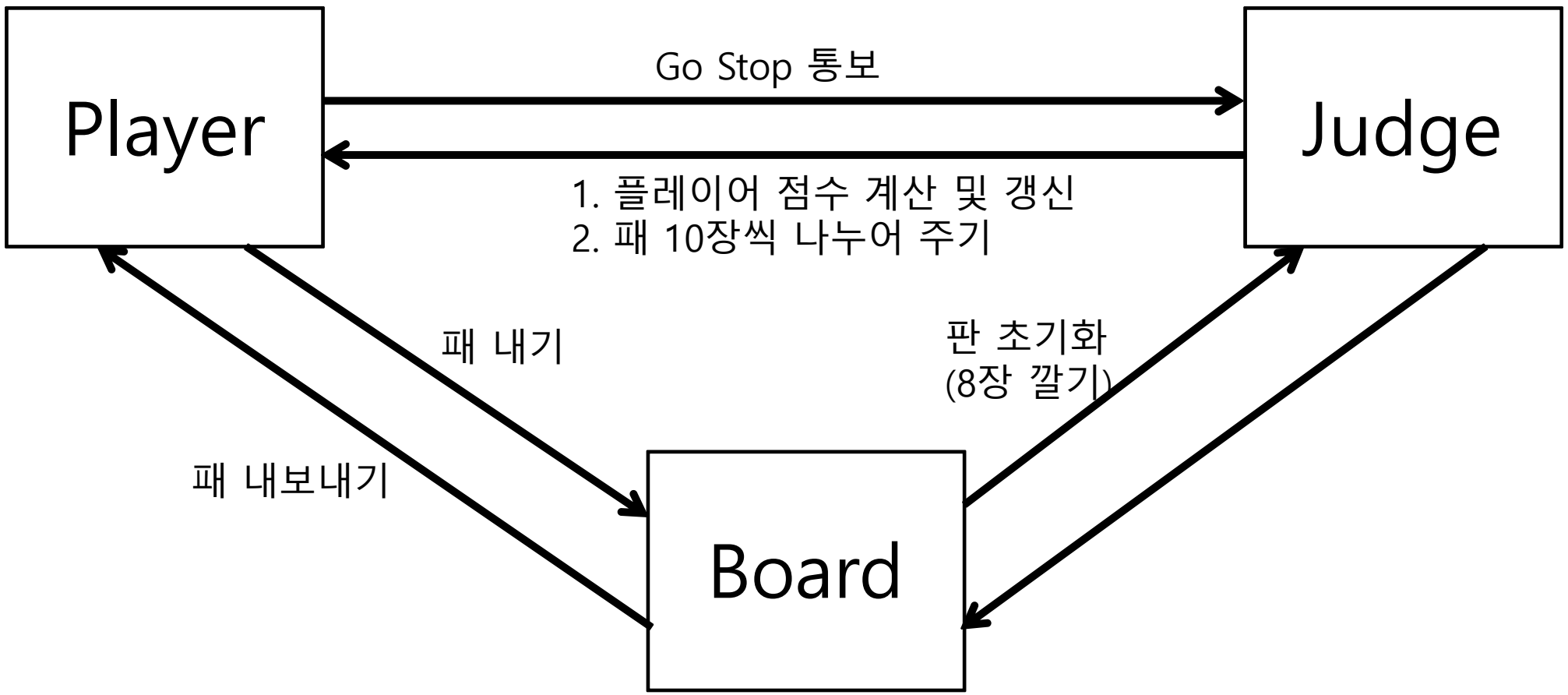
OOP20 term project

201402020 김승훈
201502104 임진현

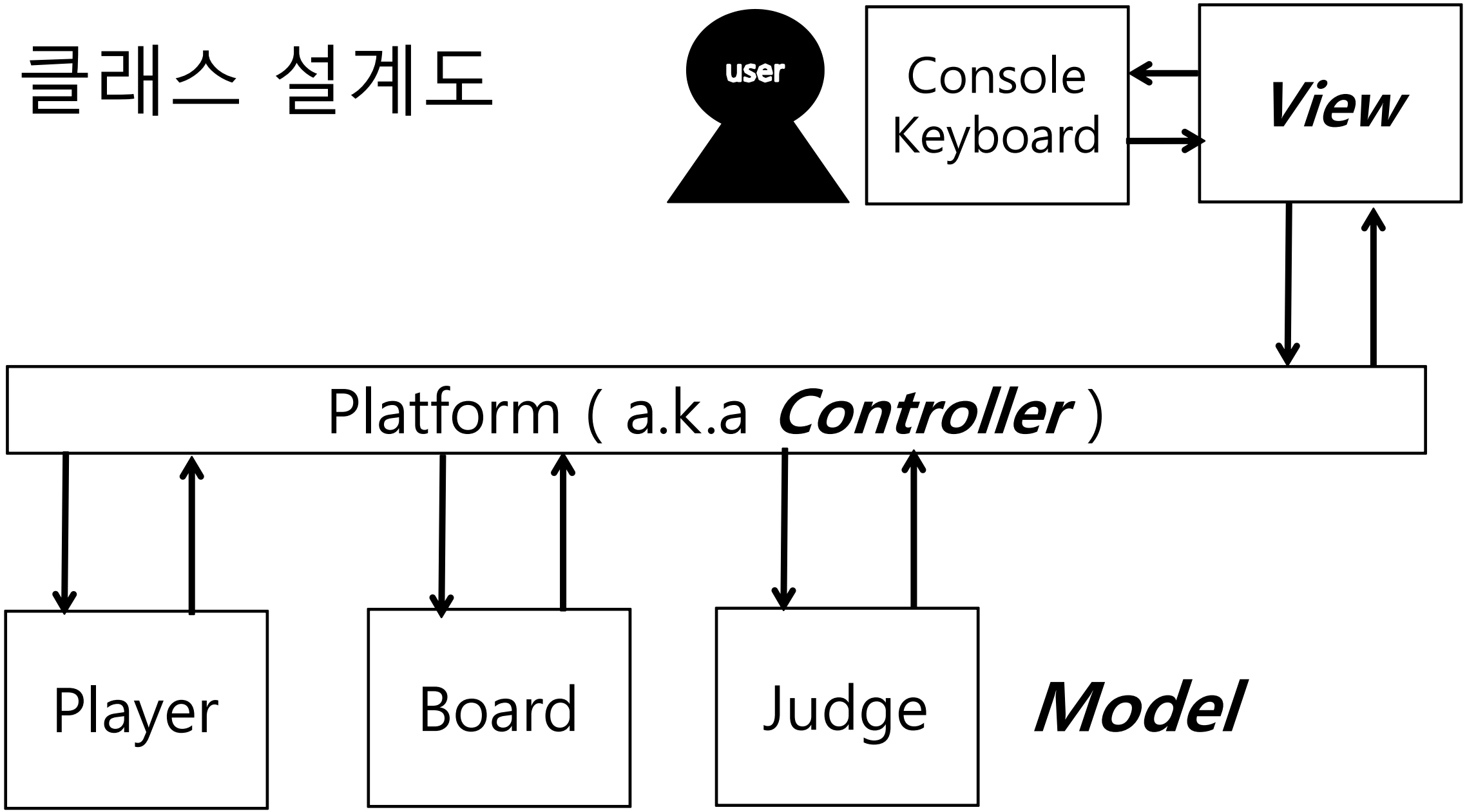
목차

- 클래스 설계도
- 게임 시나리오

클래스 설계도


Player
Judge
Board
Go Stop 통보
1. 플레이어 점수 계산 및 갱신
2. 패 10장씩 나누어 주기
패 내기
패 내보내기
판 초기화
(8장 깔기)

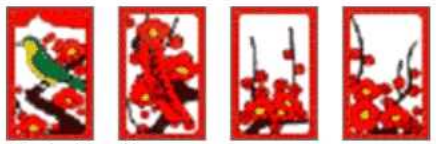
클래스 설계도


user
Console
Keyboard
View
Platform (a.k.a Controller)
Player
Board
Judge
Model

클래스 설계도(Card)

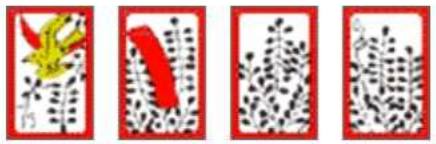
송학(1월)

매조(2월)



벚꽃(3월)

흑싸리(4월)



난초(5월)

모란(6월)

홍싸리(7월)

공산(8월)

국준(9월)

단풍(10월)

오동(11월)

비(12월)

1월 January (Jan.)

2월 February (Feb.)

3월 March (Mar.)

4월 April (Apr.)

5월 May (May)

6월 June (Jun.)

7월 July (Jul.)

8월 August (Aug.)

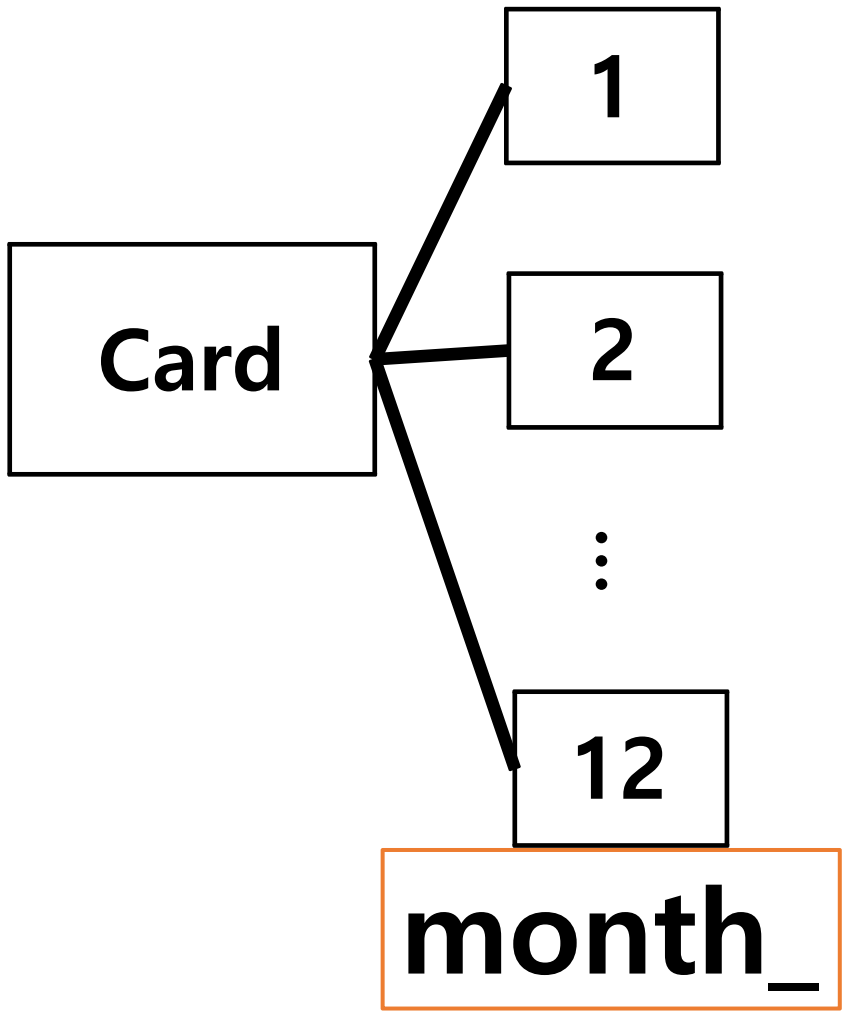
9월 September (Sep.)

10월 October (Oct.)

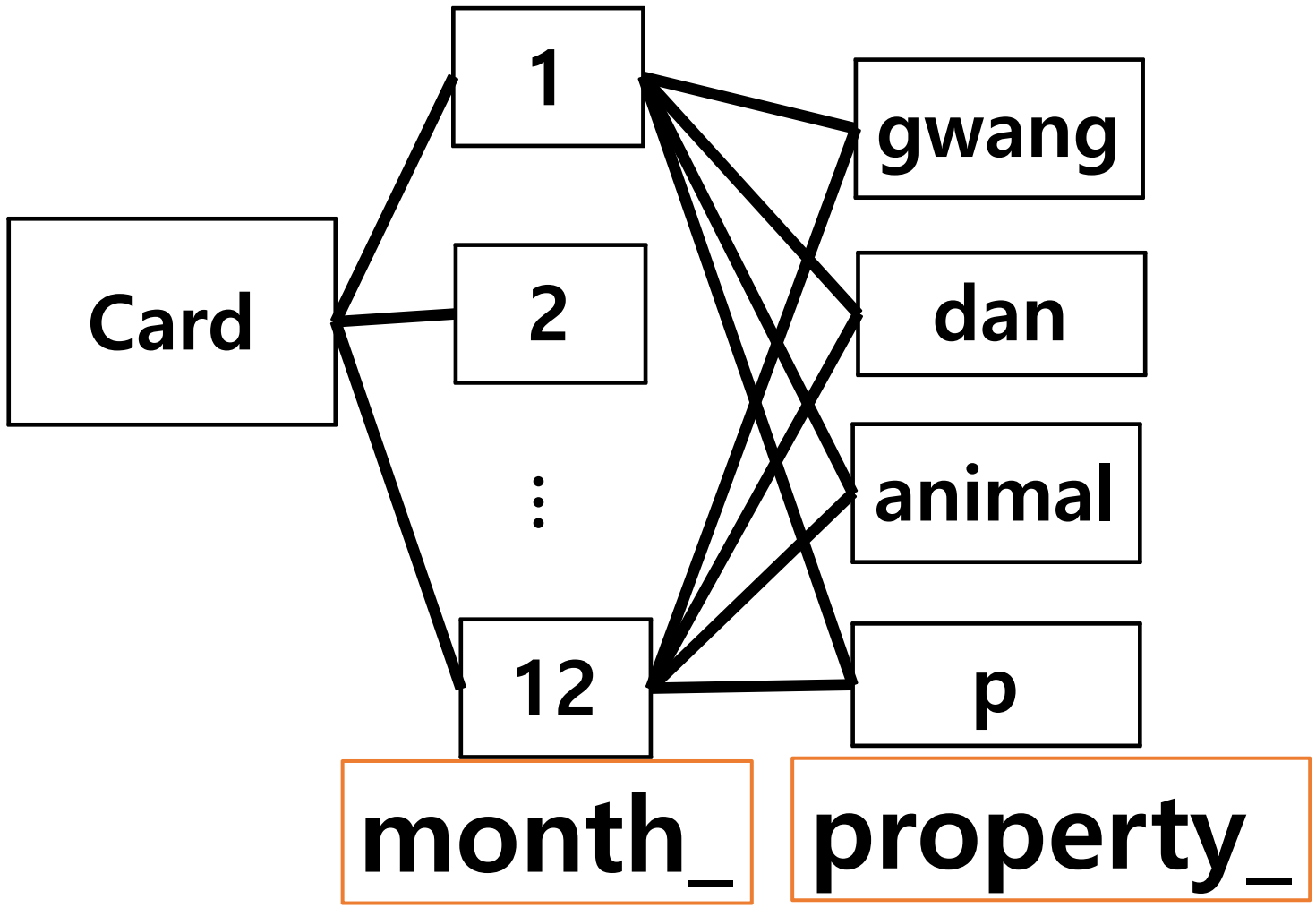
11월 November (Nov.)

12월 December (Dec.)

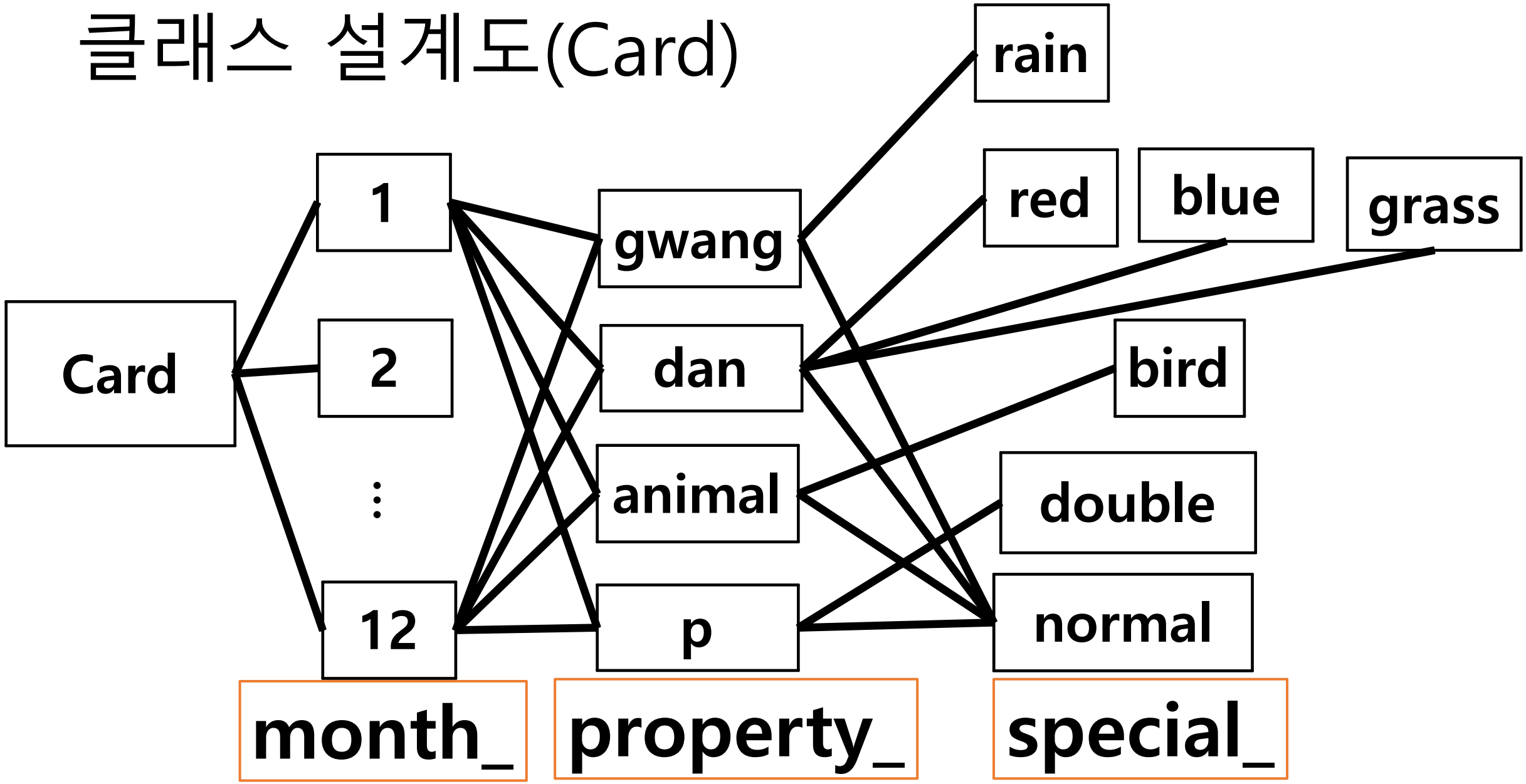
클래스 설계도(Card)



클래스 설계도(Card)



클래스 설계도(Card)

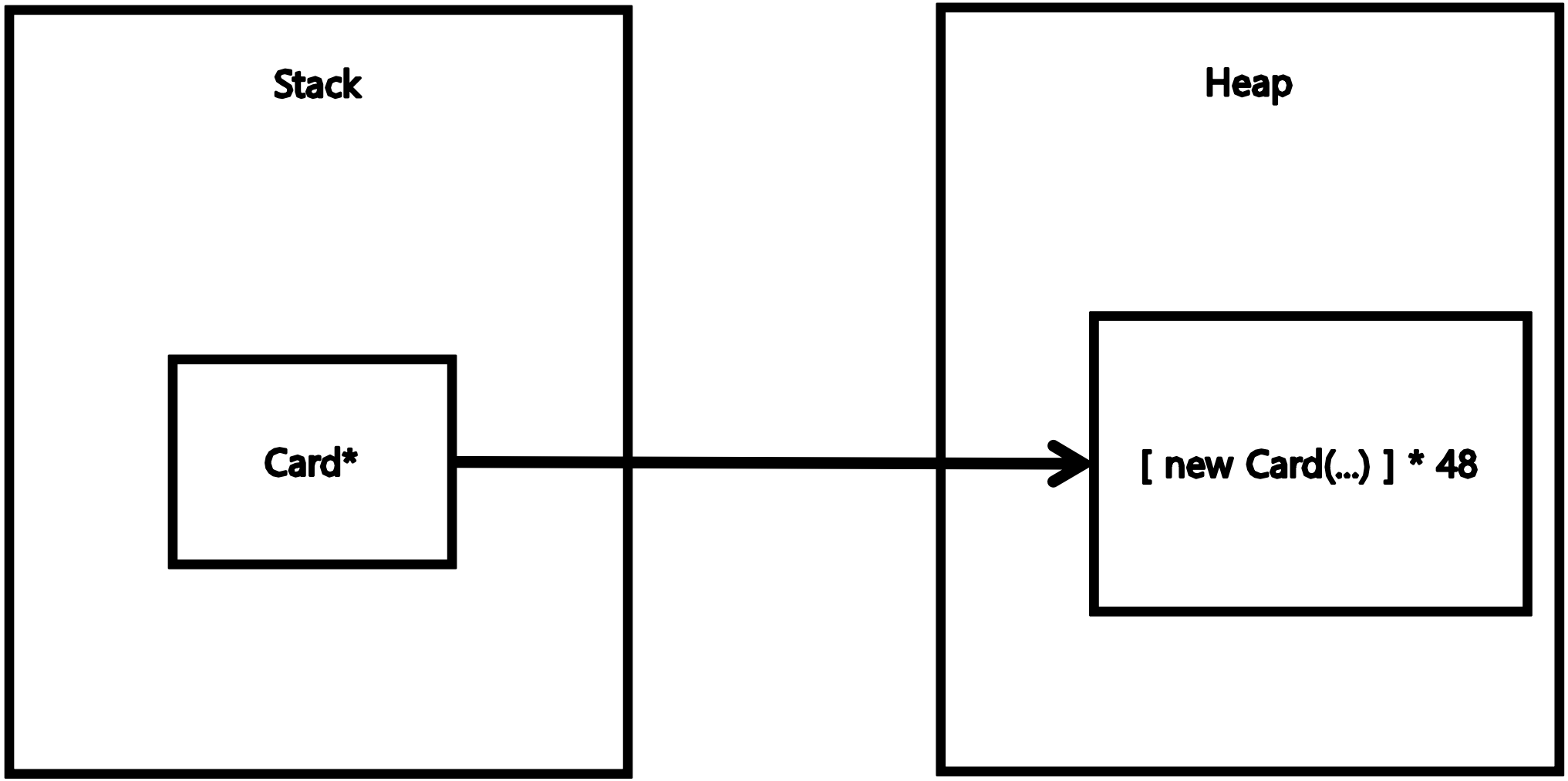

Card
1
2
12
gwang
dan
animal
p
rain
red
blue
grass
bird
double
normal
month_
property_
special_

클래스 설계도(Card)

Card
int month_ std::string property_ std::string special_
Card() Card(int month, std::string property, std::string special) std::string ToString() bool IsCardValid()

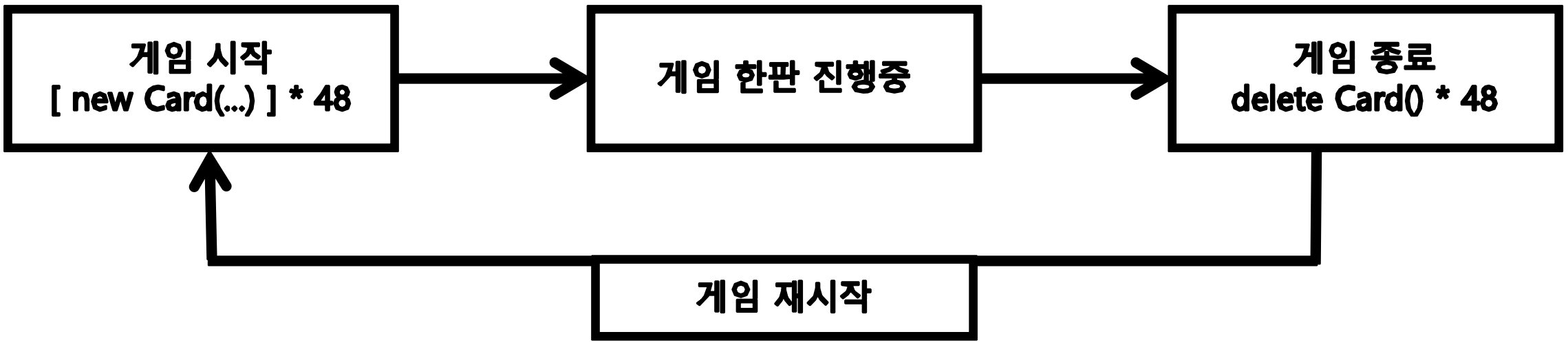
Ex) 12_gwang_rain
(month_property_special)

클래스 설계도(Card)

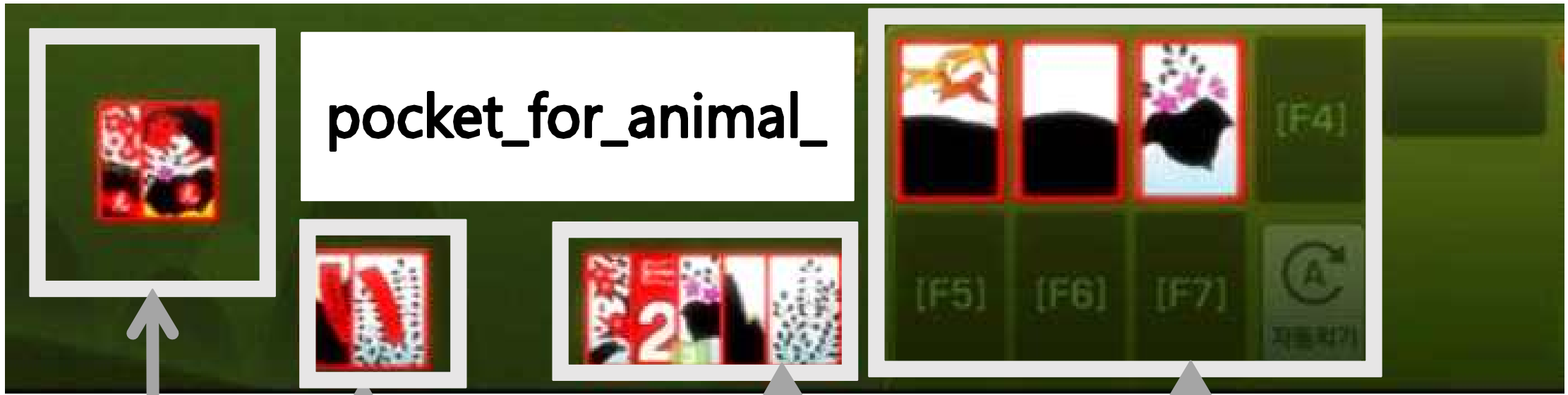


클래스 설계도(Card)


[new Card(...)] * 48
N 번의 게임
delete Card() * 48


게임 시작
[new Card(...)] * 48
게임 한판 진행중
게임 종료
delete Card() * 48
게임 재시작

클래스 설계도(Player)



pocket_for_gwang_

pocket_for_dan_

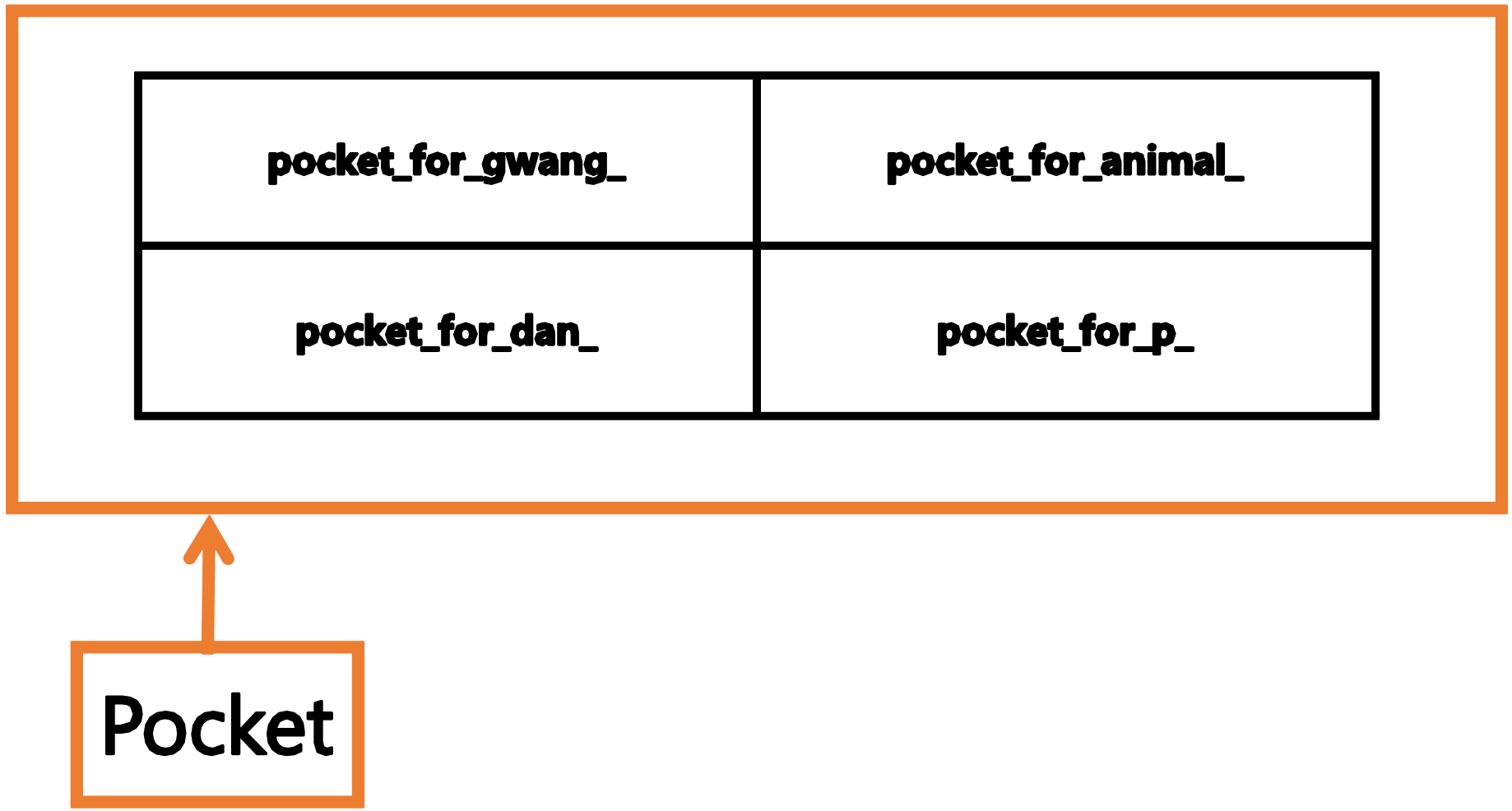
pocket_for_p_

available_cards_

잦은 선형 탐색
→ std::list<Card>

원하는 카드 즉각 제출
→ std::vector<Card>

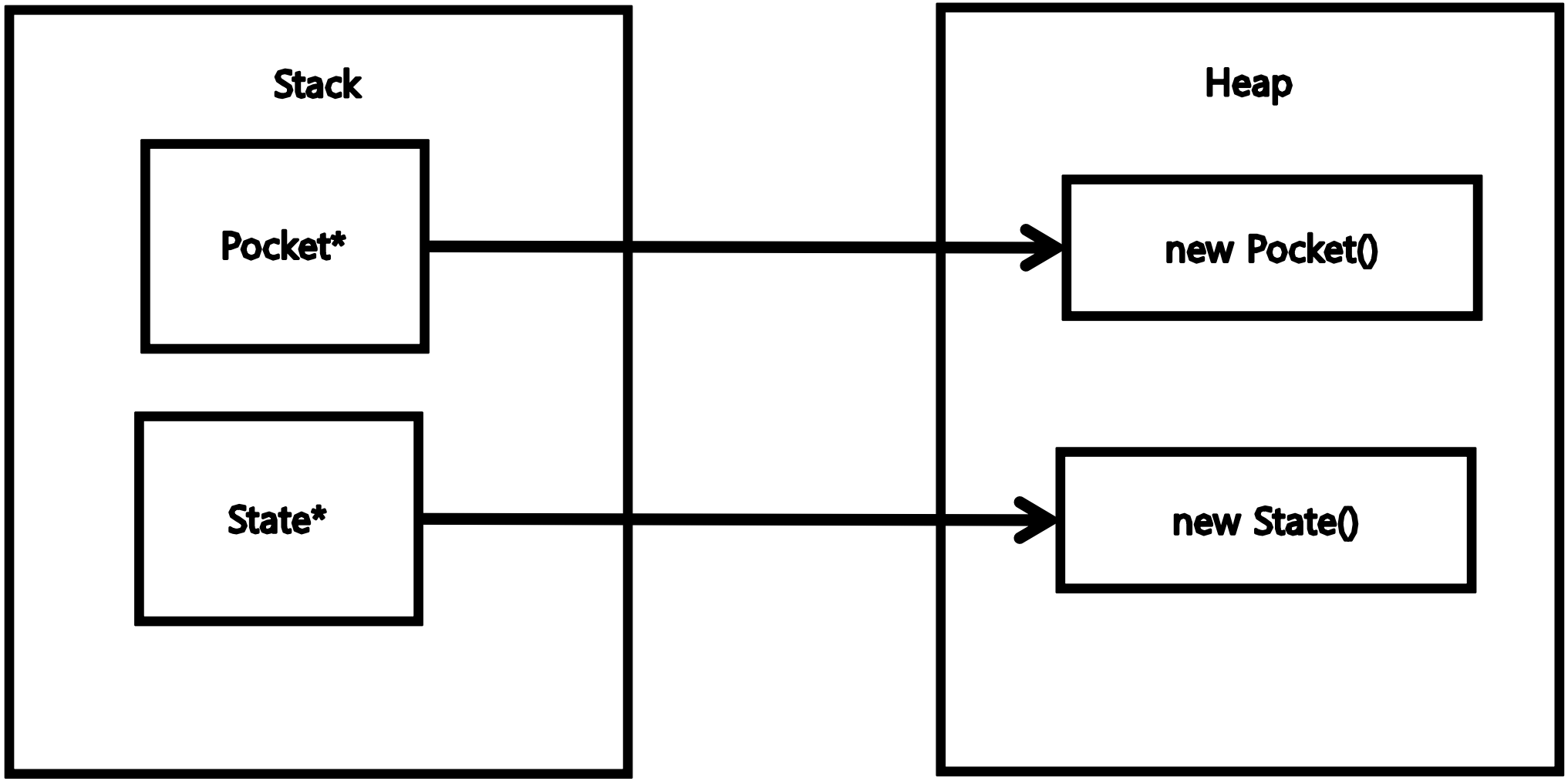
클래스 설계도(Pocket)



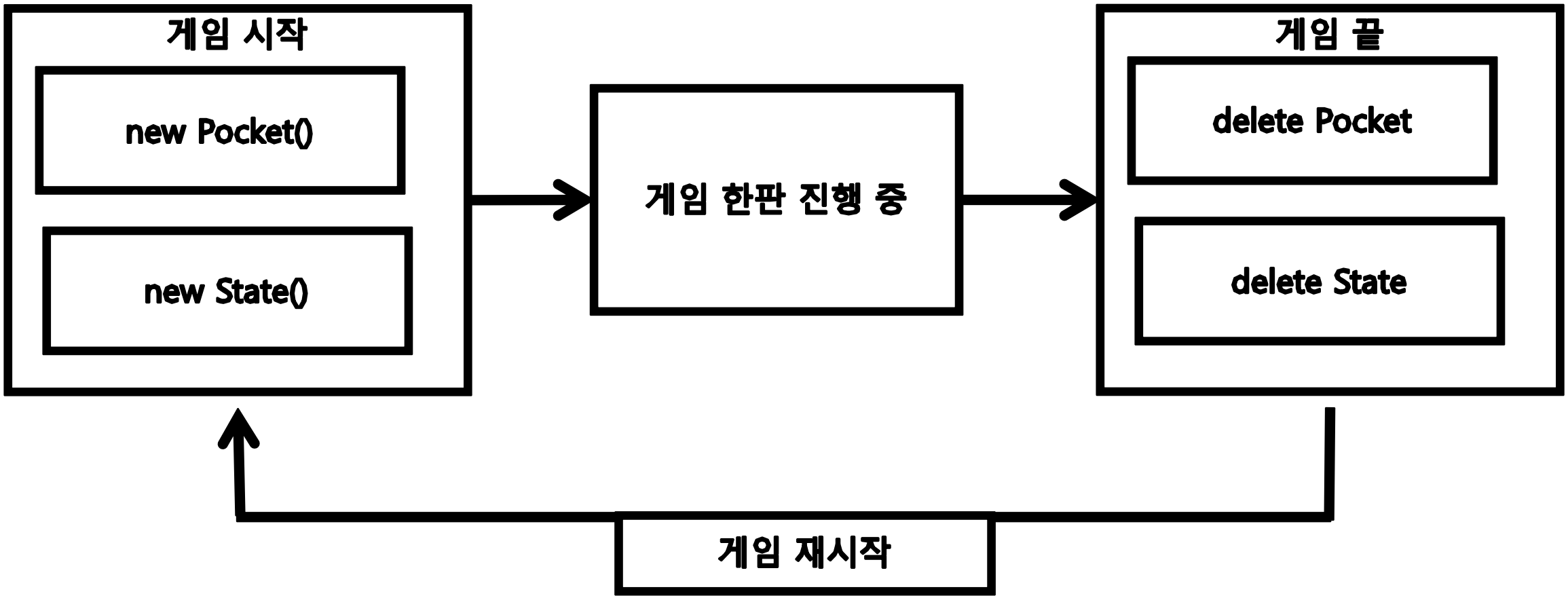
클래스 설계도(Player)

Player
std::vector<Card*>* available_cards_ Pocket* pockets_for_cards_ State* currrent_state_
Card* GetPIfPossible() void DeclareGo(int score_when_declare_go); const std::string ToStringForAvailableCards() const std::string ToStringForPockets() const Card* DropCard(int selected_card_order) void AddCard() void CleanPocketsAndResetScore() Getters & Setters

클래스 설계도(Player)



클래스 설계도(Player)



클래스 설계도(Pocket)

Pocket
std::list<Card*>* pocket_for_gwang_ std::list<Card*>* pocket_for_animals_ std::list<Card*>* pocket_for_dan_ std::list<Card*>* pocket_for_p_
const std::string ToStringForPockets() void AddCard(Card*) Card* GetPIfPossible() const std::string ToStringForGwangPocket() | const std::string ToStringForAnimalsPocket() const std::string ToStringForDanPockets() | const std::string ToStringForPPockets() void AddGwang(Card* a_card) | void AddAnimal(Card* a_card) void AddDan(Card* a_card) | void AddP(Card* a_card) Getters & Setters

클래스 설계도(State)

State
int number_of_go_ int std_score_for_win_
void ChangeStateForGo(int score_when_declare_go); Getters & Setters

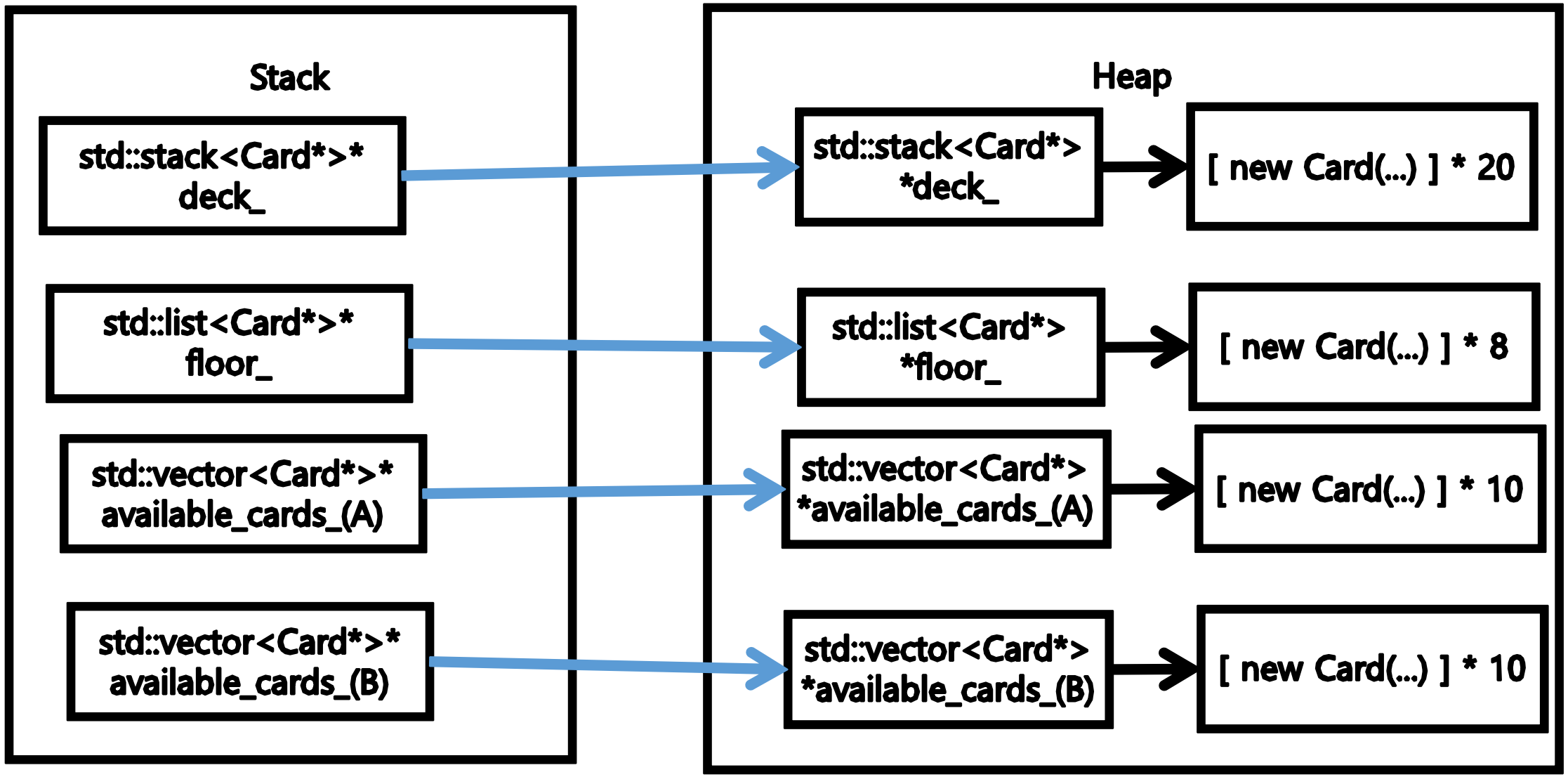
클래스 설계도(Judge)

Judge
bool is_game_over
int CalcScore(Player* a_player, Player* opponent_player) std::vector<Card*>* CardsForPlayer(std::stack<Card*>* a_deck) std::list<Card*>* CardsForBoardFloor(std::stack<Card*>* a_deck) std::stack<Card*>* InitDeck() | void ClearDeck(std::stack<Card*>* a_deck) std::vector<Card*>* MakeAllCards() | std::stack<Card*>* Suffle(std::vector<Card*>* all_cards) int CalcScoreForGwang(Player* a_player) | int CalcScoreForAnimal(Player* a_player, bool& is_mung) int CalcSocreForDan(Player* a_player) | int CalcScoreForP(Player* a_player) | bool IsGameOver(int a_score) std::stack<Card> InitGame() Getters & Setters

클래스 설계도(Board)

Board
std::stack<Card*>* dack_ std::list<Card*>* floor_
std::string ToStringForDack() std::string ToStringForFloor() Card* DrawCardFromDeck() bool StealPlfPossible(Player* opponent_player, std::list<Card*>* from_board_to_player) std::list<Card*>* CardsToPlayer(Card* card_from_player, Player* opponent_player) bool DackIsEmpty() std::vector<Card*>* MatchedCardWith(Card* card) Getters & Setters

클래스 설계도(카드 분배 이후)



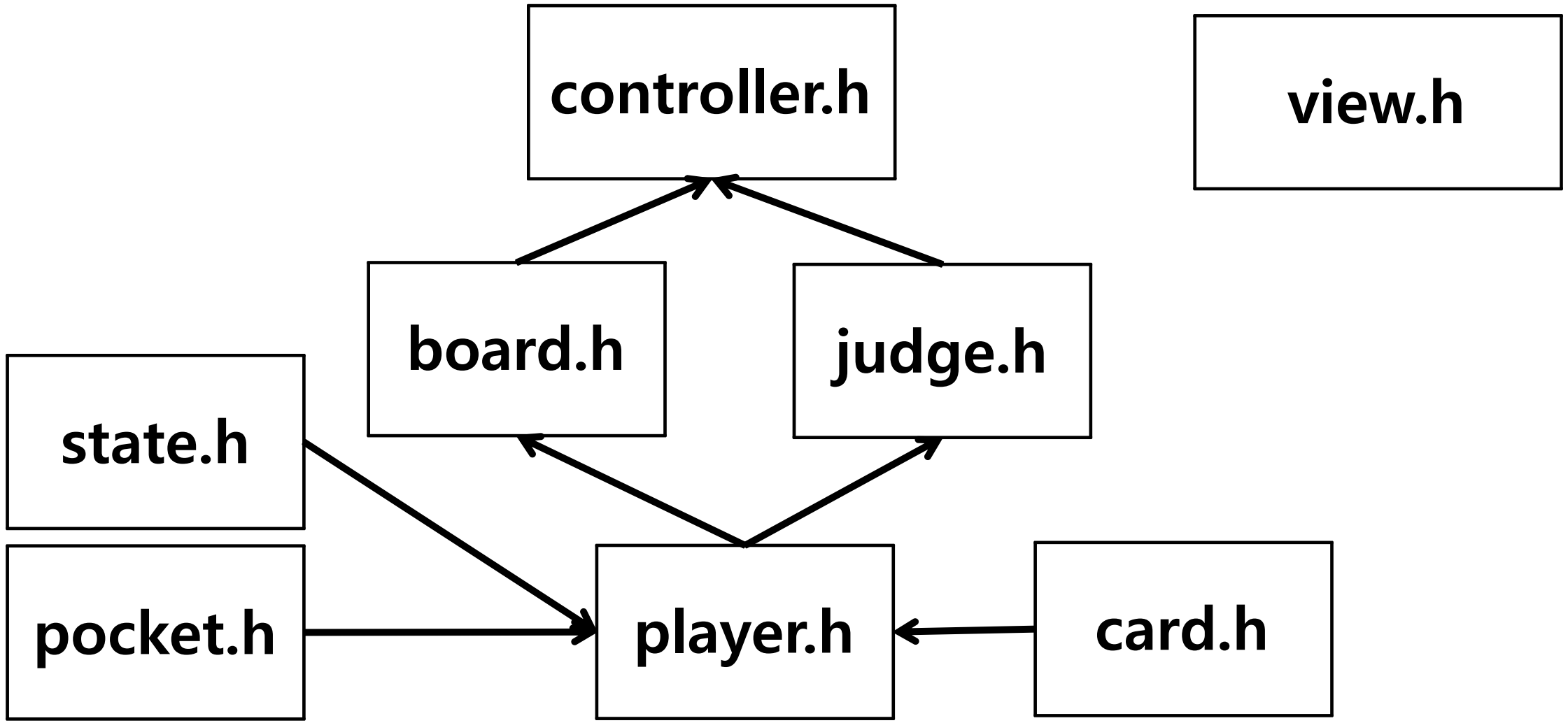
클래스 설계도(View)

View
-
View() [private constructor] static std::string InputString() | static char InputChar() | static int InputInt() static char InputUserAnswerAboutGameStart() | static int InputOrderFromPlayerAvailableCards(int bound) static void PrintStringWithLine(std::string a_string) | static void PrintStringWithoutLine(std::string a_string) static void PrintPlayerAvailableCards(std::string available_cards_info) static void PrintPlayerAllGettingCards(std::string all_getting_cards_info) static void PrintBoardDeck(std::string deck_info) static void PrintBoardFloor(std::string floor_info) static void PrintScore(int a_score)

클래스 설계도(Controller)

Controller
Player player_A_ Player player_B_ Board board_ Judge judge_
void PlayOf(Player* a_player, Player* opponent_player) void InitSetting() void PlayGame() void Run()

클래스 설계도(Dependency)



게임 진행 시나리오

- 게임 시작 시, 심판은 판을 초기화한다.
  (예시)
  >> 더미의 패들을 섞는다.
  >> 더미에서 플레이어에게 10장씩 패를 나누어 준다.
  >> 더미에서 8장의 패를 꺼내 판에 깐다.

게임 진행 시나리오

- 판에 깔린 패와 더미에 있는 패의 개수는 턴마다 출력
  (예시)
  ==========
  | Dummy (20개) |
  | 8(광) | 3(피) | 1(광) |
  ==========
- 플레이어가 낼 수 있는 패를 출력하고 낼 패 입력 받기
  (예시)
  플레이어 A가 낼 수 있는 패 : | 8(광)_0 | 1(피)_1 | 7(청단)_2 |
  플레이어 A가 낼 패 : 0 (→ “8(광)”을 낸다.)

게임 진행 시나리오

- 플레이어가 패를 내면 더미에서 패를 하나 깐다.
  (예시)
  >> 플레이어 A가 낸 패 : 1(피)
  >> 더미에서 나온 패 : 3(피)

- “플레이어가 낸 패의 숫자 == 더미에서 나온 패의 숫자”의 경우
  1. 해당 패가 판에 깔려 있는 경우, 싼다.
  2. 해당 패가 판에 깔려 있지 않은 경우, 쪽.

게임 진행 시나리오

- 플레이어가 점수를 내면 심판이 Go, Stop을 물어본다.
  (예시 1)
  !! 플레이어 A가 점수(5점)를 냈다. Go(1), Stop(0)을 고르세요.
  >> 1
  !! 플레이어 A가 Go를 하여 게임을 진행합니다.

  (예시 2)
  !! 플레이어 A가 점수(5점)를 냈다. Go(1), Stop(0)을 고르세요.
  >> 0
  !! 플레이어 A가 Stop을 하여 플레이어 A의 승리로 게임을 종료합니다. 플레이어 A는 5점을 획득합니다.

게임 진행 시나리오

- 판의 더미에 패가 없으면, Go Stop 여부에 관계 없이 게임 종료
  이 경우, 점수를 낸 사람이 없으면 무승부로 처리한다.
  (예시)
  >> 판의 더미에 패가 없어 게임을 종료합니다.
  >> 점수(7점)를 낸 플레이어 A의 승리입니다.

  (예시)
  >> 판의 더미에 패가 없어 게임을 종료합니다.
  >> 점수를 낸 플레이어가 없어 무승부입니다.