문제 정의

이번 문제는 두 명의 선수가 번갈아 가며 랜덤한 3개의 숫자를 생성하여 모두 동일한 숫자가 나오면 승리하는 갬블링 게임을 만드는 것 입니다. 각 선수의 이름은 처음 시작 시 입력 받게 되며, 숫자의 범위는 0에서 2로 제한되어 있어 일치할 확률을 높입니다. 이 게임은 Player클래스와 GamblingGame 클래스를 제작하여 해결합니다.

문제 해결 방법

문제를 해결하기 위해 Player클래스를 생성하여 각 선수의 이름을 관리하기 위해 사용합니다. GamblingGame클래스는 게임 전체를 제어하는 역할을 합니다. 게임 진행, 선수 순서, 랜덤한 숫자 생성과 비교를 통해서 승자를 결정하는 등 모든 과정을 제어합니다.
이 게임의 핵심은 각 선수의 순서에서 3개의 랜덤 숫자를 생성하고 이를 비교하는 것 입니다. 숫자의 범위를 0에서 2로 제한하여 동일한 숫자가 나올 가능성을 높였습니다.
두 명의 선수가 번갈아 가며 게임을 진행 하도록 반복문을 사용했으며 각 턴에서 엔터키를 누르고 랜덤한 숫자가 생성 되며 결과가 표시 되게 됩니다. 세 숫자가 동일할 시 승자가 결정되며 게임이 종료됩니다.

아이디어 평가

Player클래스와 GamblingGame클래스는 문제 해결 방법에 적어둔 방식으로 잘 제작되어 선수의 이름을 관리하고 게임 시작과 종료 순서 제어 승리조건 등의 과정을 잘 수행합니다. 랜던 숫자 생성의 범위를 제한 하여 승리의 확률이 적절하다는 것을 결과화면을 통해 증명하였습니다.

문제를 해결한 키 아이디어 또는 알고리즘 설명

키 아이디어는 위에서 계속 언급한 것처럼 랜덤 숫자 생성과 비교 입니다. 세 개의 랜덤한 숫자를 생성하고, 모두 동일한 경우 해당 선수를 승리자로 선언하는 것이 이 게임의 핵심입니다.

이제 각 코드의 주석을 통해 설명을 이어 가겠습니다.

Player.h 코드

```
#ifndef PLAYER_H
#define PLAYER_H
#include <string>
// Player 클래스: 선수의 이름을 관리하는 클래스
class Player
{
std::string name; // 선수의 이름 저장
public: void setName(std::string name); // 선수의 이름 설정
std::string getName(); // 선수의 이름 반환
};
```

Player.cpp 코드

```
#include "Player.h"
// setName 함수: 선수의 이름을 설정하는 함수
void Player::setName(std::string name)
{
this->name = name;
}
// getName 함수: 선수의 이름을 반환하는 함수
std::string Player::getName()
{
return name;
```

}

GamblingGame.h 코드

```
#ifndef GAMBLINGGAME_H
#define GAMBLINGGAME_H
#include "Player.h"
#include <iostream>
// GamblingGame 클래스: 게임의 전체 로직을 관리하는 클래스
class GamblingGame
{
Player p[2]; // 두 명의 선수를 저장하는 배열
public:
void play(); // 게임을 시작하는 함수
~GamblingGame(); // 소멸자, 게임 종료 메시지를 출력
};
#endif // GAMBLINGGAME_H
```

GamblingGame.cpp 코드

```
#include "GamblingGame.h"
#include <cstdlib> // 랜덤 숫자 생성을 위한 라이브러리
#include <ctime> // 현재 시간을 시드로 사용하기 위한 라이브러리
// 소멸자: 게임 종료 시 메시지를 출력
GamblingGame::~GamblingGame()
```

```
{
std::cout << "게임을 종료합니다." << std::endl;
}

// play 함수: 게임의 전체 진행 로직을 관리
void GamblingGame::play()
{
std::string p1, p2;
std::cout << "***** 겜블링 게임을 시작합니다. *****" << std::endl;
std::cout << "첫번째 선수 이름>>";
std::cin >> p1;
p[0].setName(p1); // 첫 번째 선수 이름 설정
std::cout << "두번째 선수 이름>>";
std::cin >> p2;
p[1].setName(p2); // 두 번째 선수 이름 설정
std::cin.ignore(); // 입력 버퍼 비우기
srand((unsigned)time(0)); // 랜덤 시드 초기화

int end = 0; // 게임 종료 여부를 확인하는 변수
while (true)
{ // 게임이 끝날 때까지 반복
for (int k = 0; k < 2; k++)
{ // 두 선수가 번갈아 가며 진행
std::cout << p[k].getName() << ":<Enter>" << std::endl;
while (true) { // Enter 키 입력 대기
```

```
char ch;
std::cin.get(ch);
if (ch == '₩n')
break;
}
int n[3]; // 세 개의 랜덤 숫자를 저장하는 배열
std::cout << "₩t" << "₩t";
for (int i = 0; i < 3; i++)
{ n[i] = rand() % 3; // 0에서 2 사이의 랜덤 숫자 생성
std::cout << n[i] << "₩t";
}
if (n[0] == n[1] && n[0] == n[2])
{ // 세 숫자가 모두 동일한지 확인
std::cout << p[k].getName() << "님 승리!!" << std::endl; end = 1; // 게임 종료 설정
break;
else
std::cout << "아쉽군요!" << std::endl; // 숫자가 일치하지 않을 경우
}
if (end == 1) // 승자가 결정되면 반복 종료
break;
}
}
}
```

Main.cpp 코드

```
#include "GamblingGame.h"
// 프로그램의 시작점, 게임 객체 생성 및 실행
int main()
{
GamblingGame myGame; // GamblingGame 객체 생성
myGame.play(); // 게임 시작
return 0;
}
```