**파이썬 1주차 과제**

**성적 관리 프로그램**

**청년 AI∙Big Data 아카데미 13기**

**담당교수: 윤은영**

**소속: A1조**

**이름: 최준호**

|  |
| --- |
| **명예서약(Honor code)**  **“나는 이 프로그래밍 과제를 다른 사람의 부적절한 도움 없이 완수하였습니다.”** |

1. 문제의 개요

파일로부터 데이터를 읽어서 성적 목록을 관리하는 성적 관리 프로그램을 작성한다.

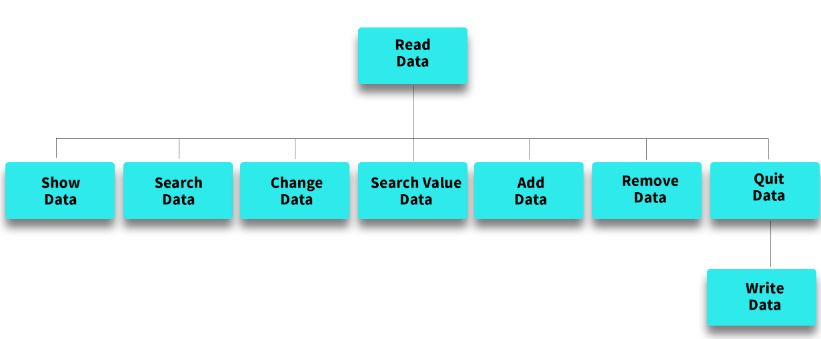
본 프로그램을 간략히 설명하면 다음과 같다.

* 이 프로그램은 사용자로부터 7개의 명령어(show, search, changescore, searchgrade, add,

remove, quit)를 입력 받아 각 기능을 수행하게 된다. 최소한 각 명령어 별로 함수를

정의하여 사용한다. 명령어 외에 필요한 함수는 추가로 정의하여 사용할 수 있다.

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

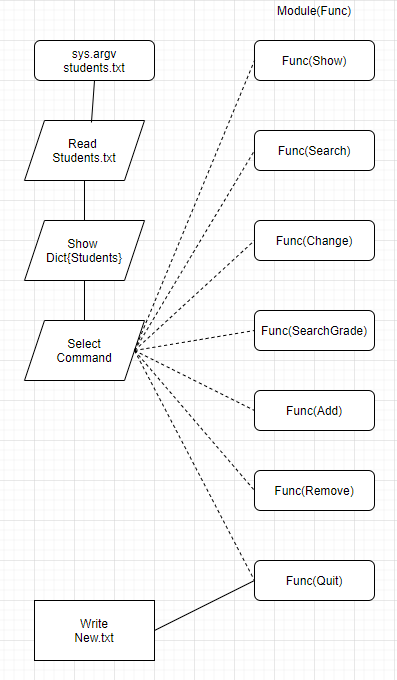


* 입력부: Student.txt 텍스트 파일을 읽어 들인다.
* 처리부: 7개의 기능을 가진 함수를 구현해 명령어 입력시 해당 함수를 실행 한다.
* 출력부: 계산이 완료된 거리 결과를 실수형태로 출력한다.

출력부: 자신의 이름을 ASCII 코드를 통해 출력한다.

2. 알고리즘

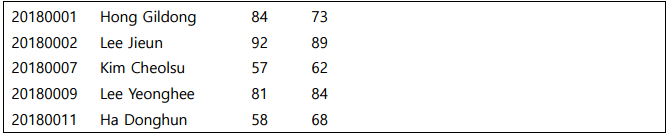
본 프로그램의 알고리즘을 Flowchart를 통해 표현하면 아래와 같다.



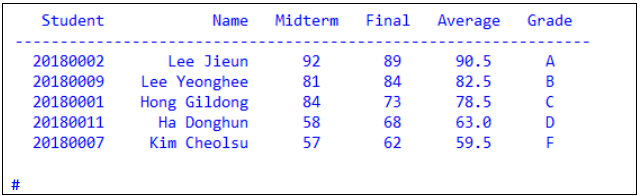
3. 프로그램 구조 및 설명

a) 프로젝트 메인 함수의 동작

* 프로그램을 실행은 리눅스의 경우 $ python project1.py students.txt를 통해 진행, 윈도우의 경우 C:\> python project1.py students.txt를 통해 이루어진다.
* 프로그램 실행 시 텍스트 파일로부터 학생들의 성적 목록 작성을 위한 데이터를 읽으며,텍스트 파일의 내용 및 구성은 아래와 같다.



* 프로그램을 실행시키면 텍스트 파일로부터 데이터를 읽어 목록을 딕셔너리(dict) 자료형을 사용하여 저장하고, 전체 목록을 평균 점수를 기준으로 내림차순으로 정렬하여 아래의 예제처럼 출력한다. 동일한 평균 점수를 가진 학생들이 있는 경우 순서는 상관없다.

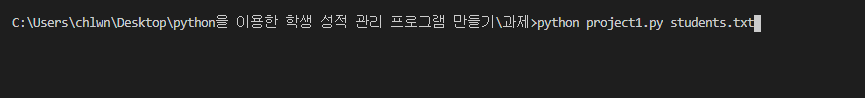


* 출력시 각 데이터간의 길이 차이로 인해 깔끔한 외형을 보여주지 않을 수 있기 때문에 Strformat을 이용하여 %-10s를 각 데이터 사이에 넣어주어 깔끔하게 정리하였다.
* 사용자는 7개의 명령어(show, search, changescore, searchgrade, add, remove, quit)를 사용할 수 있으며, 명령어를 입력하였을 때만 기능이 실행된다. 이 명령어는 사용자가 명령어 입력 시, 대소문자를 구분하지 않고 동일한 명령어의 기능을 수행하도록 입력값의 lower함수를 적용하였다.
* 명령어는 func.py를 모듈화하여 동작하였고 이를 위하여 import func를 통해 모듈을 호출한다.
* 7개의 명령어 이외의 잘못된 명령어 입력 시, 에러 메시지 없이 명령어를 입력 받을 준비를 한다.

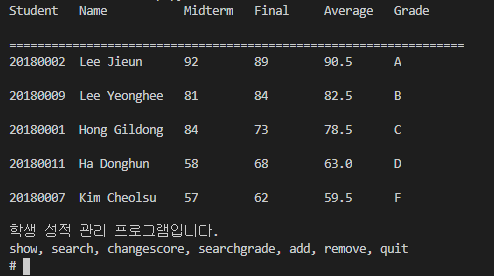
b) func 모듈

* 명령어를 모듈화하여 func.py란 파일에 모두 저장하였고 project1.py에서 import func를 통해 모듈을 호출한다.
* show (전체 학생 정보 출력)
  + show 입력 시, 저장되어 있는 전체 목록을 아래와 같이 평균 점수를 기준으로 내림차순으로 출력한다. 평균 점수는 소수점 이하 첫째 자리까지만 표시하기 위해 round 함수를 이용하였다.
* search (특정 학생 검색)
  + search 입력 시, 아래와 같이 검색하고자 하는 학생의 학번을 요구해 입력 받아 학번, 이름, 중간고사 점수, 기말고사 점수, 평균, 학점을 출력한다.
* changescore (점수 수정)
  + 목록에 저장된 학생 중 1명의 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 Grade도 다시 계산하여 수정한다
  + 이때 mid, final도 대소문자 구문없이 입력하기 위해 lower함수를 적용한다.
* add (학생 추가)
  + add 입력 시, 아래와 같이 학생의 학번, 이름, 중간고사 점수, 기말고사 점수를 차례로 요구해 입력 받는다. 추가되면, 메시지 “Student added.”를 아래 예제와 같이 출력한다.
  + Average와 Grade는 중간고사 점수와 기말고사 점수를 사용하여 계산하여 저장한다.
  + 학생 추가 후 show 명령어를 사용하면 평균을 기준으로 내림차순으로 출력된다.
* searchgrade (Grade 검색)
  + searchgrade입력 시, 특정 grade를 입력 받아 그 grade에 해당하는 학생을 모두 출력한다.
* REMOVE (특정 학생 삭제)
  + remove 입력 시, 아래와 같이 삭제하고자 하는 학생의 학번을 입력 받은 후, 학생이 목록에 있는 경우 삭제한다. 삭제하면, 메시지 “Student removed.”를 아래와 같이 출력한다. 이때 Dict형식의 데이터를 삭제하기 위해 del 함수를 사용한다.
* quit (종료)
  + quit 입력 시, 프로그램을 종료한다.
  + 해당 명령어를 실행할 경우, 현재까지 편집할 내용의 저장 여부를 묻고, 저장을 선택할 경우 파일명을 입력 받아서 저장하도록 한다. 앞서 본 “students.txt”와 같이 내용을 구성한다.

4. 프로그램 실행방법 및 예제

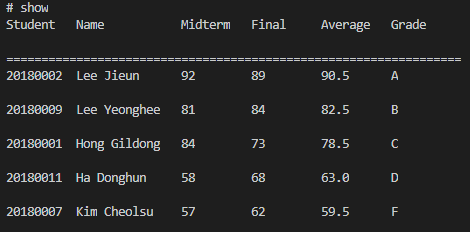


• Python터미널 환경에서 위 명령어를 통해 project1.py를 실행시키고 students.txt파일을 불러들인다.

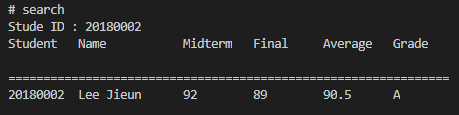


• students.txt파일의 정보가 불러들여지고 Average와 Grade 정보가 추가된 채로 평균의 내림차순으로 정렬된 데이터가 보여진다.

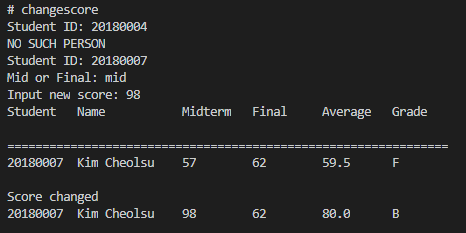
• “학생 성적 관리 프로그램입니다”라는 설명이 나오고 그 밑에는 명령어의 종류와 명령어를 입력하는 곳에 #으로 표시되어져있다.



• show 명령어를 입력하게 되면 처음 파일을 실행했을때와 마찬가지로 데이터들이 평균 내림차순으로 보여지게 된다.

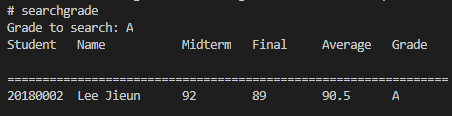


• search 명령어를 입력하게 되면 Student ID를 입력하게 되고 데이터내에 존재하면 다음과 같이한 줄 출력한다.



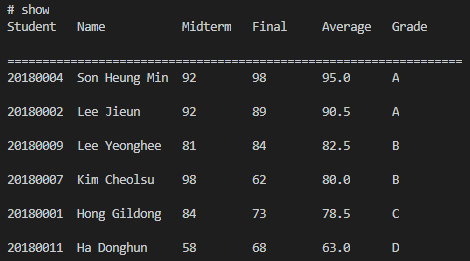
• changescore 명령어를 입력하게 되면 Student ID를 입력시 존재하지 않는 ID일 경우 “NO SUCH PERSON”이라는 문구를 출력하게 되며 존재하는 ID를 입력하게 되면 어떤 시험에 대한 점수를 수정할 것인지에 대한 Mid or Final이라는 질문과 새로운 점수를 입력하게 된다.

• 입력 후에는 수정전 데이터와 수정후의 데이터가 보여진다.



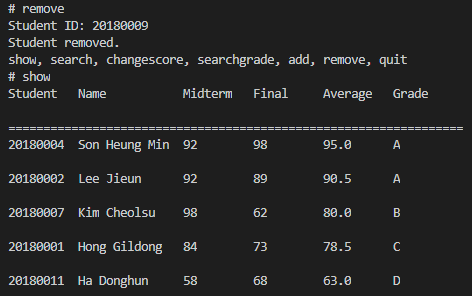
• searchgrade 명령어를 입력하게 되면 검색하고자 하는 Grade를 입력하고 이를 출력한다.





• add 명령어를 입력하게 되면 입력하고자 하는 아이디, 이름, 중간고사점수, 기말고사 점수가 나오면 입력을 마친 후에는 “Student ADDED”라는 문구가 나온다.

• show 명령어를 통해 확인하게 되면 추가 된 것을 확인할 수 있다.



• remove 명령어를 입력후 지우고자 하는 학생 아이디를 입력하면 “Student removed”라는 문구와 함께 삭제되며 show명령어를 통해 삭제를 확인할 수 있다.



• quit 명령어를 입력하게 되면 데이터를 저장여부를 묻는 질문과 승낙 하게 되면 File name 이름을 입력하게 되면 txt파일로 저장하게 된다. 이때 파일은 같은 디렉토리내로 저장된다.

5. 토론

* 데이터를 출력할 때 데이터가 정렬되어 출력되지 않아 해결방법을 찾아야 했다.
* 출력시에는 strFormat = '%-10s%-15s%-10s%-10s%-10s%-10s\n' 정렬을 하였고 모든 출력이 필요한 명령어에 이를 적용하였다.

6. 결론

* 본 과제에서는 파이썬을 활용하여 외부 파일을 읽고 모듈을 이용해 함수를 적용후 파일을 쓰는 방법을 익힐 수 있었다.

7. 개선방향

* 본 과제는 설명되어진 기능을 통해서 모듈기능을 활용하여 작성하였다. 하지만 함수를 작성할 시 중복되는 내용이 많아 이를 클래스를 활용한다면 더 직관적이고 적은 코드량으로 프로그램을 완수할 수 있었다고 생각한다.