< 시스템SW실습3 >

Project #1 – Poker

Seven Poker 게임을 진행하고 최종적으로 가장 많은 돈을 가진 사람이 승리합니다.

공통 rule

- Main 함수의 순서대로 진행
- AI의 class condition
  - User가 AI를 참조하는 경우
    - AI의 table 상태를 확인할 때 (AI 조합 정보가 이미 table에 정리되어 있음)
    - 최종 결과를 출력할 때
  - AI의 condition table 변수 정보
    - Name : 조합 번호
    - Num : 조합에서 가장 큰 숫자
      - Full house의 경우에는 3장인 카드의 번호
      - Straight는 가장 작은 숫자
    - Shape : 조합에 따라 모양 결정
      - Flush의 경우 해당 shape
      - 이외에 경우, 위의 num에 해당하는 카드의 shape
    - Pair : Two pair의 경우, 작은 숫자 pair에 해당하는 숫자
      - Two pair의 경우에만 사용할 필요 존재
    - 이외의 변수는 참조 필요 없음 (user가 필요한 경우에 사용하는 변수)

◆ 조합에 따라 num 혹은 shape이 필요하지 않은 경우, AI도 user의 정보를 일부만 참조

- Four card의 경우, shape이 무의미
- Royal straight flush의 경우, num이 무의미

◆ **AI도 user table의 name, num, shape, pair만을 참조합니다.**

- 조합

■ 같은 조합일 경우, 숫자 혹은 모양을 비교

◆ 높은 숫자일수록 상위 / Spade, Diamond, Heart, Clover순으로 상위

◆ 같은 조합일 경우, 가지고 있는 최대 숫자를 비교하고 최대 숫자가 같을 경우 최대 숫자의 모양을 비교

- 두 player가 동일하게 9, 3pair인 경우 9 pair 중에서 spade를 가진 사람이 승리
- Flush를 비교하는 경우, 최대 숫자를 비교하고 최대 숫자가 같을 경우에는 모양을 비교
- Straight를 비교하는 경우, 최대 숫자를 비교하고 최대 숫자가 같을 경우에는 모양을 비교

■ Straight 중, (A 2 3 4 5)의 조합은 배제하고 (10 J Q K A)의 조합은 사용

- Straight의 경우, condition의 num에는 가장 작은 숫자를 대입
- 처음 3장을 받은 후, 1장을 반드시 받고 100원을 적립 (1 set 필참)

■ 2 set 부터 die 가능

- 죽는 순서는 user가 가장 먼저

■ Player 모두가 죽는 경우는 배제

- 중간에 게임이 끝난 경우, 현재까지의 카드와 조합을 출력

유의사항

- 구현 방법
  - Poker.cpp만을 작성하고 필요에 따라서 poker.h의 주석 부분에 변수 추가 가능
- 판돈
  - Set 1, 2, 3, 4에서 진행할 때마다 100원, 200원, 400원, 800원씩 bank에 적립
    - Go로 결정한 경우 바로 적립
  - 결과 확인 후에 누적된 금액을 승자가 획득
  - Player의 돈이 –여도 계속 참여, 진행 가능
- User와 AI 공통으로, player의 죽음은 name 변수를 -1로 표시
  - AI vector에는 항상 3명의 AI가 들어있으며, AI의 name 변수가 -1일 경우, 현재 게임에서 죽은 상태로 인식

poker.h

- 모든 조합은 poker.h의 숫자대로 관리
- 각 문양을 define으로 정리
  - SPADE = 4 / DIAMOND = 3 / HEART = 2 / CLOVER = 1
- Class Card
  - 번호는 2~14 (2~A), 모양은 위의 define을 사용
- Class Condition
  - 조합의 상태를 표시
    - Name = 조합 번호
  - 정의 되어있는 변수들 외에 필요에 따라 추가 선언 및 사용 가능
- Class Player
  - User의 정보
    - Name은 0으로 고정, money는 시작할 때 10000원 지급

        - ◆ Mine, my_cards는 본인 시점 / table, my_table_cards는 타인 시점으로 사용
        - ◆ String table_cards는 출력을 위한 용도로 자유롭게 사용
    - ■ 정의 되어있는 변수들 외에 필요에 따라 추가 선언 및 사용 가능
- Class Player_AI
    - ■ AI의 정보
        - ◆ 기본 정보는 user와 마찬가지
        - ◆ User는 의사 결정을 위해 AI.table에 접근 가능
        - ◆ Class 내의 get_mine()을 통해서 AI의 전체 카드 열람 가능
            - ● 반드시 show_result() 내에서만 사용
- Class Poker
    - ■ Card deck 제공
    - ■ 판돈을 저장해 놓을 변수 bank 제공
        - ◆ 한 게임이 끝날 때까지의 누적 판돈이며, 매 게임마다 0으로 reset
    - ■ 게임을 위한 함수 제공

main.cpp

- Player(자신)와 AI 3명을 초기화
- 정상적으로 출력되었을 시, 게임 과정과 결과가 도출 (최종 카드, 돈)
- 전체 진행 상황
    - ■ 초기화 -> 카드 섞기 -> 3장씩 분배 -> Table 카드 상태 확인 -> 1장씩 분배 -> Table 카드 상태 확인 -> 게임 진행 의사 결정 -> 게임 완료 확인 -> 반복
    - ■ 게임 완료 확인은 player가 혼자 남은 경우를 확인

main_iter.cpp

- 게임을 여러 번 실행

- 각 player의 돈은 초기화하지 않으니 누적되도록 구현

- 결과를 간략하게 출력

poker.cpp

Shuffle_deck

- 0 ~ 51까지의 숫자를 random으로 획득 (♠ ◆ ♥ ♣ 순서)
  - Rand()에서 0을 받은 경우, ♠Ace 로 인식
  - Rand()에서 20을 받은 경우, ◆8 로 인식
  - AI는 Ace를 14번으로 인식 (2 ~ K : 2 ~ 13)
- 받는 순서대로 push_back() 함수로 deck에 넣고, back() 함수로 차례대로 사용

First_draw

- 3장 씩 받고, 뒤집어 놓을 top card를 결정
- 나눠주는 순서는 player 0부터 3장, player 1에게 3장... 순서
- First_draw_AI()에서는 각 AI가 알아서 3장 씩 획득

Show_table

- Table 상태를 출력 (자신의 카드만 출력)
- Show_table_AI()에서는 AI의 table card 출력

Verify_cards

- 자신의 카드 상태 확인
  - Table에 있는 자신의 카드 상태 update 필요 (Player.table)
    - 이후 AI가 Player.table을 참조하여 의사 결정
  - 실제 자신의 카드 상태 update 시점은 자유
- Verify_cards_AI()에서는 각 AI가 자신의 table card 상태를 update

Draw

- 카드를 1장씩 분배
  - Table에서 가장 높은 조합을 가진 사람부터 반시계 방향으로 분배
    - 0->1->2->3, 2->3->0->1과 같은 순서
    - AI의 조합을 확인하기 위해 AI.table 참조 가능
- Draw_AI()에서는 AI가 한 장의 card를 획득
  - Ex) AI 2번이 top player인 경우, draw_AI(AI2) / draw_AI(AI3) / user 알고리즘 / draw_AI(AI1) 순서로 함수 호출
  - Draw_AI() 함수가 남은 AI 수만큼 호출 필요

Go_or_die

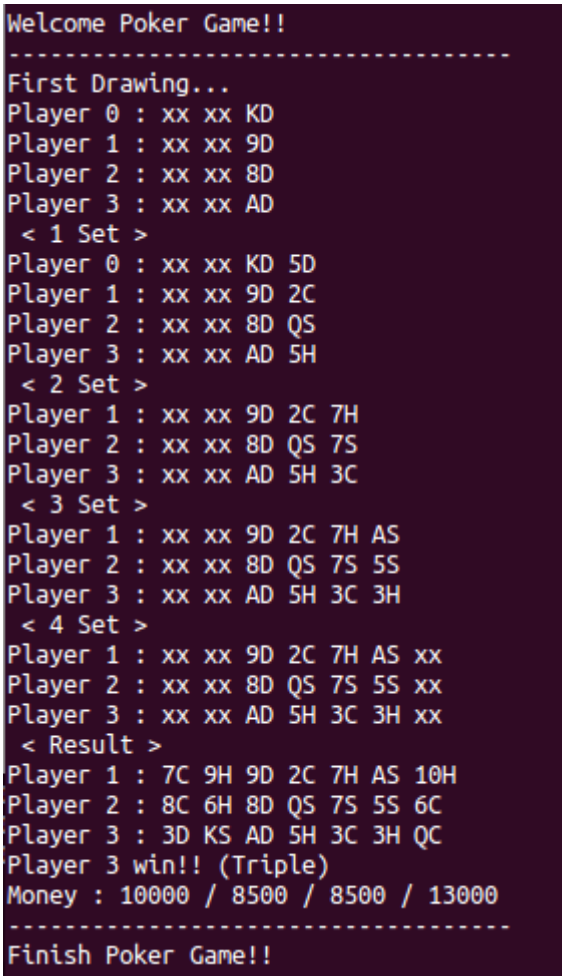
- User의 알고리즘에 따라 게임을 진행할지 말지 결정
- Go_or_die_AI()에서는 AI가 자신의 카드 상태와 table 상태를 비교하여 의사 결정

Show_result

- 최종 table 상태를 출력함과 동시에 결과 도출
  - 죽은 player는 출력하지 않음
  - 승자와 승자가 가지고 있던 조합을 출력
  - 승자를 판별하고 돈을 지급
    - 승자를 판별하기 위해 AI의 전체카드 열람 가능
      - Get_mine() 함수
  - 최종 출력은 모든 카드가 보이도록 출력
- Show_result_AI()에서는 남은 AI들의 카드를 전부 오픈

출력 결과

- Main.cpp



- Main_iter.cpp (3번 실행한 경우)

```
Welcome Poker Game!!
-----------------------------------
First Drawing...
 < Result >
Player 1 : 7C 9H 9D 2C 7H AS 10H
Player 2 : 8C 6H 8D QS 7S 5S 6C
Player 3 : 3D KS AD 5H 3C 3H QC
Player 3 win!! (Triple)
Money : 10000 / 8500 / 8500 / 13000
-----------------------------------
First Drawing...
 < Result >
Player 1 : 5S 3S QD 8S QC AS 2D
Player 2 : 6H 9S 10C 7S 9H 5D 7D
Player 2 win!! (Two Pair)
Money : 10000 / 7000 / 10700 / 12300
-----------------------------------
First Drawing...
 < Result >
Player 1 : 8H 9S AS 3S KD AD JS
Player 1 win!! (Pair)
Money : 10000 / 8400 / 10000 / 11600
-----------------------------------
Finish Poker Game!!
```

제출

- 작성한 코드에 주석 작성
  - ■ 가독성, 모든 경우가 포함되었는지 판단
- Poker.cpp / poker.h 파일만을 압축
- "학번.tar.gz" 형식으로 조교에게 제출
  - ■ 2016710580.tar.gz을 조교 메일로 전송 (두명 모두에게 보내주시길 바랍니다.)

g++-4.8 downgrade

"g++ --version"으로 현재 g++ version 확인 가능

sudo apt-get install g++-4.8

sudo apt-get install aptitude

sudo aptitude install g++-4.8

```
sudo update-alternatives --install /usr/bin/g++ g++ /usr/bin/g++-4.8 800

sudo update-alternatives --config g++
```