Assignment #2. Object Oriented Programming, Spring Semester, 2024

2023202022 이현지

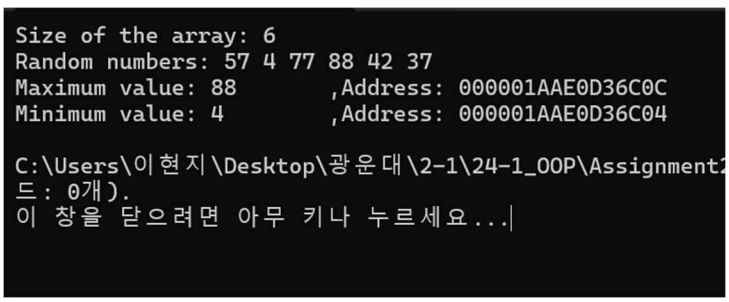
1. 랜덤한 수를 메모리를 동적할당한 배열(크기: 5~20)에 채워 넣어라. 이때 각 수의 범위는 0부터 100까지이다. 그 후, 배열의 모든 값을 출력하고 최솟값과 최댓값, 각각의 주소도 출력하여라.

   A. 메인 함수에서 배열의 크기를 입력 받고 크기에 맞게 동적할당 한다. 랜덤한 수를 만들어야 하므로 srand함수를 이용한다. 최댓값과 최솟값으로 이동하는 경로도 설정해준다.

   B. maximum함수는 배열 내에서 가장 큰 수를 찾는다. 정수형 변수 max에 배열의 첫 번째 수를 첫 비교대상으로 넣고, for반복문으로 배열을 끝까지 돌리면서 최댓값을 찾는다. 최댓값의 주소를 출력해야 하므로, 포인터에 주소값도 같이 넣어준다.

   C. minimum함수는 배열 내에서 가장 작은 수를 찾는다. maximum함수와 골조는 동일하다.

[출력화면]



[고찰]

다 풀고 나서, 이번에 새로 배운 참조 연산자를 사용해서도 풀 수 있을 것 같아 시도해봤더니 생각대로 잘 되었다. 여러가지 방안을 생각해보는 것이 좋은 공부방법인 것 같다.

2. 10개의 랜덤한 수를 생성하고, quick sort와 merge sort를 이용하여 오름차순으로 정렬하라. 그 후 입력받은 숫자를 찾고, 없을 시에는 올바른 자리에 숫자를 추가하여 정렬하여라.

   A. 메인함수

i. 랜덤한 수를 생성해야하므로 srand함수를 사용했다. 10개의 수를 ran배열에 추가해주었다.

B. QuickSort

i. pivot이라는 기준점이 필요하다. pivot을 어떻게 설정하느냐에 따라서 시간복잡도가 달라진다. quicksort는 pivot을 기준으로 왼쪽 부분과 오른쪽 부분으로 나눈다. 그렇기에 pivot이 항상 가장 작거나 가장 크면 시간복잡도가 커질 수 밖에 없다. 양 극단의 경우 배열의 크기가 줄어들어봤자 -1이기 때문에 배열의 크기만큼 재귀함수를 돌리게 된다. 이때 시간복잡도는 n^2이 된다. (크기가 n인 배열에 대하여)

ii. 최선의 경우는 pivot을 중위값으로만 골랐을 때이다. 이때는 mergesort처럼 배열이 절반으로 쪼개질 때마다 데이터의 개수가 반으로 줄게 되므로 $\log n$의 시간이 소요된다. 요소의 개수 n개 만큼의 비교하는 시간이 소요되므로 시간복잡도는 $n\log n$이 된다.

iii. separate함수는 pivot을 기준으로 배열을 나누는 함수이다. pivot을 배열의 가장 끝 부분으로 잡았다. for반복문으로 배열의 시작부터 pivot전까지 탐색하도록 한다. 배열의 수가 pivot보다 작으면 Swap함수로 배열의 두 부분이 이동한다. 변수 i를 low-1로 잡아 배열의 가장 왼쪽 끝부분부터 시작하고 j는 i바로 다음에 위치시킨다. j에 위치한 수가 pivot보다 클 경우에는 i와 위치를 변경하기 때문에 큰 수들은 오른쪽으로 올라가게 된다.

iv. quickSort함수는 separate에서 반환받은 pivot위치를 토대로, quickSort를 재귀시켜 low와 high의 위치가 반전될 때까지 진행시킨다.

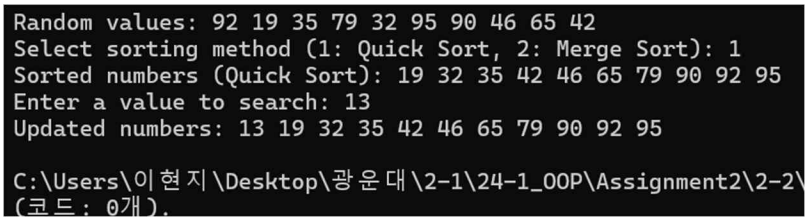
C. MergeSort

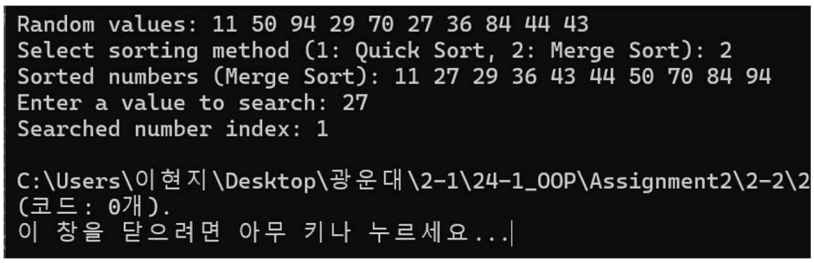
i. mergeSort는 quickSort의 최선의 경우일 때와 동일하다. 배열을 절반으로 쪼개고, n개의 요소가 있기 때문에 시간복잡도는 $n\log n$이 된다.

ii. mergeSort함수는 mid로 중간지점을 계산하고, left와 right의 위치가 반전될 때까지 재귀적으로 정렬시킨다. 다 진행한 이후에는 merge함수로 이동해서 나누어진 배열들을 재조립하는 과정을 거쳐 알고리즘을 완료시킨다.

D. swap함수는 low와 high의 주소를 매개변수로 받아서 그들의 주소를 서로 변경한다.

[출력화면]





[고찰]

퀵소트와 머지소트에 대해 공부하면서 다양한 자료구조가 있고, 그에 따라 시간복잡도가 다르다는 것을 알게 되었다. 다양한 자료구조 중에서 이 두가지가 가장 효율적인 방법임을 알게 되었다.

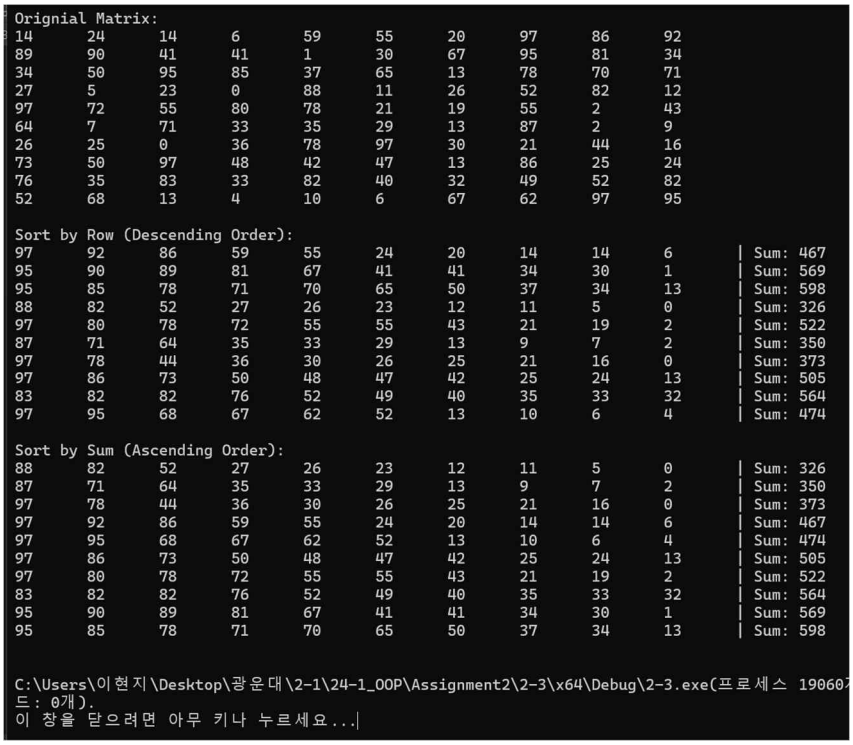
3. 랜덤한 수로 저장되어있는 int**형의 10x10행렬을 출력하는 프로그램을 작성하여라. 행을 기준으로 내림차순 정렬하여 매트릭스를 출력하고, 각 행의 합계를 오름차순으로 정렬하여 출력하여라. 합계를 정렬할 때는 포인터를 이용하여 주소를 이동시켜야 한다.

   A. 메인 함수에서는 랜덤한 수를 생성할 수 있도록 srand함수를 선언해주고, 이차원배열을 생성하였다. int**형태에 저장해야 하므로 포인터배열을 사용하여 동적할당 하였다. 합계를 저장해야 하므로 열은 11개, 행은 10개로 설정하였다.

   B. des함수는 각 행에 따라 내림차순 정렬한다. for이중반복문을 사용하여 요소들끼리의 대소를 비교했다. asc함수는 각 행에 따라 오름차순 정렬한다. des함수와 구조는 동일하다.

   C. swap함수는 바꾸어야 할 대상의 주소를 매개변수로 받아, 주소를 바꾸는 역할을 한다.

[출력화면]



[고찰]

정렬하기위해 배열의 주소를 swap하는 과정에 참조를 적용해봤다. 유용하게 잘 사용할 것 같다.

4. 문자열을 입력 받고, 원하는 구분자에 따라 나누는 프로그램을 작성하여라. 구분자는 공백도 포함한다. 문자열은 크기가 100까지이고, 구분자는 10까지이다.

   A. 메인함수

      i. 문자열을 입력 받고 다른 서브 함수로 이동하도록 한다.

   B. token함수

      i. 입력받은 문자열을 구분자에 따라 포인터배열ptr에 저장하도록 한다.

      ii. 문자열이 널문자를 만나기 전까지 while반복문을 돌린다.

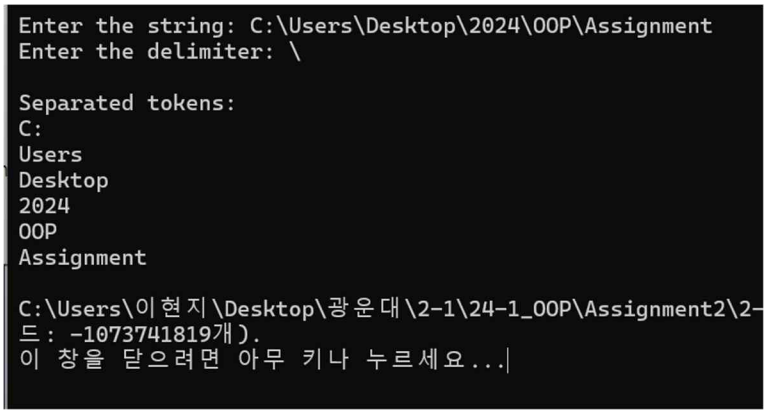
         1. 구분자의 첫글자가 동일할 경우, 그 이후도 구분자와 동일한지, 구분자가 끝날 때까지 확인한다.

         2. 만약 구분자가 끝날 때까지 while문을 돌았다면 sep의 마지막 위치는 널문자일 것이기에, if문으로 검사한다. 맞다면 ptr에 동적할당하고, 구분자 이전 문자를 넣어준다.

3. 입력받은 문자열이 끝나지 않았다면 다시 while문으로 돌아간다.

4. 마지막 토큰의 경우, 구분자가 뒤에는 없기 때문에, 따로 직접 넣어준다.

[출력화면]



5. Hardmard 매트릭스를 작성하여라. Hardmard매트릭스는 재귀함수로 작성하여라. 사용자에게 n을 입력받고 2^n크기의 정방행렬을 만들고 출력하여라.

   A. 메인 함수

      i. n을 입력 받고, power연산을 진행해 Hardmard매트릭스의 총 크기를 계산한다.

      ii. 매트릭스의 제일 처음인 (0,0)은 1로 고정이기 때문에 미리 넣어주고, had함수로 이동한다.

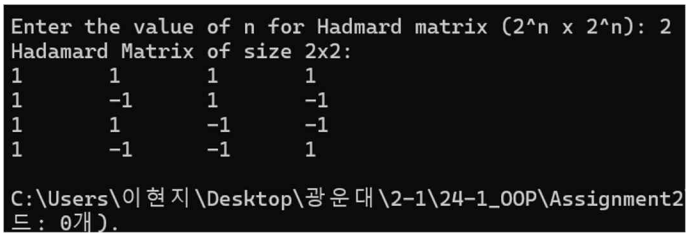
   B. had함수

      i. n이 0일 경우에는 1x1이므로 그대로 return시킨다.

      ii. 재귀함수로 설정했기 때문에, 이전 배열 *2가 n을 power of 2시킨 값보다 크다면 중지시킨다.

      iii. 그렇지 않을 경우에는 현재의 배열을 2사분면으로 삼고, 1사분면, 3사분면에는 동일한 요소, 4사분면에는 -1을 곱한 값을 넣는다.

[출력화면]



6. 각 명령어 (command, search, change, insert, delete, save, exit)에 따라 작동하는 프로그램

을 작성하여라. open은 원하는 파일을 연다. search는 원하는 단어를 찾고, 단어의 위치를 첫글자의 행과 열을 이용하여 출력한다. change는 입력 받은 두 단어 중 첫번째 단어를 두번째 단어로 바꾼다. insert는 입력 받은 행과 열의 위치에 입력 받은 단어를 넣는다. delete는 입력 받은 단어를 파일의 모든 내용에서 지운다. exit은 프로그램을 종료한다.

A. 메인 함수

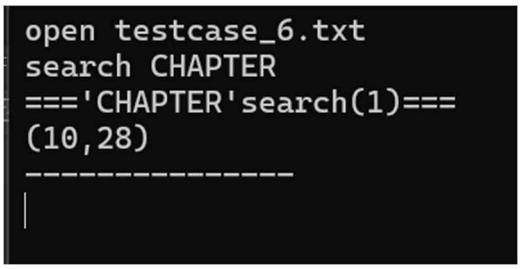
i. 파일을 이용하기 때문에 #include <fstream>을 사용한다. 파일을 읽고 쓸 예정이기 때문에 각각 readFile과 writeFile로 변수를 설정해준다.

ii. command로 open을 입력 받으면 파일을 열고, 파일이 끝나기 전까지 getline으로 한 줄 씩 입력받아 포인터배열인 line에 저장한다.

iii. search를 입력 받으면 line배열이 끝나기 전까지 원하는 문자를 찾도록 한다.

iv. 나머지 command는 구현을 완료하지 못했다. 주석으로 묶어두었다.

[출력화면]



[고찰]

- change의 명령어의 경우, 우선 바꿔줄 단어를 찾아야 한다 생각해, search의 알고리즘을 그대로 사용했다. 그러나, 바꾸고 싶은 단어의 크기가 다를 경우에는 배열을 모두 움직여야 하는 번거로움이 있다는 것을 알게 되었다. 그래서 애초부터 입력을 다르게 받아야 하는 건가 생각했으나, 다 해내지 못했다.

7. 사용자에 의해 입력 받은 formula를 계산하는 프로그램을 작성하라. 숫자는 정수형태이고, 사칙연산만 가능하다. 먼저 계산해야 하는 부분은 stack의 원리를 이용하여라. 입력가능한 숫자는 0~255, 연산자는 0~8, 소괄호 쌍은 0~8까지 가능하다.

A. 메인 함수

i. 입력받은 식에 있는 공백과 같은 것들을 지우기 위해서 blank함수를 작성하였다.

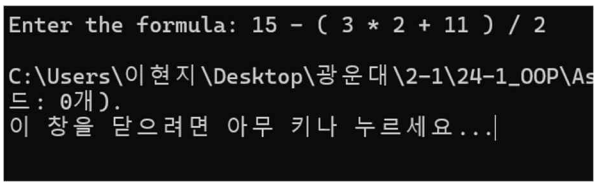
ii. 사칙연산, 괄호에 계산의 우선순위가 있기 때문에 구분자를 구별하기 위해서

classify를 이용해서 괄호 시작, 괄호 끝, 곱셈나눗셈, 덧셈뺄셈, 숫자 이렇게 5가지로 나누었다.

iii. 정리한 우선순위는 priority배열에 넣어서, 계산할 때 우선순위를 참고하여 계산하면 된다.

iv. 분류는 다했으나, stack의 원리를 활용하여 calculate함수를 작성하는 것에서 막혔다.

[출력화면]



[고찰]

계속 배열을 사용하고 있었는데, 이번 문제를 풀면서 배열의 단점 때문에 계속 문제가 막혔다. 배열을 보완한 다른 list같은 방안은 없는지 알아봐야겠다.

8. 아래에 첨부된 사진을 참고하여 class를 설정한다. class를 이용하여 학생들의 정보를 저장하고 출력하도록 한다. 또한 command에 맞춰 프로그램이 작동하도록 한다. insert는 class에 학생의 이름, 학번, 과제점수, 시험점수, 출석을 입력하도록 하고, find는 입력 받은 이름을 찾아 해당 학생의 정보를 출력한다. change는 입력 받은 학생의 정보를 바꾼다. print는 저장된 모든 학생들의 정보를 출력한다. exit는 프로그램을 종료시킨다.

   A. Student 클래스

      i. 학생들의 정보를 저장해야 하므로 char*형 변수(name, studentID)를 가지고, double형 변수(각종 점수, 출석)를 가진다.

      ii. 매개변수 생성자로 초기화와 선언을 동시에 진행하였다. command로 insert가 입력되면, 입력 받는 학생의 정보로 초기화한다.

      iii. command로 find가 입력되면 class포인터에서 학생을 찾아야 하므로 isNameCorrect함수를 사용해, 일치한지 확인한다. 매개변수로 학생의 name 포인터를 받고 일치할 경우 1을 반환한다.

      iv. change를 입력 받았을 때, 학생의 점수를 수정할 수 있는 changeScores함수를 선언하였다.

   B. 메인 함수에서는 student클래스의 포인터배열를 설정하여, 여러 클래스를 다루어

여러 학생을 다룰 수 있도록 한다. 명령을 입력받을 때 공백도 포함하므로 cin.getline을 이용한다.

i. 명령어의 첫글자가 e가 아니라면 while문을 계속 반복하게 하였다. 명령어 'exit'이 나오기 전까지 프로그램을 작동시키기 위해서이다.

ii. insert를 입력받았을 때는 입력받은 정보 중 점수는 double형태로 저장되어야 하기 때문에, convert 함수를 사용하여 char형을 double형으로 변환시킨다.

1. double형으로 변환된 숫자들은 Student클래스의 포인터배열인 students를 이용해 Student생성자의 매개변수로 넣어준다. 이때, 학생들의 점수에 따른 등급도 같이 계산해준다.

iii. find를 입력받았을 때는 isNameCorrect함수를 이용해 일치한지 확인하고 찾으면 학생의 정보를 출력하고, 없을 경우 not found를 출력하도록 하였다.

iv. change를 입력받았을 때는 원하는 학생의 클래스를 찾고 changeScores 함수를 이용하여 정보를 변경하였다.

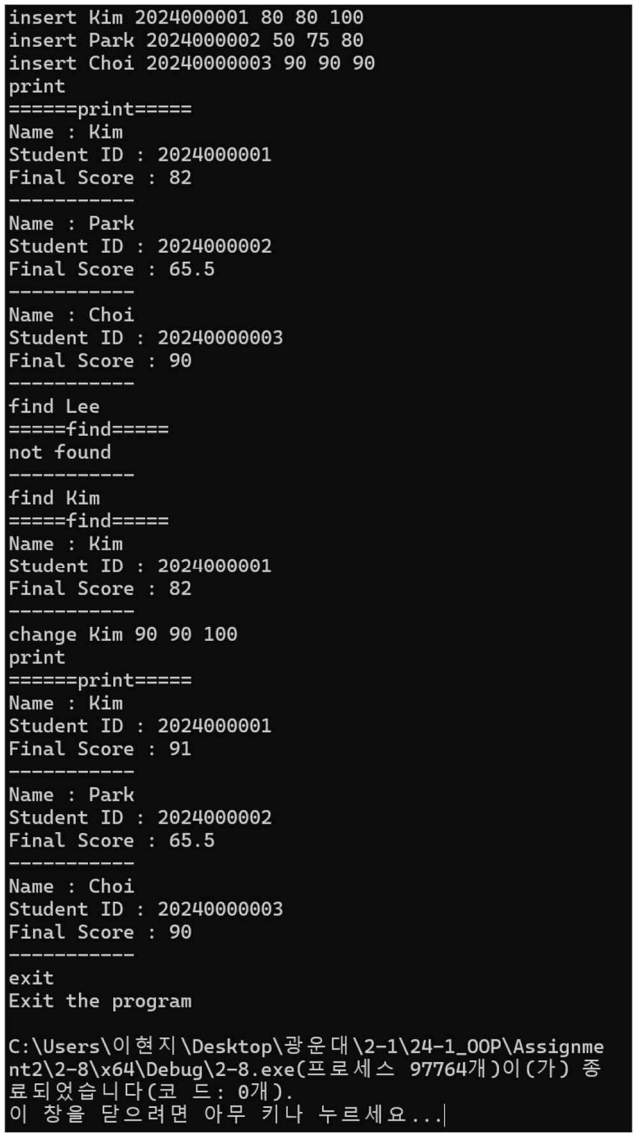
v. print를 입력받았을 때는 print함수를 통해 학생 정보를 출력하였다.

C. double형을 반환하는 convert함수는 cin.getline으로 받아서 char형이 된 학생들의 점수를 double형으로 변환하기 위한 함수이다.

i. 학생들의 점수는 문자형배열에 한자리씩 저장되어 있기 때문에 점수의 자릿수를 파악하면 십진법으로 변환하기만 하면 된다. 아스키코드표를 확인해보면 문자형인 숫자형태는 0~9까지밖에 없다. 또한, 10진수로 변환시켰을 때 각 문자에서 '0'을 빼면 double형으로 변환했을 때 원하는 숫자가 나온다. 이를 활용해서 double형으로 변환시켰다.

D. 입력받은 명령어는 공백을 기준으로 정보를 나누어야 한다. separate함수를 이용해 명령어를 나누었다. 매개변수로 char* assign과 char** ptr을 받는데, char* assign은 입력받은 명령어 배열을 말하고, char** ptr은 메인에서 정의한 char* ptr[6]를 받기 위함이다. char* ptr[6]는 포인터 배열로, 6행을 가지고 열의 크기는 동적할당 된 이차원배열이다. 공백을 기준으로 나눈 단어들은 ptr[0], ptr[1],...이렇게 저장된다.

[출력화면]



[고찰]

클래스 포인터 배열을 다루면서 동적할당과 포인터에 대한 깊은 이해가 되었다. 또한, 변수나 함수에 접근에 있어서 까다로운 부분이 있어, 프로그램의 자동 수정 기능을 많이 이용하였다. 기본기가 탄탄하지 못해서 많이 헤맸던 구간이어서 더 깊은 이해가 필요하다 생각이 들었다.

9. 학생들의 정보를 저장하고 출력하는 프로그램을 작성한다. 여러 학생들을 유연하게 다루기 위해 school classs를 이용한다. student class는 school class외부에서 접근할 수 없다. 각 클래스의 멤버변수는 첨부된 사진과 동일하다. 멤버함수는 자유롭게 이용해도 된다. 프로그램은 명령어에 따라 작동한다. 이름과 학번은 크기가 100까지이다. new_student는 school 클래스에 새로운 학생의 정보를 저장한다. sort_by_score는 학생들의 성적을 기준으로 내림차순 정렬한다. print_all는 school class에 저장된 모든 학생의 정보를 출력한다. print_A_grade는 상위 30%의 점수를 가진 학생들을 출력한다. print_B_class는 상위 50%

의 점수를 가진 학생들, A등급을 제외한 학생들을 출력한다. exit는 프로그램을 종료한다.

A. Student 클래스

i. 8번 문제에서 사용한 클래스와 거의 동일하다. 추가된 함수는 char* getname(), char* getID(). double getScore()이다. student클래스는 school 클래스 밖에서는 접근이 불가하기 때문에 클래스 내의 변수의 값을 얻기 위해서는 이런 함수가 필요하다.

B. School 클래스

i. student 클래스의 포인터변수를 변수로 가지고 있다. 학생들의 정보를 더 효율적으로 관리하기 위해서이다. size변수는 학생들의 수를 얘기한다.

ii. 함수로는 생성자를 초기화하기 위한 setInfo, 정렬하기 위한 sort, 출력하기 위한 print, grade를 계산하기위한 calculateGrade가 있다.

1. SetInfo함수에서는 student_list배열에 따라 Student클래스를 생성하도록 하였다.

2. sort함수는 student_list배열을 이용해 student클래스의 getScore()함수를 이용해 점수를 가져와 이들을 비교하고, swap하도록 설계했다.

3. calculateGrade함수는 학생들의 수에서 각각 0.3, 0.5씩을 곱해서 해당 범위에 들어가는 학생들을 각각 0,1그룹에 넣었다.

C. 메인 함수

i. student클래스는 school클래스와는 달리 외부에서 접근이 불가하므로 생성자를 메인함수에서 호출하지 않는다. school클래스 또한 호출과 동시에 초기화는 어렵기 때문에, 기본생성자를 호출 후 클래스 매서드를 통해 학생들의 정보를 저장한다.

ii. command를 입력받고 분할하는 함수들은 이전의 8번 문제와 동일하고, char* ptr의 행 길이는 각 문제에서 최대로 받는 단어 개수로 지정한다.

iii. command의 첫 글자를 distinguish라는 함수로 보내서 각 command마다 0부터 1씩 가산해서 반환하도록 하였다.

iv. switch-case문을 통해 각각의 command를 작동하도록 하였다.

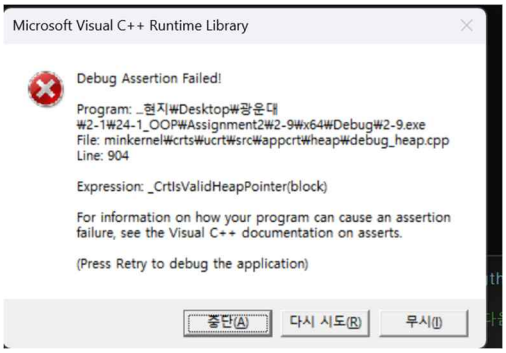
[출력화면]

```
new_student Olivia 201916625 95
new_student Emma 20135123512 55
new_student Amelia 20140000002 71
new_student Ava 20155654 64
new_student Sophia 201321562123 80
print_all
=====print=====
Name : Olivia
StudentID : 201916625
Name : 95
-------------
Name : Emma
StudentID : 20135123512
Name : 55
-------------
Name : Amelia
StudentID : 20140000002
Name : 71
-------------
Name : Ava
StudentID : 20155654
Name : 64
-------------
Name : Sophia
StudentID : 201321562123
Name : 80
-------------
sort_by_score
print_all
=====print=====
Name : Olivia
StudentID : 201916625
Name : 95
-------------
Name : Sophia
StudentID : 201321562123
Name : 80
-------------
Name : Ava
StudentID : 20155654
Name : 64
-------------
Name : Amelia
StudentID : 20140000002
Name : 71
-------------
Name : Emma
StudentID : 20135123512
Name : 55
-------------
exit
Exit the program

C:\Users\이현지\Desktop\광운대\2-1\24-1_OOP
가) 종료되었습니다(코 드: 3개).
이 창을 닫으려면 아무 키나 누르세요...
```

```
new_student Olivia 201916625 95
new_student Emma 20135123512 55
new_student Amelia 20140000002 71
new_student Ava 20155654 64
new_student Sophia 201321562123 80
print_A_grade
=====A grade=====
Name : Olivia
StudentID : 201916625
Name : 95
-------------
print_B_grade
=====B grade=====
Name : Sophia
StudentID : 201321562123
Name : 80
-------------
exit
Exit the program

C:\Users\이현지\Desktop\광운대\2-1\24-
드: 3개).
이 창을 닫으려면 아무 키나 누르세요...
```

[고찰]

exit을 입력하고 나면



이런 경고 팝업창이 떴었다. 메모리 해제 부분에서 char* ptr[4]는 메모리 할당한 게 두번째 차원에서만 할당했기 때문에 반복문을 이용해서 ptr[0],ptr[1]...에 있는 메모리를 해제해줘야 한다. 이를 깨닫기 전에는 포인터 메모리 할당에 대한 지식이 부족해서 어떤 것이 잘못되었는지 몰라 헤맸었다.

10. 9번 문제를 기반으로 student.txt에서 데이터를 제공받아서 학생들의 정보를 저장한다. 9번과의 차이점은 file에서 학생의 정보를 받아 저장한다는 것이다. 파일의 내용은 ',(comma)'를 구분점으로 한다. 모든 행은 열의 길이가 동일하며, 이름과 학번의 최대크기는 100이다. 9번의 command중 new_student는 load_student로 변경되어, 파일을 통해 학생들의 정보를 저장하도록 한다.

    A. student class와 school의 class는 9번 문제와 동일하다.

    B. 파일에서 데이터를 읽어오는 것이므로 #include <fstream>문을 추가했다. 읽어들일 파일을 readFile 변수로 두고 command의 첫글자가 l일 때 파일을 오픈하도록 하였다.

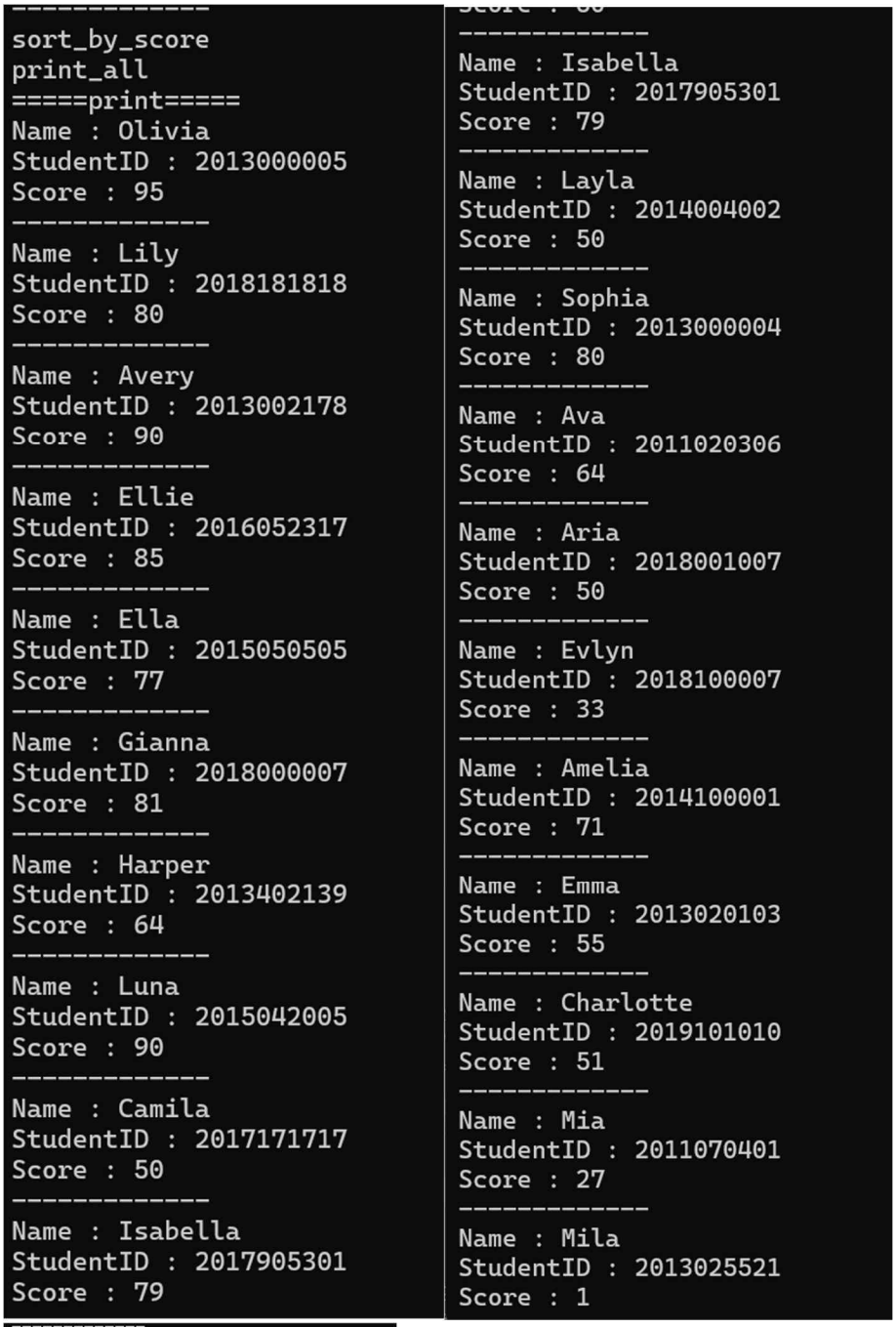
        i. load_student를 입력하면 'student.txt'파일을 오픈하고, 한줄씩 포인터배열인 line에 저장한다.

        ii. print를 입력하면 line에 저장되어있던 데이터를 separate함수로 이동시켜서 콤마를 구분자로 정보를 분류시킨다. 또한 char형으로 받아진 점수데이터는 convert함수로 보내서 double형으로 반환받도록 한다. 이후 school변수를 이용해서 정보를 저장하고 클래스 내의 print함수를 이용해 학생들의 정보를 출력한다.

        iii. exit을 입력받으면 프로그램을 종료시킨다.

[출력화면]

```
load_student
print_all
=====print=====
Name : Olivia
StudentID : 2013000005
Score : 95
-------------
Name : Emma
StudentID : 2013020103
Score : 55
-------------
Name : Amelia
StudentID : 2014100001
Score : 71
-------------
Name : Ava
StudentID : 2011020306
Score : 64
-------------
Name : Sophia
StudentID : 2013000004
Score : 80
-------------
Name : Charlotte
StudentID : 2019101010
Score : 51
-------------
Name : Isabella
StudentID : 2017905301
Score : 79
-------------
Name : Mia
StudentID : 2011070401
Score : 27
-------------
Name : Luna
StudentID : 2015042005
Score : 90
-------------
Name : Harper
StudentID : 2013402139
Score : 64
-------------
Name : Gianna
StudentID : 2018000007
Score : 64
-------------
Name : Gianna
StudentID : 2018000007
Score : 81
-------------
Name : Evlyn
StudentID : 2018100007
Score : 33
-------------
Name : Aria
StudentID : 2018001007
Score : 50
-------------
Name : Ella
StudentID : 2015050505
Score : 77
-------------
Name : Ellie
StudentID : 2016052317
Score : 85
-------------
Name : Mila
StudentID : 2013025521
Score : 1
-------------
Name : Layla
StudentID : 2014004002
Score : 50
-------------
Name : Avery
StudentID : 2013002178
Score : 90
-------------
Name : Camila
StudentID : 2017171717
Score : 50
-------------
Name : Lily
StudentID : 2018181818
Score : 80
-------------
sort_by_score
```



[고찰]

파일의 문자를 읽어오는 과정에서 파일의 끝이 널문자로 끝나는지 ENTER로 끝나는지 알 수

없어서 처음엔 널문자로 초기화를 해서 작동시켰었다. 그러나 어째선지 무한루프에 빠져 프로그램이 무한 작동하였다. 그래서 '0'으로 배열과 포인터배열을 초기화한 후 입력을 받으려 했으나, 학번이나 점수에 0이 있어서 곤혹을 치렀었다. 초기화의 중요성과 널문자가 만능은 아니라는 것을 깨닫게 된 시간이었다.

11. 정수의 값을 연결리스트로 저장하는 프로그램을 작성하여라. 프로그램은 총 3개의 command로 있다. insert는 숫자를 리스트에 입력하도록 한다. delete는 원하는 모든 숫자를 삭제한다. exit는 프로그램을 종료시킨다. insert와 delete command를 입력할 때마다 list를 순서대로 출력하도록 한다.

    A. 클래스

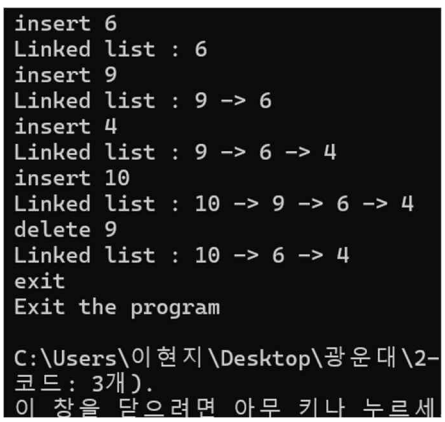
        i. 노드에 원하는 값을 넣기 위한 key와 다음을 가리키는 포인터를 변수로 설정한다. 입력받은 값을 넣고, 다음 노드를 설정할 수 있는 함수가 있고, 노드는 연결리스트 외부에서 접근이 불가하기 때문에 key를 반환하고, 다음 노드의 주소를 반환하는 함수가 필요하다.

            1. insertKey함수는 pNew를 새로운 노드로 정의해주고 pHead가 null이거나 key값이 pHead의 값보다 클 경우 pHead가 pNew로 변경된다. 값을 넣은 이후에는 다음 노드와 연결해주는 과정이 필요하다. 또한 pCur에 대해서도 key값을 비교해 정렬해야한다.

            2. deleteKey함수는 pCur노드를 사용해, pHead부터 key값을 확인 후 동일한 노드가 발견될 경우, 지운다.

        ii. 연결리스트는 가장 처음을 의미하는 pHead의 주소가 필요하다. 연결리스트를 통해 노드에 값을 넣기 때문에 insertKey, deleteKey와 같은 함수가 필요하다.

    [출력화면]



    [고찰]

C++로는 연결리스트를 처음 써보는 것이다 보니 시간이 오래 걸렸다. 앞선 문제에서 배열이 불편하다고 느꼈었는데, 연결리스트를 자유자재로 사용할 수 있으면 훨씬 효율적인 작업이 될 것이라 생각한다.

12. char의 최대크기는 15로 하는 영어 단어를 철자 순으로 저장하는 프로그램을 작성하여라. 단어를 저장하기 위해서, 프로그램은 이차원 연결리스트를 사용해야 한다. 리스트의 첫번째는 단어의 첫글자는 동일하고, 이후부터 알파벳 순서로 정렬한다. 두번째는 단어의 첫글자를 알파벳 순서를 정렬한 것이다.

    A. 11번의 연결리스트를 골조를 참고해서 char* key로만 변경을 했다. 이중연결리스트는 어떻게 하는지 숙지하지 못해 다 완료하지 못했다.