() :

- *특정 학점을 입력 받는다.*
- *A/B/C/D/F 외의 값이 입력되면 return*
- *sorted_ls를 순회하며 해당 학점의 학생들을 찾는다.*
  - 해당 학생이 없으면 'NO RESULTS' 출력
- *찾은 학생을 show()를 통해 출력한다.*

remove() :

- *학생의 학번을 입력 받는다.*
  - stu_dict를 사용하여 학생의 유무를 확인한다.
  - 해당 학번의 학생이 없으면 'NO SUCH PERSON' 출력 후 return
- *stu_dict와 sorted_sl에서도 제거해준다.*
- *'Student removed' 출력 후 return*

save_file() :

- *저장하고 싶은 파일의 이름을 입력 받는다.*
- *정렬된 sorted_sl을 순회하며 [Studentnumber][₩t][Name][₩t][Midterm][₩t][Final][₩n] 형식으로 파일 저장.*

3. 예제 : 프로그램 실행결과

각 기능에 대한 프로그램의 실행결과 간단하게 설명하면 아래와 같다.

1. Terminal에서 project.py 파일 실행하기 -> main() 함수 실행하기

```
PS C:\Users\ehgud\Desktop\포빅아\python_project> 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

# Type command :
```

Terminal에서 'python project.py'만 입력해도 default로 'student.txt'가 열리고 show()가 실행되어 학생 성적목록이 출력된다.

```
PS C:\Users\ehgud\Desktop\포빅아\python_project> python .\project.py students.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

# Type command : find
Available commands: show, search, changescore, searchgrade, add, remove, quit

# Type command :
```

'python.py student.txt'를 입력하면 'student.txt'파일이 열리고 show()가 실행되어 학생 성적목록이 출력된다.

또한 7개의 명령어 이외의 잘못된 명령어 입력 시, 에러 메시지 없이 다시 명령어를 입력 받을 준비를 한다. 또한 **명령어를 입력할 때 'Type command'를 출력**하여 사용자들이 명령어를 입력하도록 하였으며, **7가지 이외의 명령어를 입력하면 사용가능한 명령어들을 출력**한다.

2. show를 입력하여 show()함수 호출하기

```
# Type command : 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

# Type command :
```

show() 함수를 호출하여 학생들의 학번, 이름, 중간고사 점수, 기말고사 점수, 평균, 학점이 평균을 기준으로 내림차순으로 출력하였다. 이때 **중간/기말 성적은 정수라고 가정하고** 프로그램을 작성하였다. 수업시간에 질문을 한 결과 정수라고 가정하라고 하셔서 그렇게 진행하였다.

3. search를 입력하여 search()함수 호출하기

```
# Type command : search
Student ID: 20180005
NO SUCH PERSON.

# Type command : search
Student ID: 20180002
   Student         Name   Midterm   Final   Average   Grade
----------------------------------------------------------------
  20180002    Lee Jieun      92       89      90.5      A

# Type command :
```

학번을 입력 받을 후 해당 학번이 학생 목록에 없는 경우 'NO SUCH PERSON'을 출력하고, 있는 학생이라면 학번, 이름, 중간고사 점수, 기말고사 점수, 평균, 학점을 출력한다.

4. changescore를 입력하여 changescore() 함수 호출하기

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

# Type command : changescore
Student ID: 20180007
Mid/Final? miid

# Type command : changescore
Student ID: 20180007
Mid/Final? mid
Input new score: 147

# Type command :
```

학번을 입력 받는데, 목록에 없는 경우 'No SUCH PERSON'을 출력한다. 이 후 올바른 학번을 입력하고 mid/final 이외의 값을 입력하면 아무것도 실행되지 않는다. 점수의 경우에는 0~100 이외의 값을 입력하면 아무것도 실행되지 않는다.

```
# Type command : changescore
Student ID: 20180007
Mid/Final? mid
Input new score: 75
   Student           Name   Midterm   Final   Average   (
-------------------------------------------------------------
  20180007    Kim Cheolsu      57       62      59.5
Score changed.
  20180007    Kim Cheolsu      75       62      68.5

# Type command : show
   Student            Name   Midterm   Final   Average
-------------------------------------------------------------
  20180002       Lee Jieun      92       89      90.5
  20180009    Lee Yeonghee      81       84      82.5
  20180001    Hong Gildong      84       73      78.5
  20180007     Kim Cheolsu      75       62      68.5
  20180011      Ha Donghun      58       68      63.0

# Type command :
```

위와 같이 올바른 값들이 입력되면 학생을 추가한다. 그리고 changed_show()를 통해서 형식에 맞게 바뀐 학생의 정보 전/후를 출력한다. 이 후 show()를 통해 전체 학생의 성적 목록을 보면 점수가 바뀐 것을 확인할 수 있다.

5. add를 입력하여 add() 함수 호출하기

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

# Type command : add
Student ID: 2018000
Student ID must be 8 digits.

# Type command :
```

학번을 입력 받는데 학번이 이미 학생 목록에 존재하면 'ALREADY EXIST'를 출력한다. 또한 **학번을 추가할 때 8자리가 아니면 'Student ID must be 8 digits'를 출력**한다. 학번은 보통 8자리이고 학번의 자릿수가 통일되어야 한다고 생각했기 때문에 이 기능을 추가하였다.

```
# Type command : add
Student ID: 20180021
Name: Lee Hyori
Midterm Score: 83
Final Score: 95
Student added.

# Type command : add
Student ID: 20180006
Name: Lee Sansun
Midterm Score: 77
Final Score: 66
Student added.

# Type command : add
Student ID: 20180099
Name: Kim Dohyeong
Midterm Score: 95
Final Score: 90
Student added.
```

올바르게 형식에 맞게 총 3명의 학생을 추가하였다. 학생이 추가되면 'Student added'가 출력된다. **나의 정보도 추가하였다. 이때 학번은 '20180099', 이름은 'Kim Dohyeong', 성적은 중간고사, 기말고사 각각 95, 90점으로 입력**하였다.

```
# Type command : show
   Student            Name   Midterm   Final   Average
-------------------------------------------------------
  20180099    Kim Dohyeong      95       90      92.5
  20180002       Lee Jieun      92       89      90.5
  20180021       Lee Hyori      83       95      89.0
  20180009    Lee Yeonghee      81       84      82.5
  20180001    Hong Gildong      84       73      78.5
  20180006      Lee Sansun      77       66      71.5
  20180007     Kim Cheolsu      75       62      68.5
  20180011      Ha Donghun      58       68      63.0

# Type command :
```

이 후 show()를 통해 학생들의 성적 목록을 확인하니 평균을 중심으로 내림차순 정렬이 되었으며 3명이 정상적으로 추가되었다는 것을 확인할 수 있다.

6. searchgrade를 입력하여 searchgrade() 함수 호출하기

```
# Type command : searchgrade
Grade to search: E

# Type command : searchgrade
Grade to search: F
NO RESULTS.

# Type command : 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

# Type command :
```

특정 학점을 입력하는데 A/B/C/D/F 외의 값이 입력되면 아무것도 실행되지 않는다. 또한 해당 학점에 해당하는 학생이 없으면 'NO RESULTS'가 출력된다. 예를 들어 D를 입력한 경우 해당하는 학생들이 show()를 통해서 출력된다.

7. remove를 입력하여 remove() 함수 호출하기

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

# Type command : remove
Student ID: 20180011
Student removed.

# Type command : show
   Student            Name   Midterm   Final   Average   Grade
-----------------------------------------------------------------
  20180099    Kim Dohyeong      95       90      92.5      A
  20180002       Lee Jieun      92       89      90.5      A
  20180021       Lee Hyori      83       95      89.0      B
  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

# Type command :
```

학번을 입력 받는데 해당 학생이 목록에 없으면 'NO SUCH PERSON.'이 출력된다. 목록에 있는 학생의 학번을 입력하면 'Student removed'가 출력되고 show()를 통해 확인하면 적절하게 삭제된 것을 확인할 수

있다.

```
PS C:\Users\ehgud\Desktop\포빅아\python_project> python .\project.py students.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

# Type command : remove
Student ID: 20180002
Student removed.

# Type command : remove
Student ID: 20180009
Student removed.

# Type command : remove
Student ID: 20180001
Student removed.

# Type command : remove
Student ID: 20180011
Student removed.

# Type command : remove
Student ID: 20180007
```

또한 이는 목록이 비었을 때 remove를 입력하면 어떻게 되는지 설명하기 위해 잠시 'students.txt'를 다시 불러와서 5명 모두 삭제하였다.

```
# Type command : remove
List is empty.

# Type command :
```

학생들의 목록이 빈 상태에서 remove를 입력하면 'List is empty.'가 출력되는 것을 확인할 수 있다.

8. quit를 입력하여 save_file() 함수 호출하기(yes일 때)

```
# Type command : quit
Save data?[yes/no] yes
File name: newStudents.txt
PS C:\Users\ehgud\Desktop\포빅아\python_project> python .\project.py newStudents.txt
   Student            Name   Midterm   Final   Average   Grade
-----------------------------------------------------------------
  20180099     Kim Dohyeong      95       90      92.5      A
  20180002        Lee Jieun      92       89      90.5      A
  20180021        Lee Hyori      83       95      89.0      B
  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

# Type command :
```

quit를 입력하면 'Save data?[yes/no]'가 출력된다. 이때 yes를 입력하면

'File name'이 출력되는데, 이때까지 편집한 내용을 저장하고 싶은 파일 이름을 입력하면 된다. 'newStudents.txt'로 입력한 후 프로그램이 정상적으로 종료되었다. 이후 파일을 확인해보면 잘 저장된 것을 확인할 수 있다.

```
# Type command : quit
Save data?[yes/no] no
PS C:\Users\ehgud\Desktop\포빅아\python_project>
```

quit을 입력한 후 'Save data?[yes/no]'가 출력될 때 no를 입력하면 편집한 내용이 저장되지 않고 프로그램을 종료하게 된다.

4. 토론 : 결정사항 및 문제

위에서도 중간중간 설명하였지만 프로그램 개발 과정에서 맞닥뜨린 결정사항은 아래와 같다.

결정사항 :

1. 7개의 명령어 이외의 명령어 입력하면 사용가능한 명령어들 출력

   해당 기능은 이 프로그램을 사용 방법에 대해 잘 모르는 사람이 사용했을 때를 고려하여 추가하였다. 이 프로그램을 개발하고 이는 본인은 7개의 핵심 명령어(show, search, changescore, searchgrade, add, remove, quit)에 대해서 잘 알고 있지만 처음 사용하는 사람들은 어떤 명령어가 있는지 모를 수도 있다. 그렇기에 다른 명령어를 입력했을 때 기능을 지원하는 7개의 명령어를 출력하도록 하였다.

2. 명령어를 입력할 때 'Type command'를 출력

   이 또한 처음 이용하는 사람들을 위한 기능이다. 단순히 입력창에 '#'만 있으면 어떤 행동을 취해야 하는지 알기 어렵다. 그렇지만 '# Type command'를 출력하면서 사용자가 '명령어를 입력하는 거구나!'라고 생각하고 행동을 취할 수 있다.

3. 학생들의 중간/기말 성적은 정수라고 가정

   해당 결정사항을 통해서 보다 간단한 프로그램으르 구현하였다. 만약 성적이 실수로도 입력이 된다면 내부적으로 더 처리를 거쳐야 한다. 그래서 수업시간 질문을 통해서 성적은 정수로 가정해도 좋다고 판단하여 프로그램을 개발하였다.

4. 학생의 학번이 8자리가 아니라면 함수 'Student ID must be 9 digits.' 출력

   이는 학생을 추가하는 명령어인 add를 입력한 후 add()함수에서 구현된 내용이다. 이 결정사항은 학생들의 성적 데이터들의 통일성을 위해 추가하였다. 만약 누군가가 실수로 학생의 학번은 8자리가 아닌 수를 입력하였다면 다시 remove를 하고 add를 해야하는 번거로움이 있다. 이를 사전에 방지하고자 학생의 학번은 8자리로 추가될 수 있게 프로그램을 구현하였다.

문제사항 및 해결 :

1. changesocre() 함수 구현 시 변경 전/후 관리

   changescore() 함수에서 학생의 학번과 시험의 종류와 성적을 입력하면 학생의 바뀐 시험 정보가 모두 출력된다. 이때 sorted_ls가 list 타입이라 즉 mutable 변수이기 때문에 출력되는 부분을 확인하면 모두 변경 후의 데이터가 출력되는 문제가 있었다. 이를 해결하기 위해 list()를 이용해서 새로운 list를 만들어 다른 값을 참조하게 하여 이를 해결했다.

2. main() 함수 구현을 위한 변수 관리

   이때까지 프로그램을 구현할 때는 main함수를 사용하지 않았다. 그러나 이 프로그램은 Terminal에서 작동시킨다는 점을 고려하여 main() 함수로 구현하고자 하였다. 그러나 기존에 전역변수로 사용하고 있는 sorted_ls와 stu_dict는 사용하는 모든 함수에서 접근이 가능했지만, main() 함수를 사용하려고 하면 이를 지역변수로 정의되기 때문에 다른 함수에서 상용하는데 무리가 있었다. 그래서 'global'을 통해서 함수 내에서 전역변수로 정의할 수도 있지만 이는 관리하기 어렵다는 단점이 있다. 그래서 모든 함수마다 sorted_ls와 stu_dict를 입력값으로 추가하여 사용할 때마다 함수의 파라미터로 넣어주어 이를 해결하였다.

5. 결론

이번 숙제를 통하여 배운 점은 아래와 같다.

1. Pseudo code의 중요성

   프로그램을 개발하다 보면 많은 요구사항들을 구현해야 할 때가 있다. 이때 pseudo code 없이 바로 코드를 작성하려고 하면 이전에 구현했던 내용들을 자주 수정하면서 프로그램을 개발하고 비효율적인 작업과 헷갈려서 잘못된 프로그램이 개발될 수 있다. 그러나 이번 숙제에서는 psedo code를 미리 작성하면서 보다 효율적으로 작업을 할 수 있었고, 요구사항들을 구현할 때 이전 코드들을 수정하는 일을 최소화할 수 있었다.

2. 최소 기능 단위로 함수 나누기

   프로그램에서는 함수가 중요한 역할을 한다는 것을 깨달았다. 요구사항들을 구현하다 보면 반복해서 사용해야 할 기능들이 많다. 그러나 이를 함수화하면서 반복적인 작업을 줄일 수 있었고, 코드에 에러가 발생했을 때 해당 함수만 수정하면 되기 때문에 효율적인 개발이 가능하다.