**자료구조 실습보고서**

**실습 2-3. 동적 메모리**

**2016년 3월 16일**

**학번:201404051**

**이름 :정용석**

1. **실습 문제 소개**

실습1-3 때와 동일한 프로그램이다. 하지만 이번 실습에는 구조체와 동적 할당을 이용하여 구현 화하게 되었다. 학생수와 3과목에 대한 점수가 들어있는 score.txt을 열어 동적 할당 된 구조체에 정의 된 변수에 저장한 뒤, 이를 이용하여 평균과 총합을 계산 저장한 뒤, 추가적으로 과목별 총 학생의 평균 점수를 마지막으로 report.txt에 다시 적는 실습이다.

1. **소스코드**

#include <stdio.h>

#include <stdlib.h>

#pragma warning(disable:4996)

//학생 구조체 정의. 각 학생의 3개의 점수와 그 합과 평균을 저장할 수 있는 변수가 들어있다.

typedef struct STUDENT

{

int score[3];

int sum;

int average;

}STUDENT;

int main()

{

FILE\* fp;

int nStudent; //학생수

int nSubject = 3; //고정 과목수 3

int i, j;

int subjectAve[3]; //과목별 평균

int subjectTot[3]; //과목별 총점

int chaser = 0; //score.txt의 실제 입력 된 학생 수 확인 용 변수.

STUDENT\* student;

fp = fopen("score.txt", "r"); //score파일 오픈

if (fp == NULL)

printf("ERROR!\n");

else

printf(" file successfully opened!\n\n");

fscanf(fp, "%d", &nStudent); //총 학생수 저장

//score.txt에서 알아낸 학생수만큼 동적할당 \*sizeof(studnet) = 20 int형자료 5개!

student = (STUDENT\*)malloc(sizeof(STUDENT)\*nStudent);

for (i = 0; i < nStudent; i++)

{

if (feof(fp) != 0) //파일이 끝나면 반복문 종료

break;

chaser++; //반복문이 몇 번 돌았나 확인 = 몇 명의 학생 수의 점수가 인식되었는지 확인

student[i].sum = 0;//총합 초기화

for (j = 0; j < nSubject; j++)

{

fscanf(fp, "%d", &student[i].score[j]);

student[i].sum += student[i].score[j];

student[i].average = student[i].sum / nSubject;

}

}

//과목별 총 평균 및 총점 계산

for (i = 0; i < nSubject; i++)

{

subjectTot[i] = 0;

subjectAve[i] = 0;

for (j = 0; j < chaser; j++) //실제 입력된 학생 수 만큼 반복

{

subjectTot[i] += student[j].score[i];

subjectAve[i] = subjectTot[i] / nStudent; //실제 입력된 학생 수? 주어진 학생수?

}

}

//score.txt의 학생수와 실제 입력된 학생점수와 불일치시 출력내용

if (chaser != nStudent)

{

printf(" THE GIVEN NUMBER OF STUDENTS AND THE NUMBER OF SCORES DO NOT MATCH!! \n\n");

printf(" REPORT.TXT WILL BE WRITTEN WITH THE CURRENT NUMBER OF SCORES!! \n\n");

}

fp = fopen("report.txt", "wt");

if (fp == NULL)

printf("ERROR!\n");

fprintf(fp, "%d\n", nStudent); //학생수 적고

//score.txt파일에 저장된 값 입력시 chaser만큼의 데이터만 입력하게 만들었다. 그 이후의 자료는 쓰레기

//값으로 불필요함.

for (i = 0; i < chaser; i++)

fprintf(fp, "%d %d\n", student[i].sum, student[i].average);

for (i = 0; i < nSubject; i++)

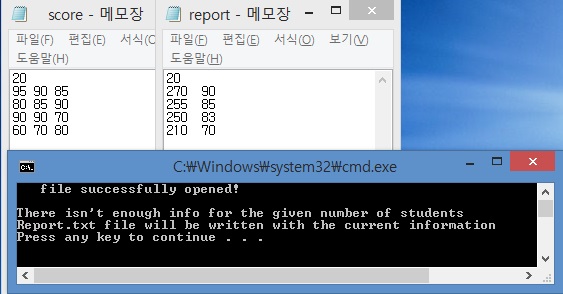
fprintf(fp, "%d ", subjectAve[i]);

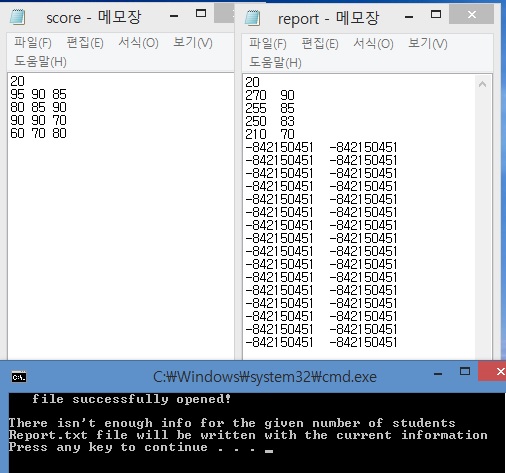
//메모리 수거

free(student);

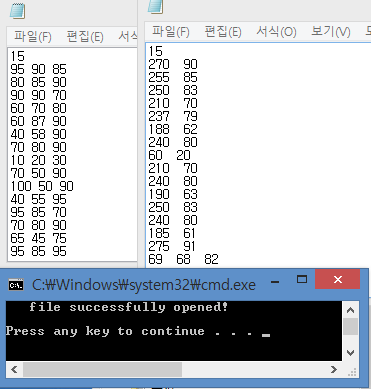
return 0;

}

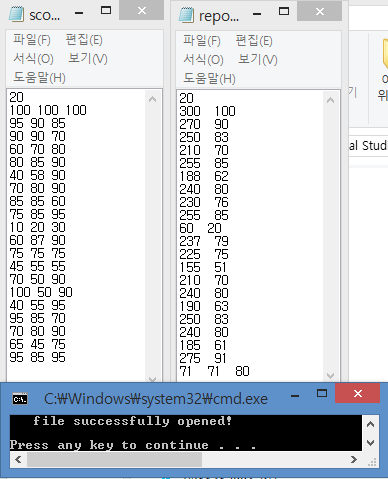
1. **테스트**
2. 학생 수에 대한 과목 점수 불일치 시
3. Score.txt의 학생 수 내용과 실제 과목 점수 개수 불일 치 나타나는 현상

c)cc

1. 정상적인 출력 결과



1. 정상적인 출력 결과 2



1. **작성자 코멘트**

이번 실습은 저번 실습과 전체적인 구조는 차이가 없고, 구현 방법에 대한 차이만 있었기에, 따로 문제 해결방법에 대한 이해가 필요없었기에 굉장히 수월했다. 굉장히 낯설었던 점은 구조체를 동적 할당 했을 시에 배열처럼 사용할 수 있다는 점이 새로웠다. 분명히 배운 부분이라 생각했지만, 기억이 희미 했기에 이번 기회에 확실히 개념을 잡을 수 있었다. 구조체 정의와 이를 포인터를 이용하여 동적할당하는 부분까지는 문제가 없었고, 문제라고 할만한 것은 report.txt를 출력할 때, score.txt의 학생 수보다 적은 혹은 더 많은 수의 점수 정보가 있을 시에 였다. 위의 테스트 a)와 같이 20명의 학생수로 입력 받았지만 4명의 점수만 있을 시에는 report.txt에는 4명의 총합과 평균과 더불어 밑에는 구조체 배열에 아무것도 저장 되어있지 않은 쓰레기 값이 출력되어 같이 나온다. 이를 방지하기 위해서 개인적으로 chaser라는 변수를 설정하여, 반복문 안에 삽입하여 feof함수를 이용하여 파일을 다 읽었을 때 반복문이 몇 번 돌았는지, 즉 몇 명의 학생이 실제로 score.txt안에 있는지 chaser변수안에 저장 후, report.txt 출력 시에도 이를 이용하여 실제 학생 수만큼만 출력하게 하였다. 또한 이러한 정보들이 불일치 시 커멘트창에 이에 대한 정보도 출력시켜 주었다. 여기서 고민한 점은, 그렇다면 학생수와 그에 상응하는 점수에 대한 정보가 불일치할 때에 report텍스트 파일 출력 시, 첫 줄에 학생 수에 대한 정보 입력 시, score에 있는 학생 수를 넣을 것인지, 아니면 실제로 입력 되어있는 성적의 학생 수를 넣을 지에 대한 것이었다. 물론 여러가지 방법을 할 수도 있었지만, 지금은 크게 중요하다고 생각하지 않아 score 텍스트 파일에서 저장된 학생 수를 그대로 출력하게 구현 하였다. 추가적으로 저번 실습과는 다르게 report.txt파일 생성 시 가장 밑 줄에 과목별 평균을 추가하는 것이었다. 크게 어렵지는 않았다. 학생 별 총합과 평균을 구했을 때와 반대의 행렬로 반복문을 사용하여 쉽게 찾아낼 수 있었다. 다른 점이라고 하면 학생 구조체에는 이를 저장하는 변수가 없기에 메인함수에 따로 총합과 평균을 저장할 수 있는 배열 변수를 선언하고 계산하였다. 이때도 마찬가지로 주어진 학생 수와 실제 학생 수에 대한 점수 정보가 일치하지 않을 시에 대한 고민이 있었다. 왜냐하면 5명의 학생 수를 입력 받고 4명 분의 점수만이 있다면, 평균을 4명의 평균을 넣어야 할 지, 아니면 5명의 평균을 그대로 넣어야 할 지 갈피를 못 잡았지만, 딱히 주어진 조건이 없기에 nStudent, 주어진 학생 수로 평균을 내었다. 이것들를 제외하고는 큰 문제없이 프로그램을 구현, 테스트할 수 있었다.