객체지향 프로그래밍 과제 보고서

202104389 황형진, 202104309 이진용

1. 문제 정의

1번에서 작성한 코드에서 복사 생성자를 제거하고 원래의 기능을 유지하는 것이 문제입니다. 기존 코드에선 동적 할당을 사용한 멤버 변수가 있었기 때문에 깊은 복사를 위한 복사 생성자가 필요했습니다. 이를 위해 동적 할당을 제거하고 정적 배열을 사용해 문제를 해결해볼 것입니다.

2. 문제 해결 방법

1. 동적 할당 제거: 동적 할당을 사용하는 scores를 위해 복사 생성자가 필요했습니다. 이를 해결하기 위해 동적 할당을 제거하고, 정적 배열을 사용하도록 변경하였습니다.

2. 얕은 복사: 복사 생성자를 제거하고 컴파일러가 자동으로 제공하는 디폴트 복사 생성자를 활용하도록 했습니다. 정적 배열을 사용하면 얕은 복사만으로도 안전하게 데이터를 복사할 수 있어, 복사 생성자를 작성하지 않아도 문제가 없습니다.

3. 아이디어 평가

복사 생성자가 없을 때 동적 할당된 배열의 복사는 소멸 시 메모리 문제가 생길 수 있습니다. 이를 정적 배열을 사용하여 복사 생성자 없이도 컴파일러 제공 복사 생성자를 통해 객체를 복사할 수 있고 에러 없이 코드가 실행될 수 있도록 했습니다.

4. 문제 해결한 키 아이디어 또는 알고리즘 설명

얕은 복사와 깊은 복사: 원래의 코드에서는 동적 할당된 배열을 복사할 때 얕은 복사로 인해 발생할 수 있는 메모리 문제가 있었습니다. 얕은 복사란 객체의 멤버 변수들을 단순히 값만 복사하는 것이며, 이로 인해 동적 메모리를 복사할 때 원본과 사본이 같은 메모리 공간을 공유하고 이는 소멸시 문제가 생깁니다. 이를 해결하기 위해 깊은 복사 생성자가 필요했으나, 이번에는 동적 할당 자체를 제거하여 얕은 복사로도 오류없이 작동하도록 했습니다.

정적 배열 사용: scores에서 동적 할당에서 정적 배열로 변경함으로써, 복사 생성자 없이도 객체의 복사가 가능해졌습니다. 정적 배열은 객체가 소멸될 때 자동으로 메모리가 해제되므로, 메모리 관리에 대한 부담이 줄어듭니다.