<객체지향프로그래밍_6주차과제_소스구현설명>

7조(202104140김경록,202104276이선우)

(1)번 문제 코드

```
#include<iostream>
using namespace std;

class Dept {
    int size;
    int* scores;

public:
    Dept(int size) {
        this->size = size;
        scores = new int[size];
    }

    Dept(Dept& dept);

    ~Dept();

    int getSize() { return size; }
    void read();
    bool isOver60(int index);
};
```

```
int countPass(Dept dept) {
	int count = 0;
	for (int i = 0; i < dept.getSize(); i++) {
		if (dept.isOver60(i)) count++;
	}
	return count;
}

Dept::Dept(Dept& dept) {
	this->size = dept.size;
	this->scores = new int[this->size];
	for (int i = 0; i < this->size; i++) {
		this->scores[i] = dept.scores[i];
	}
}

Dept::~Dept() {
	delete[] scores;
}

void Dept::read() {
	cout << size << "개 점수 입력>> ";
	for (int i = 0; i < size; ++i) { // 배열 크기만큼 반복
		cin >> scores[i];
	}
```

```
}

bool Dept::isOver60(int index) {
    if (scores[index] > 60)
        return true;
    else
        return false;
}

int main() {
    Dept com(10);
    com.read();
    int n = countPass(com);
    cout << "60점 이상은 " << n << "명";
}
```

(3)번 문제 코드 (복사생성자 제거)

```
#include<iostream>
using namespace std;

class Dept {
    int size;
    int* scores;

public:
    Dept(int size) {
```

```
        this->size = size;
        scores = new int[size];
    }

    ~Dept();

    int getSize() { return size; }
    void read();
    bool isOver60(int index);
};

int countPass(Dept& dept) {
    int count = 0;
    for (int i = 0; i < dept.getSize(); i++) {
        if (dept.isOver60(i))
            count++;
    }
    return count;
}

Dept::~Dept() {
    delete[] scores;
}

void Dept::read() {
```

```
    cout << size << "개 점수 입력>> ";
    for (int i = 0; i < size; ++i) {
        cin >> scores[i];
    }
}

bool Dept::isOver60(int index) {
    if (scores[index] > 60)
        return true;
    else
        return false;
}

int main() {
    Dept com(10);
    com.read();
    int n = countPass(com);
    cout << "60점 이상은 " << n << "명";
}
```

----문제 정의----

(1번, 3번) 이 문제는 Dept 클래스의 scores 배열에 size만큼의 학생들의 점수를 기입하고 점수가 60점 이상인 학생들이 몇 명인지 출력하는 프로그램이다. 배열, 즉 주소를 사용하기 때문에 복사 생성자는 필수적이다. 그 이유는 깊은 복사 즉 메모리 중복 반환 오류 때문이다. 그리고 복사 생성자를 제거하였을 때 오류 없이 실행시키도록 만들게 해야 합니다.

----문제 해결 방법, 아이디어들----

(1번) 먼저 우리는 main()함수의 실행결과가 주어진 결과와 같게 만들기 위해서 Dept 클래스를 구현해야 했습니다. Dept 클래스에는 멤버 변수와 멤버 함수, 복사 생성자, 소멸자 등이 선언되어 있었습니다. 우리는 선언된 이것들을 구현하기 위해 몇몇 방법과 아이디어를 통해 구현했습니다. 먼저 복사 생성자부터 구현을 했습니다. 이 때 우리는 깊은 복사를 위해 다른 주소에 객체를 생성해서 Dept 클래스를 복사하기 위해 위와 같은 코드를 작성하는 방법을 택했습니다. 즉, 안정성을 위하는 아이디어였습니다. 다음으로 소멸자는 객체 소멸할 때 할당된 메모리를 없애기 위해 delete[] scores; 를 작성하였습니다. 그리고 read()함수는 이름 그대로인 역할을 해야 하므로 size를 입력받아 scores배열에 저장시켰습니다. 마지막으로 isOver60()함수는 점수가 60점 넘었는가에 대한 함수인데 이 함수는 정수형인 index라는 변수를 매개변수로 가지고 있으므로 이를 이용해 점수들이 담긴 scores배열에서 매개변수 index를 이용해 60점이 넘었으면 true를 반환, 아니면 false를 반환시켰습니다.

(3번) 복사 생성자 없이 코드를 실행시키면 실행은 되나 오류 하나가 발생하게 됩니다. 이 문제는 메모리에 대한 문제 같았는데 그래서 우리는 countPass() 함수에서 복사 생성자가 없으니 복사 없이 원본 객체를 참조시켜 문제 없이 코드를 실행시켰습니다.

----아이디어 평가----

문제 해결을 위해 제시한 위의 아이디어들은 문제에서 주석이 달려있어 보다 작성하기는 수월하였습니다. 작성한 코드들에서 복사 생성자를 깊은 복사 생성자로 만드는 것은 성공적이었고 코드가 잘 실행되었습니다. 또한 복사 생성자를 없앴을 때 실행시켜보았을 때 오류가 발생하여 고민을 많이 해 메모리 문제라는 것을 유추하여 참조를 사용하였는데 이 아이디어는 없어서는 안되는 중요한 아이디어였습니다.

----문제를 해결한 키 아이디어 또는 알고리즘 설명----

(1)번 문제를 해결할 수 있게 만든 키 아이디어는 복사 생성자를 만들 때 깊은 생성자를 만드는 것이었습니다. 만약 얕은 복사를 만들었다면 메모리가 자칫 같은 곳을 가리키게 되어 문제를 야기하게 되어 문제가 될 것 같았습니다. 그래서 복사 생성자를 사용한 것이 키 아이디어였습니다.

(3)번 문제를 해결할 수 있게 만든 키 아이디어는 바로 참조라는 아이디어 덕분이었습니다. 복사 생성자가 없는 3번 문제에서는 객체가 같은 메모리를 공유해 오류가 생겼는데, 이를 참조연산자를 통해 원본 객체를 참조하게 시켜 메모리를 달리 하게 만드는 것이 키 아이디어였습니다.