<객체지향프로그래밍_5주차과제_소스구현설명>

7조(202104140김경록,202104276이선우)

```
#include<iostream>
#include<string>
#include <cstdlib>
#include <ctime>
using namespace std;
class Player{
	string name;
public:
	void inputName(string name);
	string returnName() { return name; };
};
class GamblingGame {
	Player *p = new Player[2];
public:
	GamblingGame();
	void nameInput();
	string ranNum(string n);
	void startGame();
	~GamblingGame() { delete [] p; }
};
GamblingGame::GamblingGame(){
	cout << "***** 갬블링 게임을 시작합니다. *****₩n";
	srand(time(NULL));
```

```
}

void GamblingGame::nameInput() {
	string name;
	cout << "첫번째 선수 이름>>";
	cin >> name;
	p[0].inputName(name);
	cout << "두번째 선수 이름>>";
	cin >> name;
	p[1].inputName(name);
}
string GamblingGame::ranNum(string n){
	int r[3];
	cout << "₩t₩t";
	for (int i = 0; i < 3; i++) {
		r[i] = rand() % 3;
		cout << r[i] << "₩t";
	}
if(r[0] == r[1] && r[0] == r[2]) {
		n += "님 승리!!";
		return n;
	}
	else
		return "아쉽군요!";
}
```

```
void GamblingGame::startGame(){
	string n;
	int i = 0;
	while(true){
		string m;
		cout << p[i % 2].returnName() <<"엔터키를 입력하시오" << "₩n";
		getline(cin, n);
		m = p[i % 2].returnName();
		n = ranNum(n);
		if(n == "님 승리!!") {
			cout << m + n;
			break;
		}
		else
			cout << n << endl;
		i++;  //
	}
}
void Player::inputName(string n){
	name = n;
}
int main(){
	GamblingGame game;
	game.nameInput();
```

```
    game.startGame();
}
```

----문제 정의----

위 문제는 플레이어 2명의 이름을 저장하고 각 플레이어의 차례마다 랜덤한 숫자를 출력하여 3개가 같을 경우 이기는 게임입니다. 플레이어들은 배열로 구성하며 Player클래스로 작성하고 게임은 GamblingGame클래스로 작성하는 것입니다. 그렇게 하여 main함수에서는 별다른 코드 없이 클래스 멤버함수를 불러내어 프로그램을 동작할 수 있습니다.

----문제 해결 방법, 아이디어들----

그래서 생각했던 큰 틀의 헤더파일은 iostream, string, cstdlib, ctime을 선택했습니다. 그 이유는 iostream은 이 프로그램에서 필수적으로 필요한 입력과 출력을 위한 것입니다. 그리고 string 헤더파일은 플레이어의 이름을 받기 위해 문자열 처리를 위한 것입니다. Cstdlib는 게임에서 랜덤 숫자를 뽑아내기 위해 필요한 것입니다. Ctime은 시간 관련 함수를 이용해 랜덤 숫자를 뽑기 위해 보조적으로 쓰기 위해 설정했습니다. 먼저, Player 클래스, GamblingGame 클래스를 선언하여 이 게임을 진행해야 합니다. 그래서 저희는 생각했던 것이 Player 클래스에서는 플레이어의 이름을 저장하고 이 이름을 반환하는 식으로 생각했습니다. 그래서 Player 클래스의 멤버 변수는 name, 함수로는 inputName, returnName으로 설정하였습니다. Name 변수는 이름을 저장할 스트링형 변수이고 inputName 함수는 함수의 이름 의미 그대로 멤버의 이름을 받는 함수입니다. returnName 함수는 받은 이름을 리턴해주는 역할을 하는 함수입니다. 그 후 GamblingGame 클래스에서 Player 클래스를 상속받아 포인터형 변수인 *p를 선언합니다 이때 배열을 2명으로 설정하여 플레이어의 이름을 저장할 수 있도록 선언합니다. 그리고 GamblingGame() 생성자를 선언합니다. 이 때 과제를 하는 과정에서 이 생성자를 삽입하지 않았을 때 숫자가 랜덤하게 나오지 않은 문제점을 발견하고 어떻게 해결하면 좋을 지에 대해 고민하다가 이 생성자에 랜덤한 숫자를 생성할 때 사용하는 srand(time(NULL)) 함수를 이용하였더니 보다 더 랜덤한 숫자가 나온다는 점을 발견하였습니다. 그 후 nameInput 함수입니다. 먼저 스트링형 변수인 name을 설정하고 각 선수의 이름을 p[0], p[1]에 각각 저장합니다. 그 후 ranNum 함수를 사용하여 각 플레이어가 차례마다 3개의 랜덤한 숫자를 출력 받을 수 있도록 설정합니다. 이때 승리조건인 3개의 숫자가 모두 같은 경우 승리하는 메시지를 출력하도록 설정합니다. 다음으로 stratGame 함수입니다. 변수 i=0을 선언하여 순서를 제어하기 위한 인덱스를 설정합니다. 같은 숫자 3개가 나와 승리 조건을 만족하기 전까지 무한루프를 하기위해 while(true)문을 활용합니다. 이때 플

레이어의 순서를 어떻게 하면 차례가 돌아갈 수 있을지에 대해 생각해보다가 인덱스 i를 활용하는 것을 떠올렸습니다. 인덱스 i가 초기에 0인 것을 활용하여 p[0], p[1]를 표현하기 위해서는 % 연산자를 활용해야 한다는 것을 생각했습니다. 최초의 플레이어는 p[0]이고 엔터키를 눌러 차례가 돌아갈 수 있도록 설정합니다. 그리고 선언했던 스트링형 변수인 n,m을 활용하여 이때 Player 클래스의 멤버함수인 returnName을 이용하여 m에 현재 플레이어의 이름을 저장하고 n에 ranNum 함수를 이용해 게임에서 출력되는 3개의 숫자를 받아옵니다. 그리고 if 구문을 이용하여 승리조건을 확인합니다. 3개의 숫자가 같을 경우엔 승리한 플레이어의 이름과 메시지를 출력하고 break를 이용해 게임을 종료합니다. 하지만 승리하지 못했을 경우에는 아쉽군요! 메시지를 출력하고 i++구문을 이용하여 % 연산자를 이용했을 때 나머지가 바뀌는 것을 이용해 차례를 변경합니다. 그리고 GamblingGame의 소멸자를 이용해 게임이 종료되었을 경우 동적 m로 할당된 메모리를 해제하며 프로그램의 안정성을 높였습니다. 메인 함수에는 GamblingGame의 객체를 생성하고 nameInput 함수를 이용해 플레이어의 이름을 설정하고 startGame을 통해 게임을 시작합니다.

----아이디어 평가----

클래스를 만들어 main 함수에서 별 다른 코드들 없이 프로그램을 작동할 수 있게 만들 수 있게 하는 것은 처음이라 낯설었습니다. 그러나 각 클래스에서 생성자, 함수들, 소멸자 등을 사용한 것은 코드의 가독성을 높였습니다. 우리는 그 안에서 여러 아이디어들을 사용하였는데, 랜덤성을 부여하기 위해서 rand함수를 이용하는 것, 이름을 받아 리턴 할 수 있게 만드는 inputname, returnname함수 그리고 엔터키 입력을 대기할 수 있게 하는 getline(cin,n)이라는 코드 등을 만들었습니다. 이 아이디어들을 평가하자면 없어서는 안될 코드였으며 getline(cin,n)이라는 코드는 엔터키를 쳤을 때 넘어가게 하는 것인데 이 아이디어가 좋았던 것 같습니다.

----문제를 해결한 키 아이디어 또는 알고리즘 설명----

우리는 생성자를 사용하지 않고 srand(time(NULL))을 다른 함수에 넣었을 때 랜덤하지 않은 것을 발견했으며 생성자에 따로 넣었을 때 랜덤하게 나오는 것을 확인할 수 있었습니다. 이 때 생성자에 따로 넣는 아이디어를 생각하게 되었는데 이것이 키 아이디어였다고 생각합니다. 만약 다른 함수에 그냥 넣었다면 srand(time(NULL))이 main함수에서 계속 실행하게 되어서 랜덤한 숫자가 나오기도 전에 출력이 되는 경우가 잦아 랜덤한 결과를 볼 수 없었을 것입니다. 그러나 생성자에 넣어서 srand(time(NULL))이 한번만 실행되

게 하여서 랜덤한 결과를 도출할 수 있었습니다. 즉, 랜덤한 결과를 만들게 한 이것이 이번에 저희 코드에서 키 아이디어 및 코드라고 생각합니다.