문제 정의

10-1 문제는 Dept 클래스를 정의하고, 그 클래스를 사용하여 학생들의 성적을 관리하며 60점 이상을 받은 학생의 수를 계산해야 하는 프로그램입니다. 학과에 속한 학생들의 성적을 관리하는 클래스를 생성하고 학과에 속한 학생 수를 받아서 성적 배열을 할당할 생성자, 복사 생성자, 소멸자를 만듭니다.
크기가 10인 Dept 객체를 생성해서 10명의 학생 성적을 저장할 수 있게끔 하고 10명의 성적을 입력받는 함수 및 60점 이상인 학생 수를 계산하고 확인하는 함수를 만들고 60점 이상인 학생 수를 출력해야 하는 문제입니다.
10-3번 문제는 10-1의 코드를 사용하여 복사 생성자를 제거하고 복사 생성자 없이 오류가 발생하지 않게 하기 위해서는 수업시간에 배운 참조 매개변수를 사용하면 될 것이라고 생각합니다.

문제 해결 방법 및 알고리즘 설명

이 문제를 풀기 위해서는 깊은 복사 생성자를 사용해야합니다. 깊은 복사 생성자를 사용하는 이유는 main.cpp에서 "int n = countPass(com);" 코드를 통해 복사 생성자가 호출됩니다. 복사 생성자가 없다면 디폴트 복사 생성자가 호출되는데 디폴트 복사생성자는 기본적으로 얕은 복사를 수행하여 com 객체의 scores와 디폴트 복사 생성자 객체의 scores는 같은 메모리를 가리키게 됩니다. 그렇기 때문에 소멸자가 실행될 때, 디폴트 복사 생성자의 객체가 소멸하여 메모리를 반환하고, com 객체가 소멸될 때 동적 메모리가 중복하여 해제하려고 하기 때문에 오류가 발생할 것입니다. 따라서 깊은 복사 생성자를 작성하여 깊은 복사를 수행시켰습니다.

그렇기 때문에 클래스 선언부에서는 성적 배열의 크기(학생 수) 및 학생들의 성적을 저장하기 위해 동적 할당을 받을 정수 배열을 포인터 변수로 선언하고 생성자 및 메모리 공유 및 얕은 복사로 인한 오류를 없애기 위한 깊은 복사 생성자, 그리고 소멸자를 작성합니다. 학생들의 성적을 입력받는 함수 read(), 그리고 학생 성적이 60 이상인지 확인하는 함수 isOver60()을 선언했습니다.

클래스 구현부에서는 깊은 복사 생성자를 통해 기존 객체의 size 값을 복사하여 저장하고 새로운 배열을 동적으로 할당시켜 반복문을 통해 기존에 있던 배열을 복사하였습니다. getSize()함수에서는 size 값을 반환한 후, read()함수에서는 for문을 통해 학생 수 만큼 점수를 입력하여 scores 배열에 저장할 수 있도록 합니다.
마지막으로 isOver60()함수로 저장 된 배열에서 리턴 타입을 bool로 설정하여 학생의 점수가 60점 이상이면 true, 60점 이하면 false를 반환하였습니다.

마지막으로 메인 함수 입니다. 학생 10명이 있는(Dept) 객체를 생성하여 read()함수를 통해 10명의 성적을 저장할 수 있게끔 합니다. 그 후 합격자 수를 계산하기 위해서 countPass 함수를 이용하여 60점 이상인 학생 수를 카운트하고 변수 n에 저장한 후 출력합니다. 메인 함수에서 사용한 CountPass()함수는 매개변수로 Dept 객체를 전달받습니다. 60점 이상의 학생들의 수를 0으로 초기화 하고, for문에서 dept 객체의 getSize()함수를 호출하여 객체의 멤버 변수 size만큼 반복합니다. 60점 이상이면 isOver60()함수를 사용해 true나 false가 반환되고 count++가 실행하도록 하고 변수 count를 반환합니다.

CountPass 함수의 매개변수로 10-1번 문제에서는 값에 의한 호출로 (Dept dept)를 받아왔지만 복사 생성자를 제거하고 10-3번 문제의 해결 방법으로 참조에 의한 호출로 문제를 해결하면 되겠습니다. 참조에 의한 호출을 사용하면 객체가 동일한 메모리 주소를 공유하게 됩니다. 참조를 사용하면 원래 객체의 메모리에 직접 접근하므로 복사본이 필요 없습니다. 동일한 배열을 두 객체가 공유하게 되어 메모리 소멸 문제를 피할 수 있습니다.