

20101576 정재찬.

1. PlayerRLGL::act()함수 구현

```
29
30 bool PlayerRLGL::act()
31 {
32     int FearlessBonusDistance = 0;
33     if (fearlessness > possibility(random_engine)*100) // fearlessness가 높을수록 이 이벤트가 자주 발생하도록 설계함.
34     {
35         double rand_num = possibility(random_engine); // 확률을 하나 집는다.
36
37         if (rand_num < fallDownRate) // 0.1보다 작다면 → 10%의 확률, 넘어져서 죽는다.
38         {
39             if (this->playing == false) // 더이상 플레이 하고 있지 않음.
40             {
41                 return false; // 죽었다고 결론내리고 끝남.
42             }
43             else if (rand_num < 0.65) // 0.1을 제외한 0.75의 확률이 됨.
44                 FearlessBonusDistance = (agility + double(fearlessness * 0.01)); // 추가 거리 보너스를 얻는다.
45
46         current_distance += agility + rand() % 11 + FearlessBonusDistance; // 즉, 지나간 거리는 agility + [0:10] + fearlessbonusdistance이다.
47         if (current_distance >= 1000) // 걸기한 거리가 전체 거리를 넘는다면
48         {
49             printStatus();
50             std::cout << "safely escaped from the ground." << '\n';
51             this->playing = false; // 게임을 하고 있지 않게 되고
52             return true; // 생존으로 마무리.
53         }
54     }
55 }
56 }
```

2. RedLightGreenLight::play()함수 구현

```
36
37 void RedLightGreenLight::play()
38 {
39     printGameName(); // 게임 이름 출력.
40
41     for (int t = 0; t < turn; ++t) // 0~19까지 20번 반복한다.
42     {
43         std::cout << "[Turn #" << t+1 << ']' << '\n'; // 0~19로 1가 바꿔므로 +1을 해줌.
44         auto cursor = players.begin(); // 생존해 있는 플레이어들을 하나씩 돌아면서 act()를 수행할 때 그 플레이어를 추적해줄 커서.
45
46         while(cursor != players.end()) // 처음부터 끝까지 순회함.
47         {
48             if ((*cursor)->isPlaying() && !((*cursor)->act())) // 이 플레이어가 플레이 중이고, 한 턴을 실행한 뒤 죽었다면,
49             {
50                 (*cursor)->dyingMessage(); // 죽었다고 메시지를 보내고,
51                 delete (*cursor); // 이 커서가 가리키는 공간을 없앤다.
52                 cursor = players.erase(cursor); // 그리고, players리스트에서 죽은 이 플레이어를 제거함.
53             }
54             else // 플레이중이지 않다 : 죽었거나 이미 통과함.
55                 ++cursor; // 다음 플레이어로 넘어감.
56         }
57     }
58
59     std::cout << "[Game Over]" << std::endl;
60
61     auto player = players.begin();
62     while (player != players.end())
63     {
64         }
65 }
```

3. 나만의 게임 Tazza 구현.

게임클래스를 상속받은 Tazza클래스. 선언

```
58
59
60 //// 간단해진 것다 게임(타짜)
61 //// 적은 컴퓨터라고 하고, 랜덤으로 서로 화투패를 두개씩 뽑아서 맵과 꽃 으로만 승부를 보게 설계해봤습니다.
62 //// 셋다의 죽보처럼 맵(x * 100) → 알리(90) → 독사(80) → 구별(70) → 장사(60) → 세룡(40) → 꽃(0-9) 순서로 점수를 부여하고,
63 //// 같은 맵이라면 수가 더 큰 사람이 더 높은 점수를 맞도록 설계했습니다.
64 //// 만약 같은 점수라면 즉시 재경기를 진행하도록 했습니다.
65 class Tazza : public Game
66 {
67 public:
68     Tazza() : Game("Tazza Game") {} // 이름만 정해준다.
69     ~Tazza() {};
70     void join(Player* player); // 똑같이 살아있는 플레이어만 추가.
71     void play();
72 }
```

Player클래스를 상속받은 PlayerTazza 클래스 선언

```
41
42
43     class PlayerTazza : public Player
44     {
45         public:
46             PlayerTazza(const Player& player) : Player(player) {};
47             bool act(); // 가상 함수 오버라이드
48             void dyingMessage(); // 가상 함수 오버라이드
49     };
50
51 }
```

Tazza 밑의 멤버 함수들

```
124
125
126     void Tazza::join(Player* player) // 살아있는 플레이어를 받아옴.
127     {
128         players.push_back(new PlayerTazza(*player)); // 이번 게임의 참가자로 추가.
129     }
130
131     void Tazza::play()
132     {
133         printGameName(); // 게임 이름 출력
134
135         auto cursor = players.begin(); // 플레이어들 하나하나를 추적할 커서.
136         while (cursor != players.end()) // 플레이어들을 한 번씩만 순회함. (한번의 act로 생사를 결정할 것임.)
137         {
138             if (!(*cursor)->act()) // 게임에 플레이어가 지면 false이므로
139             {
140                 (*cursor)->dyingMessage(); // 죽었다고 띄우고
141                 delete (*cursor); // 이 플레이어 객체를 동적으로 저장한 공간을 제거.
142                 cursor = players.erase(cursor); // 생존자 리스트에서도 제거.
143             }
144             else
145                 ++cursor; // 살았다면 다음 플레이어로
146         }
147
148         printAlivePlayers(); // 이 게임을 한 뒤에 살아남은 플레이어들을 출력.
149
150         std::cout << players.size() << " players are alive." << std::endl << std::endl;
151     }
152 }
```

PlayerTazza 밑의 멤버 함수들.

```

128 Ebool PlayerTazza::act() // true면 플레이어가 이긴거, false면 플레이어가 진거.
129 {
130     std::pair<int, int> enemyCard = { card_num(random_engine), card_num(random_engine) }; // 1~10사이의 카드를 두장 뽑는다.
131     std::pair<int, int> playerCard = { card_num(random_engine), card_num(random_engine) }; // 1~10사이의 카드를 두장 뽑는다.
132     int enemyPoint;
133     int playerPoint;
134
135     // 적(컴퓨터)의 점수
136     if (enemyCard.first == enemyCard.second)
137         enemyPoint = enemyCard.first * 100;
138     else if (enemyCard.first + enemyCard.second == 3) // 3이 되는건 1,2 밖에 없음. 알리
139         enemyPoint = 90;
140     else if ((enemyCard.first == 1 && enemyCard.second == 4) || (enemyCard.first == 4 && enemyCard.second == 1)) // 독사
141         enemyPoint = 80;
142     else if ((enemyCard.first == 1 && enemyCard.second == 9) || (enemyCard.first == 9 && enemyCard.second == 1)) // 구엔
143         enemyPoint = 70;
144     else if ((enemyCard.first == 1 && enemyCard.second == 10) || (enemyCard.first == 10 && enemyCard.second == 1)) // 장별
145         enemyPoint = 60;
146     else if ((enemyCard.first == 4 && enemyCard.second == 10) || (enemyCard.first == 10 && enemyCard.second == 4)) // 장사
147         enemyPoint = 50;
148     else if ((enemyCard.first == 6 && enemyCard.second == 4) || (enemyCard.first == 4 && enemyCard.second == 6)) // 세록
149         enemyPoint = 40;
150     else
151         enemyPoint = (enemyCard.first + enemyCard.second) % 10; // 0~9사이의 수를 갖는다.
152
153
154     // 플레이어 점수
155     if (playerCard.first == playerCard.second)
156         playerPoint = playerCard.first * 100;
157     else if (playerCard.first + playerCard.second == 3) // 3이 되는건 1,2 밖에 없음. 알리
158         playerPoint = 90;
159     else if ((playerCard.first == 1 && playerCard.second == 4) || (playerCard.first == 4 && playerCard.second == 1)) // 독사
160         playerPoint = 80;
161     else if ((playerCard.first == 1 && playerCard.second == 9) || (playerCard.first == 9 && playerCard.second == 1)) // 구엔
162         playerPoint = 70;
163     else if ((playerCard.first == 1 && playerCard.second == 10) || (playerCard.first == 10 && playerCard.second == 1)) // 장별
164         playerPoint = 60;
165     else if ((playerCard.first == 4 && playerCard.second == 10) || (playerCard.first == 10 && playerCard.second == 4)) // 장사
166         playerPoint = 50;
167     else if ((playerCard.first == 6 && playerCard.second == 4) || (playerCard.first == 4 && playerCard.second == 6)) // 세록
168         playerPoint = 40;
169     else
170         playerPoint = (playerCard.first + playerCard.second) % 10; // 0~9사이의 수를 갖는다.
171

```

```

148     enemyPoint = 50;
149     else if ((enemyCard.first == 6 && enemyCard.second == 4) || (enemyCard.first == 4 && enemyCard.second == 6)) // 세록
150         enemyPoint = 40;
151     else
152         enemyPoint = (enemyCard.first + enemyCard.second) % 10; // 0~9사이의 수를 갖는다.
153
154
155     // 플레이어 점수
156     if (playerCard.first == playerCard.second)
157         playerPoint = playerCard.first * 100;
158     else if (playerCard.first + playerCard.second == 3) // 3이 되는건 1,2 밖에 없음. 알리
159         playerPoint = 90;
160     else if ((playerCard.first == 1 && playerCard.second == 4) || (playerCard.first == 4 && playerCard.second == 1)) // 독사
161         playerPoint = 80;
162     else if ((playerCard.first == 1 && playerCard.second == 9) || (playerCard.first == 9 && playerCard.second == 1)) // 구엔
163         playerPoint = 70;
164     else if ((playerCard.first == 1 && playerCard.second == 10) || (playerCard.first == 10 && playerCard.second == 1)) // 장별
165         playerPoint = 60;
166     else if ((playerCard.first == 4 && playerCard.second == 10) || (playerCard.first == 10 && playerCard.second == 4)) // 장사
167         playerPoint = 50;
168     else if ((playerCard.first == 6 && playerCard.second == 4) || (playerCard.first == 4 && playerCard.second == 6)) // 세록
169         playerPoint = 40;
170     else
171         playerPoint = (playerCard.first + playerCard.second) % 10; // 0~9사이의 수를 갖는다.
172
173
174     if (playerPoint > enemyPoint)
175         return true;
176     else if (playerPoint < enemyPoint)
177         return false;
178     else
179     {
180         printStatus();
181         std::cout << "는 카드게임에서 비겼기에 재경기를 진행합니다." << '\n';
182         act();
183     }
184
185
186     Evoid PlayerTazza::dyingMessage()
187     {
188         printStatus();
189         std::cout << "가 카드 게임에서 졌습니다..." << std::endl;
190     }
191
192

```

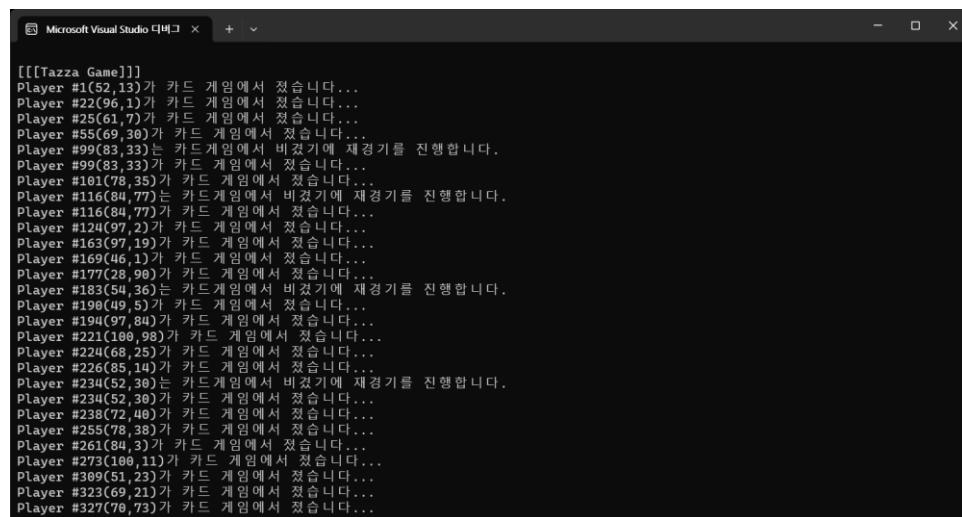
4. 타짜게임 설명.

- 이 게임은 영화 타짜에 나오는 '셧다'를 간단하게 구현을 해 보았습니다.
- 그 카드의 패는 1월 ~ 10월 까지의 상징이 그려져 있는 카드들이고, 그 월의 수를 이용해서 높고 낮음이 결정이 됩니다.

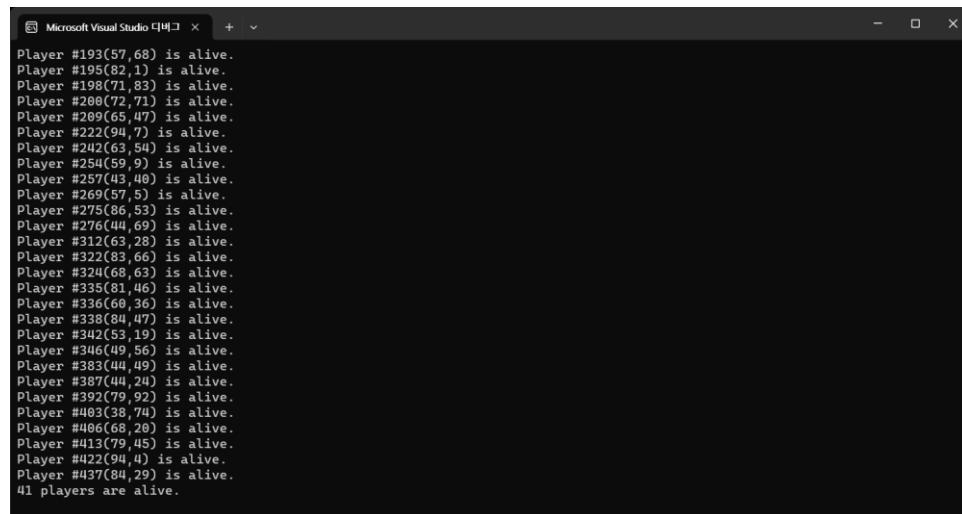
- 저는 여기서 플레이어와 상대(컴퓨터)가 2개의 카드를 뽑고(광이나 이런 것은 무시한 뒤) 구현할 수 있는 범위 내의 카드쌍들의 점수를 더 높은 단계의 순서대로 배치했습니다.
- 그리고 뽑은 카드에 맞는 점수를 플레이어와 상대(컴퓨터)에게 준 뒤 플레이어가 더 낮은 점수라면 패배(죽음), 더 높은 점수라면 승리(생존), 점수가 같다면 재경기를 하도록 설계했습니다.
- 재경기는 다른 플레이어가 모두 끝난 뒤 하는 게 아닌 게임이 끝나자마자 바로 다시 실행하게 했습니다.
- 이 타짜게임을 수행한 뒤 생존 확률은 50%라고 기대하고 있습니다.

5. 타짜게임의 출력 결과(일부)

- 출력 전체는 sample_output.txt파일을 참고해주시면 될 것 같습니다.



```
[[[Tazza Game]]]
Player #1(52,13)가 카드 게임에서 죽습니다...
Player #22(96,1)가 카드 게임에서 죽습니다...
Player #25(61,7)가 카드 게임에서 죽습니다...
Player #55(69,36)가 카드 게임에서 죽습니다...
Player #99(83,33)가 카드 게임에서 비겼기에 재경기를 진행합니다.
Player #99(83,33)가 카드 게임에서 죽습니다...
Player #101(78,35)가 카드 게임에서 죽습니다...
Player #116(84,77)는 카드 게임에서 비겼기에 재경기를 진행합니다.
Player #116(84,77)가 카드 게임에서 죽습니다...
Player #124(97,22)가 카드 게임에서 죽습니다...
Player #163(97,19)가 카드 게임에서 죽습니다...
Player #169(46,1)가 카드 게임에서 죽습니다...
Player #177(28,99)가 카드 게임에서 죽습니다...
Player #183(54,36)는 카드 게임에서 비겼기에 재경기를 진행합니다.
Player #190(49,5)가 카드 게임에서 죽습니다...
Player #194(97,84)가 카드 게임에서 죽습니다...
Player #221(100,98)가 카드 게임에서 죽습니다...
Player #224(68,25)가 카드 게임에서 죽습니다...
Player #226(85,14)가 카드 게임에서 죽습니다...
Player #234(52,39)는 카드 게임에서 비겼기에 재경기를 진행합니다.
Player #234(52,39)가 카드 게임에서 죽습니다...
Player #238(72,40)가 카드 게임에서 죽습니다...
Player #255(78,38)가 카드 게임에서 죽습니다...
Player #261(84,3)가 카드 게임에서 죽습니다...
Player #273(100,11)가 카드 게임에서 죽습니다...
Player #309(51,23)가 카드 게임에서 죽습니다...
Player #323(69,21)가 카드 게임에서 죽습니다...
Player #327(70,73)가 카드 게임에서 죽습니다...
```



```
Player #193(57,68) is alive.
Player #195(82,1) is alive.
Player #198(71,83) is alive.
Player #200(72,71) is alive.
Player #209(65,47) is alive.
Player #222(94,7) is alive.
Player #242(63,54) is alive.
Player #254(59,9) is alive.
Player #257(43,40) is alive.
Player #269(57,5) is alive.
Player #275(86,53) is alive.
Player #276(44,69) is alive.
Player #312(63,28) is alive.
Player #322(83,66) is alive.
Player #324(68,63) is alive.
Player #335(81,46) is alive.
Player #336(68,36) is alive.
Player #338(84,47) is alive.
Player #342(53,19) is alive.
Player #346(49,56) is alive.
Player #383(44,49) is alive.
Player #387(44,24) is alive.
Player #392(79,92) is alive.
Player #403(38,74) is alive.
Player #406(68,20) is alive.
Player #413(79,45) is alive.
Player #422(94,4) is alive.
Player #437(84,29) is alive.
41 players are alive.
```