

# CSED101. Programming & Problem solving Spring, 2020

## Programming Assignment #4 (75 points)

차동민 (cardongmin@postech.ac.kr)

■ Due: **2020.07.20 23:59**

■ Development Environment: Windows Visual Studio 2019

### ■ 제출물

- C Code files (**assn4.c**)
  - 프로그램의 소스 코드를 이해하기 쉽도록 반드시 주석을 붙일 것.
- 보고서 파일 (assn4.docx, assn4.hwp 또는 assn4.pdf)
  - AssnReadMe.pdf 를 참조하여 작성할 것.
  - **명예서약(Honor code)**: 표지에 다음의 내용을 포함한다. “나는 이 프로그래밍 과제를 다른 사람의 부적절한 도움 없이 완수하였습니다.” 보고서 표지에 명예서약이 없는 경우는 과제를 제출하지 않은 것으로 처리한다.
  - 소스코드와 보고서 파일을 LMS를 이용하여 제출한다.

### ■ 주의사항

- 각 문제에 해당하는 요구사항을 반드시 지킬 것.
- 모든 문제의 출력 형식은 채점을 위해 실행 예시와 최대한 비슷하게 작성해 주세요.
- 문제에 제시되어 있는 파일이름으로 제출 할 것. 그 외의 다른 이름으로 제출하면 감점 또는 0점 처리된다.
- 컴파일 & 실행이 안되면 무조건 0점 처리된다.
- 하루 late시 20%가 감점되며, 3일 이상 지나면 받지 않는다. (0점 처리)
- 부정행위에 관한 규정은 POSTECH 전자컴퓨터공학부 학부위원회의 ‘POSTECH 전자컴퓨터공학부 부정행위 정의’를 따른다. (LMS의 과목 공지사항의 제목 [document about cheating]의 첨부파일인 disciplinary.pdf를 참조할 것.)
- 이번 과제에서는 추가 기능 구현에 대한 추가 점수는 없습니다.

## ■ Problem: 성적 관리 프로그램

### [목적]

이번 과제를 통하여 동적 할당을 응용한 Structure(구조체)와 Linked List(연결 리스트)의 사용법을 익힌다.

### [문제]

파일로부터 데이터를 읽어서 성적 목록을 만들어 관리하는 성적 관리 프로그램을 작성한다.

### [주의사항]

- 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. 즉, 명령어 외에 필요한 함수는 정의하여 사용할 수 있다.
- 이번 과제는 구조체와 연결리스트를 활용하는 것이 목표이므로, 문제에 구조체와 연결리스트를 언급한 부분을 배열을 통해서 해결할 경우 감점 처리한다.
- 명시한 에러 처리 외에는 고려하지 않아도 된다.
- string.h 에서 제공하는 라이브러리 함수를 사용할 수 있다.
- 전역 변수, goto 문을 사용할 수 없다.

### [설명 및 요구사항]

- 프로그램 구현은 구조체와 linked list를 사용하여야 하며, 각 항목에 대한 제약은 다음과 같다.
    - 학생 목록 리스트: 학번, 이름, 중간점수, 기말점수 등의 정보를 가진 학생에 대한 구조체를 정의하고 링크드 리스트로 구현한다.
    - 정렬에 대한 함수는 있어도 좋으나, 이번 과제에서는 학생 정보 추가 시에만 (add할 때, 파일에서 읽어 올 때) 정렬이 되도록 하는 것으로 충분하고 또 권장한다.
  - 학번: 학번은 8자리 숫자로만 구성되며, 동일한 학번을 가진 경우는 없다.
  - 이름: 영문자로 공백포함해서 최대 30자
  - 중간, 기말고사 점수: 0~100까지의 값을 가지는 integer
- 프로그램을 실행하면, 아래와 같이 성적처리 할 파일 이름을 입력 받는다. 파일이름에는 공백이 없으며 확장자를 포함하여 30자를 넘지 않는다고 가정한다.

```
Input the score file name:
```

- 예외처리:  
존재하지 않는 파일명을 입력한 경우에는 아래와 같은 에러 메시지를 출력 후, 프로그램을 종료한다. (실행 예시의 밑줄은 사용자 입력에 해당함)

```
Input the score file name: wrong.txt  
Could not open wrong.txt file
```

```
계속하려면 아무 키나 누르십시오. . .
```

- 프로그램 실행 시 텍스트 파일로부터 학생들의 성적 목록 작성을 위한 데이터를 읽으며, 텍스트 파일의 내용 및 구성은 아래와 같다.

20210001	Hong Gildong	84	73
20210002	Lee Jieun	92	89
20210007	Kim Cheolsu	57	62
20210009	Lee Yeonghee	81	84
20210011	Ha Donghun	58	68

- 각 줄은 각 학생의 학번, 이름, 중간고사 점수, 기말고사 점수로 구성되어 있으며 각 항목 사이는 `tab(\t)`으로 구분된다.
- 학생과 학생 사이는 줄 바꿈 문자(`\n`)으로 구분된다.  
예. `[Student number][\t][Name][\t][Midterm][\t][Final][\n]`  
- 이름(Name)은 성(Family name)과 이름(Given name)으로 구성되어 있으며, 스페이스 하나로 구분되어 있다고 가정한다.
- 프로그램을 실행시키면 텍스트 파일로부터 데이터를 읽어 목록을 linked list에 저장하고, 전체 목록을 평균(Average)을 기준으로 내림차순으로 정렬하여 아래의 예제처럼 출력한다. 동일한 평균 점수가 여러 명일 경우 순서는 상관없다.
- 성적(Grade) 항목은 중간고사 점수와 기말고사 점수의 평균을 계산하여 저장한다. Grade의 기준은 아래와 같다.

- A: 평균이 90 점 이상
- B: 평균이 80 점 이상, 90 점 미만
- C: 평균이 70 점 이상, 80 점 미만
- D: 평균이 60 점 이상, 70 점 미만
- F: 평균이 60 점 미만

- 아래와 같이 학생들의 전체 목록을 출력한 후에는 명령어 입력을 대기하는 # 표시가 뜨며, 이 상태에서 사용자는 명령어를 입력할 수 있다.

Student	Name	Midterm	Final	Average	Grade
20210002	Lee Jieun	92	89	90.5	A
20210009	Lee Yeonghee	81	84	82.5	B
20210001	Hong Gildong	84	73	78.5	C
20210011	Ha Donghun	58	68	63.0	D
20210007	Kim Cheolsu	57	62	59.5	F
#					

- 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 작성한다. 예를 들면, show, SHOW, Show, shoW 는 동일한 동작을 수행한다.  
(명령어 입력 부분은 30자를 넘는 입력에 대해서는 고려하지 않는다.)
- 7개의 명령어 이외의 잘못된 명령어 입력 시, 아래와 같이 "WRONG INPUT"이라는 에러 메시지 출력 후, 아래와 같이 다시 명령어를 입력 받을 준비를 한다. (실행 예시에 밑줄로 표시된

문자는 사용자 입력에 해당)

```
# find
WRONG INPUT

#
```

## [기능]

- 성적 관리 프로그램은 아래와 같은 기능을 가진다.
- 명시된 7가지 명령어 외의 명령어가 입력될 경우 무시하고 다시 명령어 입력을 대기한다.

### 1. show (전체 학생 정보 출력)

- **show** 입력 시, 저장되어 있는 전체 목록을 아래와 같이 평균점수를 기준으로 내림차순으로 출력한다. 평균 점수는 소수점 이하 첫째 자리 까지만 표시한다.

```
# show
Student          Name      Midterm  Final   Average  Grade
-----
20210002         Lee Jieun    92       89     90.5     A
20210009         Lee Yeonghee 81       84     82.5     B
20210001         Hong Gildong 84       73     78.5     C
20210011         Ha Donghun   58       68     63.0     D
20210007         Kim Cheolsu  57       62     59.5     F

#
```

### 2. search (특정 학생 검색)

- **search** 입력 시, 아래와 같이 찾고자 하는 학생의 학번을 요구해 입력 받은 후, 그 학생이 목록에 존재하면 찾아서 아래와 같이 출력한다.
- 예외처리:
  - ✓ 찾고자 하는 학생이 목록에 없는 경우에는 "NO SUCH PERSON" 이라는 에러 메시지를 출력

```
# search
Student ID: 20210099
NO SUCH PERSON

# search
Student ID: 20210001

Student          Name      Midterm  Final   Average  Grade
-----
20210001         Hong Gildong 84       73     78.5     C

#
```

### 3. changescore (점수 수정)

- 목록에 저장된 학생 중 1 명의 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 바뀔에 따라 Grade 도 다시 계산하여 수정한다.
- 예외처리:
  - ✓ 학번이 목록에 없는 경우에는 "NO SUCH PERSON"이라는 에러 메시지 출력
  - ✓ "mid" 또는 "final" 외의 값이 입력된 경우, WRONG INPUT 출력 후, 실행되지 않음  
mid, Mid, MID 는 모두 동일한 값으로 처리, 즉 대소문자 구분하지 않고 처리되도록 구현할 것.
  - ✓ 점수에 0~100 외의 값이 입력된 경우에는 "INVALID SCORE" 출력 후, 다시 점수를 입력 받는다.

```
# changescore
Student ID: 20210099           → 학번이 없는 경우
NO SUCH PERSON

# changescore
Student ID: 20210007
Mid/Final?: miid             → mid/final 외의 값이 입력된 경우
WRONG INPUT

# changescore
Student ID: 20210007
Mid/Final?: mid
Input new score: 175          } → 0~100 외의 값이 입력된 경우
INVALID SCORE - Please retype: 75

Student      Name      Midterm  Final  Average  Grade
-----
20210007     Kim Cheolsu      57      62      59.5     F
Score changed
20210007     Kim Cheolsu      75      62      68.5     D

# show
Student      Name      Midterm  Final  Average  Grade
-----
20210002     Lee Jieun       92      89      90.5     A
20210009     Lee Yeonghee    81      84      82.5     B
20210001     Hong Gildong    84      73      78.5     C
20210007     Kim Cheolsu     75      62      68.5     D
20210011     Ha Donghun     58      68      63.0     D

#
```

#### 4. 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.
- Average 와 Grade 는 중간고사 점수와 기말고사 점수를 사용하여 계산하여 저장한다.
- 학생 추가 후, **show** 명령어를 사용하면 평균을 기준으로 **내림차순**으로 출력된다.
- 이름은 공백 포함 영어 30 자, 점수는 숫자로만 이루어진 값이 입력된다고 가정한다.
- 예외처리:
  - ✓ 목록에 있는 학생의 학번을 입력 시, "ALREADY EXISTS" 라는 에러 메시지 출력
  - ✓ 중간고사와 기말고사에 0~100 외의 값이 입력된 경우, "INVALID SCORE" 출력 후, 다시 점수를 입력 받는다.

```
# add
Student ID: 20210001
ALREADY EXISTS

# add
Student ID: 20210021
Name: Lee Hyori
Midterm Score: 930
INVALID SCORE - Please retype: 93
Final Score: 95
Student added

# add
Student ID: 20210006
Name: Lee Sangsun
Midterm Score: 77
Final Score: 66
Student added

# show
Student          Name          Midterm   Final   Average   Grade
-----
20210021         Lee Hyori         93        95      94.0      A
20210002         Lee Jieun         92        89      90.5      A
20210009         Lee Yeonghee      81        84      82.5      B
20210001         Hong Gildong      84        73      78.5      C
20210006         Lee SangSun       77        66      71.5      C
20210007         Kim Cheolsu       75        62      68.5      D
20210011         Ha Donghun        58        68      63.0      D

#
```

## 5. searchgrade (Grade 검색)

- **searchgrade** 입력 시, 특정 grade 를 입력 받아 그 grade 에 해당하는 학생을 모두 출력한다.
- 예외처리:
  - ✓ A, B, C, D, F 외의 값이 입력된 경우, "WRONG INPUT" 메시지 출력
  - ✓ 해당 grade 의 학생이 없는 경우, "NO RESULTS" 출력

```
# searchgrade
Grade to search: E          ➔ A, B, C, D, F 외의 값이 입력된 경우
WRONG INPUT

# searchgrade
Grade to search: F          ➔ 해당 grade의 학생이 없는 경우

Student      Name      Midterm  Final  Average  Grade
-----
NO RESULTS

# searchgrade
Grade to search: D

Student      Name      Midterm  Final  Average  Grade
-----
20210007     Kim Cheolsu    75       62     68.5     D
20210011     Ha Donghun     58       68     63.0     D

#
```

## 6. remove (특정 학생 삭제)

- **remove** 입력 시, 아래와 같이 삭제하고자 하는 학생의 학번을 입력 받은 후, 학생이 목록에 있는 경우, 학생 정보 출력 후 삭제하겠냐는 메시지에 y 를 입력하면, 삭제 후 메시지 "Student removed"를 아래와 같이 출력한다.
- 삭제 시, 해당 학생 정보 저장을 위해 동적 할당 받은 메모리를 할당 해제(free) 한다.

```
# remove
Student ID: 20210011

Student      Name      Midterm  Final  Average  Grade
-----
20210011     Ha Donghun     58       68     63.0     D

Remove 20210011 (y/n)? y
Student removed

# show
Student      Name      Midterm  Final  Average  Grade
-----
20210021     Lee Hyori      93       95     94.0     A
20210002     Lee Jieun      92       89     90.5     A
20210009     Lee Yeonghee   81       84     82.5     B
20210001     Hong Gildong   84       73     78.5     C
```

```
20210006      Lee SangSun      77      66      71.5      C
20210007      Kim Cheolsu      75      62      68.5      D
```

```
# remove
```

```
Student ID: 20210033
```

```
NO SUCH PERSON
```

→ 해당 학번의 학생이 없는 경우

예외처리:

- ✓ 해당 학번이 목록에 없는 경우에는 "NO SUCH PERSON"이라는 에러 메시지 출력
- ✓ 목록에 아무도 없을 경우, 아래의 예제와 같이 "LIST IS EMPTY" 메시지 출력

```
# remove
LIST IS EMPTY
```

```
#
```

## 7. quit (프로그램 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 같은 내용을 구성한다.

**[Student number][Wt][Name][Wt][Midterm][Wt][Final][Wn]**

- 저장할 때 목록의 순서는 평균을 기준으로 내림차순으로 한다.
- 파일 이름은 확장자를 포함하여 최대길이는 30 자이며, 파일 이름에 공백이 없다고 가정한다.
- 동적 할당된 memory 를 모두 free 시킨 후 종료한다.

```
# quit
```

```
Save data(y/n)? y
```

```
File name: newStudents.txt
```

```
계속하려면 아무 키나 누르십시오. . .
```