

# 소스 파일 구현 설명

202304148 조하린

## ▶ 갬블링 게임

두 사람이 게임을 진행하며, 선수의 이름을 초기에 입력 받는다. 선수가 번갈아 자신의 차례에서 <Enter> 키를 치면 랜덤한 3개의 수가 생성되고 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 5번의 힌트(랜덤 정수)를 참고하라. 선수는 Player 클래스로 작성하고, 2명의 선수는 배열로 구성하라. 그리고 게임은 GamblingGame 클래스로 작성하라.

```
***** 갬블링 게임을 시작합니다. *****
첫번째 선수 이름>>수연이
두번째 선수 이름>>제갈이
수연이:<Enter>
      2      1      2      아쉽군요!
제갈이:<Enter>
      1      0      2      아쉽군요!
수연이:<Enter>
      2      2      1      아쉽군요!
제갈이:<Enter>
      2      0      2      아쉽군요!
수연이:<Enter>
      0      0      0      수연이님 승리!!
```

## 문제 정의

랜덤한 3개의 수가 모두 동일한 수로 나오면 게임에서 이기게 되는 겜블링 게임이다. 키보드로부터 두 선수의 이름을 모두 입력받고 선수가 번갈아 가며 자신의 차례에서 <Enter> 키를 치면 0~2의 랜덤한 수 3개와 결과가 화면에 출력된다.

1. 클래스 : 선수 각각의 이름 정보는 Player 클래스로 작성하여 변수 name을 멤버로 가지고 setName(), getName() 멤버 함수를 구현한다. 겜블링 게임의 전반적인 운영은 GamblingGame 클래스로 작성하여 Player 타입의 배열 변수 player를 멤버로 가지고 GamblingGame(), setPlayer(), playGame() 멤버 함수를 구현한다.
2. 입력 : 키보드로부터 두 명의 player 이름을 모두 입력 받고, 첫 번째 선수부터 차례로 <Enter> 키를 치면 게임이 진행된다.  
(단, <Enter> 키를 읽는 코드가 있는 것은 아니다.)
3. 무한 루프 : 승부가 나기 전까지 프로그램은 선수의 차례를 순서대로 바꿔가며 무한 루프를 돌아 게임을 계속 진행하게 된다. 승부가 결정 날 경우 승부 멘트와 함께 break에 걸려 while문을 빠져나온다.
4. 랜덤 정수 : 랜덤 정수를 발생시키기 위해서 <cstdlib>와 <ctime> 헤더 파일을 include 한 뒤, 시작할 때마다 랜덤수 발생을 위한 seed를 설정하고, rand() 함수를 통해 0에서 RAND\_MAX(32767) 사이의 랜덤한 정수를 발생시킨다.

## 파일에 대한 설명

### (1) player.h

Player 클래스 정의,  
멤버 변수: name,  
멤버 함수: setName(), getName()

### (2) gamblingGame.h

GamblingGame 클래스 정의,  
멤버 변수: Player player[2].  
멤버 함수: GamblingGame(), setGame(), playGame()

### (3) Player.cpp

Player 클래스 멤버 함수 구현: setName(), getName()

### (4) GamblingGame.cpp

GamblingGame 클래스 멤버 함수 구현: setPlayer(), playGame()

### (5) main.cpp

프로그램 시작  
GamblingGame 객체 생성  
함수 호출: setPlayer(), playGame()

## 문제 해결 방법 (아이디어)

### 1. 클래스

- 선수 각각의 이름 정보 : Player 클래스  
선수의 이름을 저장하는 변수 name  
선수 이름을 설정하는 setName() 함수  
선수 이름을 반환하는 getName() 함수
- 겜블링 게임의 전반적인 운영 : GamblingGame 클래스  
두 선수의 정보를 저장하는 배열 변수 Player player[2]  
생성자 GamblingGame()  
선수의 이름 입력받고 저장하는 setPlayer() 함수  
게임을 실행하는 함수 playGame()

### 2. 함수

- setName(string name): 선수 이름 name 변수에 저장
- getName(): 저장된 선수 name 변수 반환
- GamblingGame(): GamblingGame 객체 생성될 때 호출되는 생성자
- setPlayer(): 키보드로부터 두 선수 이름 입력받고, Player 클래스의 객체 배열 player에 name 변수에 저장
- playGame(): 두 선수의 순서 차례대로 바꾸고 각 턴마다 0~2인 랜덤수 생성하고 랜덤수 3개가 모두 같은지 비교 후 결과 출력
- main(): GamblingGame 객체 생성, setPlayer(), playGame() 함수 호출

### 3. 배열

- Player player[2]: 두 선수의 Player 객체의 배열
- int rint[3]: 랜덤수 저장위한 배열

### 3. 키보드 입력

- 키보드로부터 두 선수의 이름을 입력받고 Player 객체에 저장

### 4. this 포인터

- 현재 객체를 가리키는 포인터
- 함수의 매개변수가 자기 자신에 할당되는 경우 사용
- this->name은 현재 객체의 name 변수, name은 함수의 매개변수

## 5. 랜덤 정수

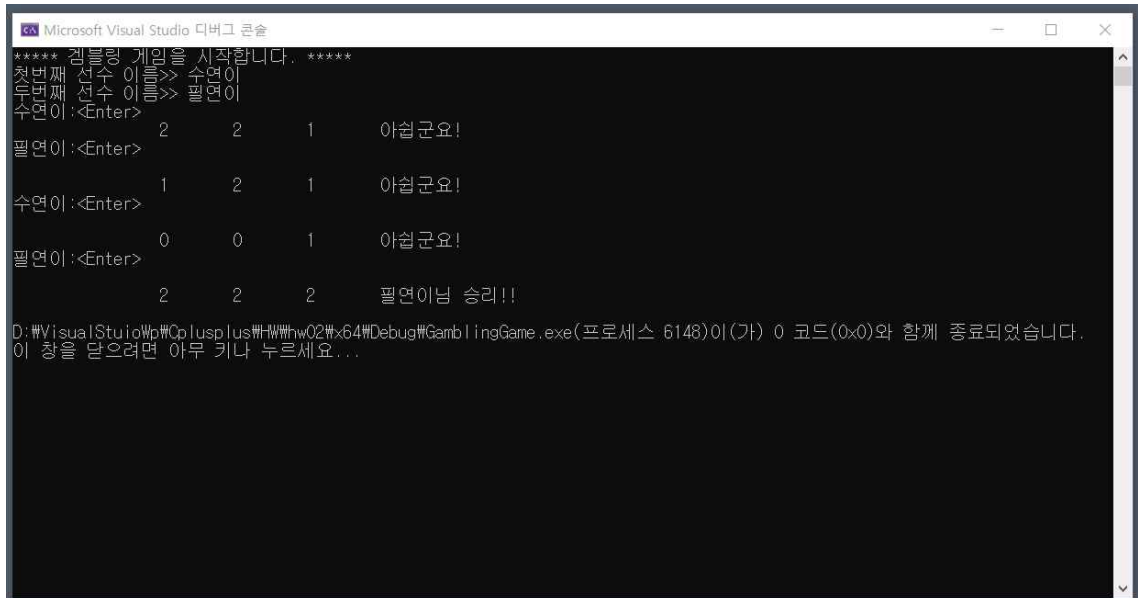
- `<cstdlib>`와 `<ctime>` 헤더 파일을 include
- `srand((unsigned)time(0));`  
시작할 때마다 랜덤수 발생을 위한 seed를 설정
- `rand()` 함수  
0에서 `RAND_MAX(32767)` 사이의 랜덤한 정수를 발생

## 프로그램 순서

1. 게임 시작
2. 선수 이름 입력
3. 게임 루프 (while문)
4. 랜덤수 생성
5. 랜덤수 출력
6. 랜덤수 비교 if (`rint[0] == rint[1] && rint[0] == rint[2]`)
7. false - 아쉽군요! 출력 -> 게임 루프 반복 (다시 3으로)  
ture - 승리 메시지 출력
8. 게임 종료

## 아이디어 평가

1. `srand((unsigned)time(0));` 코드를 사용하지 않았을 경우



```
Microsoft Visual Studio 디버그 콘솔
**** 잭팟 게임 시작합니다. ****
첫번째 선수 이름>> 수연이
두번째 선수 이름>> 필연이
수연이 :<Enter>
필연이 :<Enter>
2 2 1 아쉽군요!
필연이 :<Enter>
1 2 1 아쉽군요!
수연이 :<Enter>
0 0 1 아쉽군요!
필연이 :<Enter>
2 2 2 필연이님 승리!!
D:\VisualStudio\Cpp\plus\hw02\Debug\GamblingGame.exe (프로세스 6148)이 (가) 0 코드(0x0)와 함께 종료되었습니다.
이 창을 닫으려면 아무 키나 누르세요...
```

랜덤 숫자의 패턴과 실행 결과가 위에 경우로만 한정적이다.

2. `this` 포인터

`this`를 사용하지 않다면 클래스의 멤버 변수와 메서드의 매개 변수 이름이 동일해서 두 개의 변수를 만들어야 하는데 `this`를 사용함으로써 코드의 가독성을 높이고 변수들의 혼동을 줄일 수 있었다.

3. 파일 분리

파일 분리를 통해 각 파일의 역할에 맞게 나눠져 코드의 가독성을 향상시킬 수 있었다. 또한 코드를 재사용할 수 있고, 각 클래스를 독립적으로 관리할 수 있게 되었으며 디버깅에 용이해졌다.

`player.h` : 선수 각각의 이름 정보 `Player` 클래스 정의

    멤버 변수: `name`, 멤버 함수: `setName()`, `getName()`

`gamblingGame.h` : 잭팟 게임의 전반적인 운영 `GamblingGame` 클래스 정의

    멤버 변수: `Player player[2]`.

    함수: `GamblingGame()`, `setGame()`, `playGame()`

`Player.cpp` : `Player` 클래스 멤버 함수 구현

`GamblingGame.cpp` : `GamblingGame` 클래스 멤버 함수 구현

`main.cpp` : `GamblingGame` 객체 생성, 함수 호출: `setPlayer()`, `playGame()`