객체지향프로그래밍
7주차 과제 소스구현 설명

ICT융합학부 소프트웨어학과
202284002 고범주

1.문제 정의:

이 프로그램은 Dept 클래스를 활용하여, 학생들의 성적을 관리하고 60점 이상인 학생들의 수를 계산하는 프로그램을 구현하는 것이 목표입니다. 성적을 입력받고, 그 성적이 60점 이상인 학생들의 수를 확인하는 기능을 제공합니다. 기본적으로 동적 할당을 사용해 성적 데이터를 처리합니다

2.문제 해결 방법:

(1) Dept 클래스 복사생성자 설계

Dept 클래스는 학생들의 점수를 동적으로 관리하는 배열과 배열의 크기를 저장하는 멤버 변수를 포함합니다. 또한 생성자와 소멸자를 통해 동적 메모리 관리를 수행하며, 복사 생성자를 통해 깊은 복사를 지원합니다.

(2) read 함수 설계

read() 함수에서 사용자로부터 10개의 점수를 입력받아, scores 배열에 저장합니다. 이 배열은 동적으로 할당되므로 입력 후에도 유지됩니다.

(3)isOver60 함수 설계

isOver60() 함수는 특정 인덱스의 점수가 60점 이상인지를 확인하고, countPass() 함수는 전체 학생 중 60점 이상인 학생의 수를 계산합니다.

3.아이디어 평가

(1)동적 메모리 관리

Dept 클래스에서 new 연산자를 통해 동적으로 배열을 할당하고, 소멸자에서 delete[]를 사용하여 메모리를 해제하는 구조는 효율적입니다. 이는 프로그램이 메모리 누수 없이 데이터를 처리하도록 돕습니다.

(2) 복사 생성자 구현

깊은 복사를 지원하는 복사 생성자를 통해, 객체 복사 시 배열의 데이터를 안전하게 복사합니다. 이를 통해 객체 간 데이터를 독립적으로 관리할 수 있습니다.

(3)성적 처리의 유연성

성적 데이터의 비교는 isOver60() 함수를 통해 캡슐화되어 있어, 점수 기준이 바뀌더라도 함수만 수정하면 전체 프로그램에 영향을 주지 않고 유연하게 변경이 가능합니다.

4.문제를 해결한 키 아이디어 또는 알고리즘 설명

(1)함수 설명

Dept(int size): 생성자로, size 크기의 성적 배열을 동적으로 할당합니다.

~Dept(): 소멸자로, 동적으로 할당된 성적 배열을 해제하여 메모리 누수를 방지합니다.

Dept(const Dept& dept): 복사 생성자로, 객체 복사 시 원본과 독립적인 배열을 할당하고 데이터를 복사하여 관리합니다.

void read(): 사용자로부터 성적을 입력받아 배열에 저장하는 함수입니다.

bool isOver60(int index): 해당 인덱스의 성적이 60점 이상인지 확인하는 함수로, 60점 이상이면 true, 그렇지 않으면 false를 반환합니다.

int countPass(Dept dept): 입력된 성적 중 60점 이상인 학생의 수를 계산하는 함수입니다. 이 함수는 Dept 객체를 매개변수로 받아, 각 성적을 isOver60() 함수로 체크하여 60점 이상인 성적을 카운트합니다.

(2)깊은 복사 사용

깊은 복사를 통해 메모리를 안전하게 관리할 수 있습니다. 각 객체가 자신의 메모리를 할당하고 해제함으로써, 메모리 누수나 이중 해제(double free) 문제를 방지할 수 있습니다.

Dept 클래스의 소멸자에서는 delete[]를 사용하여 동적으로 할당된 메모리를 해제하므로, 객체가 파괴될 때 메모리 관리가 확실하게 이루어집니다.

Dept 클래스의 복사 생성자는 원본 객체의 상태를 정확히 복사하여 새로운 객체를 생성하는 데 사용됩니다. 이를 통해 다른 곳에서 복사된 객체를 안전하게 사용할 수 있습니다.

복사 생성자를 사용하면 countPass 함수에서 Dept 객체를 매개변수로 받을 때, 값에 의한 복사 (pass by value)가 발생하며, 이때 깊은 복사가 적용됩니다. 이는 프로그램의 안전성과 안정성을 높입니다.