객체지향 프로그래밍 2차 과제

문제

갬블링 게임을 만들어보자. 두 사람이 게임을 진행하며, 선수의 이름을 초기에 입력 받는다. 선수가 번갈아 자신의 차례에서 <Enter> 키를 치면 랜덤한 3개의 수가 생성 되고 모두 동일한 수가 나오면 게임에서 이기게 된다. 숫자의 범위가 너무 크면 3개의 숫자가 일치할 가능성이 낮아 숫자의 범위를 0~2로 제한한다. 랜덤 정수 생성은 문제
3번의 힌트를 참고하라. 선수는 Player 클래스로 작성하고, 2명의 선수는 배열로 구성하라. 그리고 게임은GamblingGame 클래스로 작성하라

소스 수행 결과 화면 캡쳐

```
첫번째 선수 이름을 입력하세요: moonsu
두번째 선수 이름을 입력하세요: gain
***** 갬블링 게임을 시작합니다. *****
첫번째 선수 이름 >> moonsu
두번째 선수 이름 >> gain
moonsu:<Enter> 키 입력

moonsu: 1 0 0
아쉽군요!
moonsu: 0
gain: 0
gain:<Enter> 키 입력

gain: 2 2 0
아쉽군요!
moonsu: 0
gain: 0
moonsu:<Enter> 키 입력
```

```
moonsu: 1 2 2
아쉽군요!
moonsu: 0
gain: 0
gain:<Enter> 키 입력

gain: 1 1 1
gain 승리!!
Program ended with exit code: 0
```

소스 구현 설명

문제 정의

두 명의 플레이어가 번갈아 가며 랜덤 숫자를 생성하고, 모든 숫자가 동일할 경우 승리하는 것을 구현합니다.

플레이어의 이름을 입력받고, 각 플레이어가 자신의 차례에 `<Enter>` 키를 누르면 0부터 2까지의 랜덤 숫자 3개를 생성하고, 동일한 숫자가 모두 생성되면 해당 플레이어가 승리하고, 게임은 종료되는 문제입니다.

문제 해결 방법

Player

- 각 플레이어의 이름을 저장하는 클래스를 설계합니다.

GamblingGame

- 게임의 로직을 구현하고, 두 명의 플레이어가 번갈아 가면서 게임을 진행할 수 있도록 합니다.
- 랜덤 숫자 생성 및 승리 조건을 처리해서, 승리할 경우 승리 메시지를 출력하고 게임을 종료합니다.

랜덤 숫자 생성 함수

- 0~2 사이의 숫자를 랜덤하게 3개 생성하여 그 결과가 모두 동일하면 승리로 처리합니다.

게임 진행

- 두 명의 플레이어가 번갈아 가며 랜덤 숫자를 생성하는 과정이 게임이 종료될 때까지 계속 반복됩니다.

알고리즘 설명

Player.h

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

class Player {
private:
    string name; // 플레이어 이름

public:
    Player(const string& playerName); // 생성자, 플레이어 이름을 인자로 받는다.
    string getName() const; // 플레이어 이름을 반환
};

#endif
```

플레이어 이름을 저장하는 멤버 변수와, 플레이어 객체를 생성하고 이름을 반환하는 메서드를 정의했습니다.

Player.cpp

```
#include "Player.h"

Player::Player(const string& playerName) : name(playerName) {} // 플레이어 클래스 생성자 정의

string Player::getName() const { // getName 메서드 정의
    return name;
}
```

플레이어의 이름을 설정하고 반환하는 기능을 처리했습니다.

GamblingGame.h

```
#ifndef GAMBLINGGAME_H
#define GAMBLINGGAME_H

#include "Player.h"

class GamblingGame {
private:
    Player players[2]; // 플레이어 인원 수
    int currentPlayerIndex; // 현재 플레이어의 Index넘버
    int scores[2]; // 각 플레이어 점수

public:
    GamblingGame(const string& player1Name, const string& player2Name); // 생성자, 플레이어
    void play(); // 게임 시작
};

#endif
```

게임 구조를 정의하고, 플레이어와 관련된 정보를 선언했습니다.

GamblingGame.cpp

```
#include <iostream>
#include <cstdlib>
#include <ctime>
#include "GamblingGame.h"
using namespace std;

GamblingGame::GamblingGame(const string& player1Name, const string& player2Name) // 생성자
    : players{ Player(player1Name), Player(player2Name) }, currentPlayerIndex(0) {
    srand(static_cast<unsigned int>(time(0))); // 랜덤 시드 초기화
    scores[0] = 0; // 첫 번째 플레이어 점수
    scores[1] = 0; // 두 번째 플레이어 점수
}

void GamblingGame::play() { // 게임 메서드
    cout << "***** 갬블링 게임을 시작합니다. *****" << endl;
    cout << "첫번째 선수 이름 >> " << players[0].getName() << endl;
    cout << "두번째 선수 이름 >> " << players[1].getName() << endl;

    while (true) {
        cout << players[currentPlayerIndex].getName() << ":<Enter> 키 입력" << endl; // 지
        cin.ignore(); // 엔터키 기다리기

        // 랜덤숫자 생성
        int num1 = rand() % 3;
        int num2 = rand() % 3;
        int num3 = rand() % 3;

        cout << players[currentPlayerIndex].getName() << ": " << num1 << " " << num2 << "

        // 숫자가 같은 지 확인
        if (num1 == num2 && num2 == num3) {
            scores[currentPlayerIndex]++; // 점수 증가
            cout << players[currentPlayerIndex].getName() << " 승리!!" << endl;
            break; // 게임 종료
        } else {
            cout << "아쉽군요!" << endl;
        }

        // 현재 점수 출력
        cout << players[0].getName() << ": " << scores[0] << endl;
        cout << players[1].getName() << ": " << scores[1] << endl;


        currentPlayerIndex = (currentPlayerIndex + 1) % 2; // 다음 플레이어로
    }
}
```

게임의 전반적인 로직을 처리했습니다.

플레이어의 차례 관리, 랜덤 숫자를 생성하는 기능 등을 구현했습니다.

main.cpp

```
#include <iostream>
#include "GamblingGame.h"
using namespace std;

int main() {
    string player1Name, player2Name;
```

```
    // 플레이어 이름 입력
    cout << "첫번째 선수 이름을 입력하세요: ";
    getline(cin, player1Name);
    cout << "두번째 선수 이름을 입력하세요: ";
    getline(cin, player2Name);

    GamblingGame game(player1Name, player2Name); // 게임 시작
    game.play(); // 게임 실행

    return 0; // 소멸
}
```

흐름 설명

플레이어 이름 입력

- 시작되면, 두 명의 플레이어의 이름을 차례대로 입력받습니다.
- 각 플레이어의 이름은 `Player` 객체를 통해 저장됩니다.

게임 시작

- `GamblingGame` 클래스의 `play()` 메서드를 호출하면서 게임이 시작됩니다.
- 첫 번째 플레이어부터 차례가 시작됩니다.
- 현재 차례인 플레이어는 Enter 키를 눌러 자신의 차례를 진행합니다.

랜덤 숫자 생성

- Enter 키가 눌리면 0~2 사이의 랜덤 숫자 3개가 생성됩니다.
- 생성된 숫자가 출력되고, 3개의 숫자가 동일한지 확인합니다.

승리 조건 확인

- 3개의 숫자가 동일하다면, 현재 차례의 플레이어가 승리하고, 승리 메시지를 출력하며 게임이 종료됩니다.
- 그렇지 않으면 "아쉽군요!" 메시지를 출력하고, 다음 플레이어에게 차례가 넘어갑니다.

게임 종료

- 승리자가 결정되면 게임이 종료되고, 프로그램이 끝납니다.