**2024 Spring OOP Assignment Report**

과제 번호 : Assignment #2 - Prob1

학번 : 20230563

이름 : 김홍근

Povis ID : hongsimi7

**명예서약 (Honor Code)**

나는 이 프로그래밍 과제를 다른 사람의 부적절한 도움 없이 완수하였습니다.

I completed this programming task without the improper help of others.

1. **프로그램 개요**
   * 본 프로그램은 학생들의 성적을 추가, 편집, 출력 등 여러 기능을 하는 프로그램이다.
   * Structure을 사용하여 학생의 정보를 저장한다.
   * Loop & Branch 문을 사용하여, 등급을 구별하거나 정렬을 하는 등의 작업을 한다.
2. **프로그램의 구조 및 알고리즘**

* Int형 변수로 choice, error, count를 생성하여, 각각 유저의 입력, 오류 여부, 학생 수 카운트를 저장한다.
* Struct로 stu\_info를 생성하여, 학생 정보를 다 담았다. 여기서 id, name, midterm\_exam\_score, final\_exam\_score, retake, grade는 각각 학번, 이름, 중간 점수, 기말 점수, 재수강 여부, 학점을 저장한다.
* Stu\_info 형의 stu\_list를 통해 학생 정보를 저장할 수 있는 array를 생성한다.
* While 문을 통해 종료를 입력 받을 때까지 프로그램이 실행되도록 했다.
* 메뉴 화면에서 만약 1을 입력 받았다면, stu라는 학생을 생성하고, stu 안의 멤버 변수인 id, name, midterm\_exam\_score, final\_exam\_score, retake을 입력 받는다. 이때 오류가 생긴다면 오류 메시지를 출력한 다음, 메뉴화면으로 돌아간다. 성공적으로 입력이 되었다면, 이를 stu\_list에 추가하고, count에 1을 더하여 학생 수를 추가한다.
* 메뉴 화면에서 만약 2를 입력 받았다면, 중간 혹은 기말 점수 중 무엇을 바꿀지 입력을 받고, 학생 ID를 입력 받는다. 이 때, ID가 존재하지 않는다면, 오류 메시지를 띄우고 메뉴화면으로 돌아간다. ID가 있다면 점수를 받되, 1~100의 범위를 벗어날 경우 오류 메시지를 띄우고 메뉴화면으로 돌아간다. 성공적으로 점수를 입력을 받으면 정상적으로 점수를 편집한다.
* 메뉴 화면에서 만약 3을 입력 받았다면, 평균 점수와 표준 편차를 계산해서 출력한다. 이때, 학생 수가 0명이라면 평균 점수: 0, 표준 편차: 0을 출력한다. 전체 점수를 계산할 때, for을 사용해 간편하게 덧셈을 실시했다.
* 메뉴 화면에서 만약 4를 입력 받았다면, bubble sort를 통해 총점 내림차순으로 정렬을 실시한다. 이때, 총점이 같다면, 학번 오름차순으로 bubble sort를 사용해 똑같이 정렬을 한다. 그러고 나서 점수표를 출력한다.
* 메뉴 화면에서 만약 5를 입력 받았다면, bubble sort를 통해 총점 내림차순으로 정렬을 실시한 다음, 각각의 조건에 맞게 등급을 메긴다. 그리고 나서 등급표를 출력한다.
* 메뉴 화면에서 만약 6을 입력 받았다면, 프로그램을 종료한다.

1. **토론 및 개선**
   * + 구조체를 생성하는 방법 및 구조체를 유용하게 사용하는 방법에 대해 알 수 있었다. 구조체를 사용하면 훨씬 더 간결한 코딩이 가능하다는 것이다.
     + If & Branch문을 사용하여 간결한 코딩을 할 수 있었다.
     + 등급표를 출력할 때, 먼저 모든 학점을 ‘D’로 두고 시작했다. 이는 불필요한 계산을 막기 위해서이다. 그리고 버블 정렬을 통해 총점 내림차순으로 정렬했다. A학점을 받을 사람을 먼저 계산했다. 그리고 A학점을 받은 사람을 제외시켜 계산을 줄이고, B학점을 받을 사람을 계산했다. 그 다음, 앞에서 학점을 받은 사람은 제외시키고, C학점을 받을 사람을 계산했다. 마지막으로 for문을 이용하여 재수강 여부를 확인한 다음, 다운그레이드를 실시하였다.
     + 등급표 출력에서 내가 사용한 방법보다 훨씬 간결하고 속도가 빠른 방법이 무수히 많다고 생각한다. 따라서 다른 알고리즘을 다시 구현해보는 것이 실력 향상에 도움이 될 것이라 생각했다.
2. **참고 문헌**
   * + 해당사항 없음.