



algorithm



abstraction



computational thinking



pattern

decomposition

# 컴퓨팅사고와 SW코딩



coding

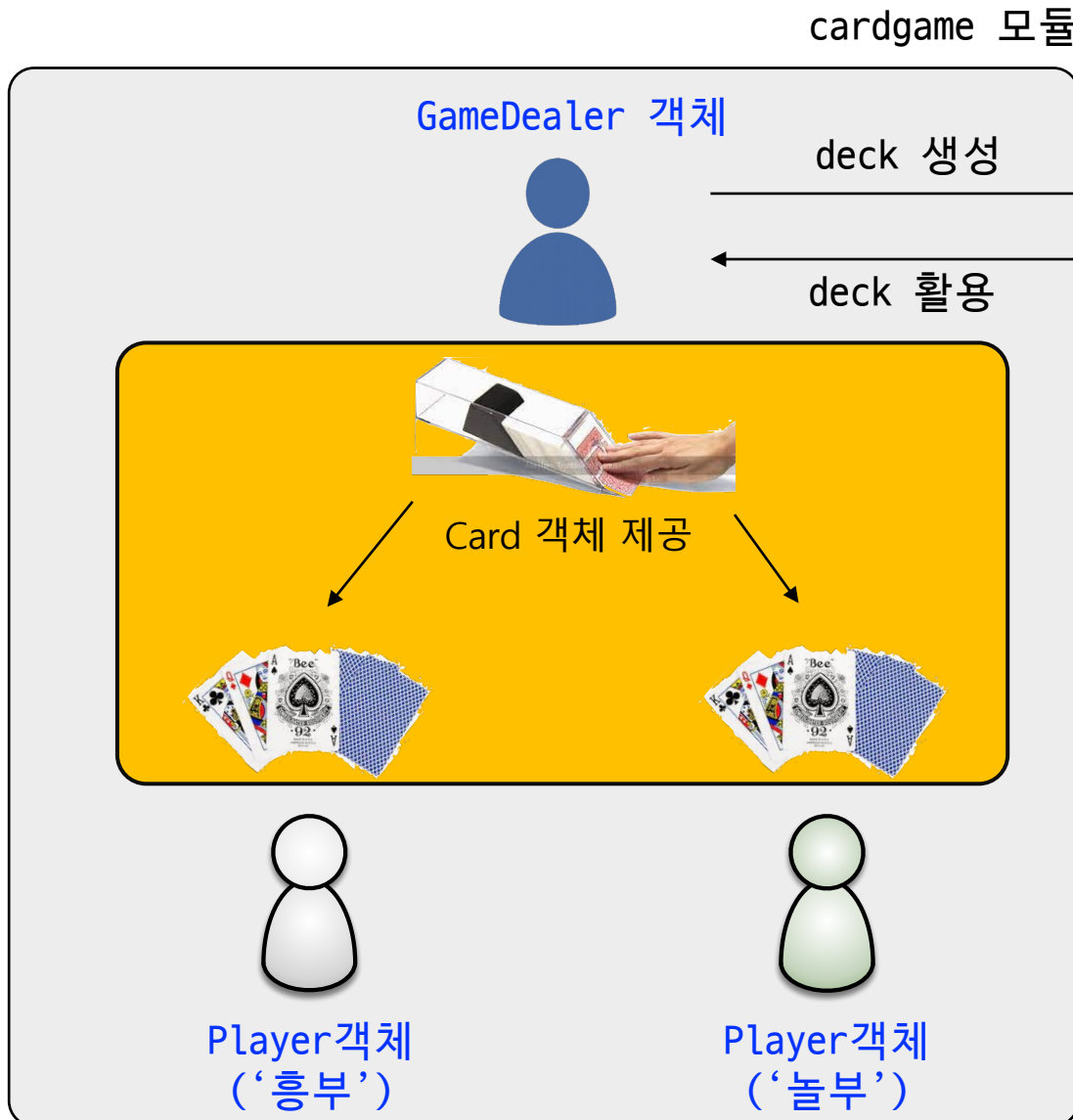


Problem  
analysis

## 과제 3: 카드 게임

# 카드 게임 구성

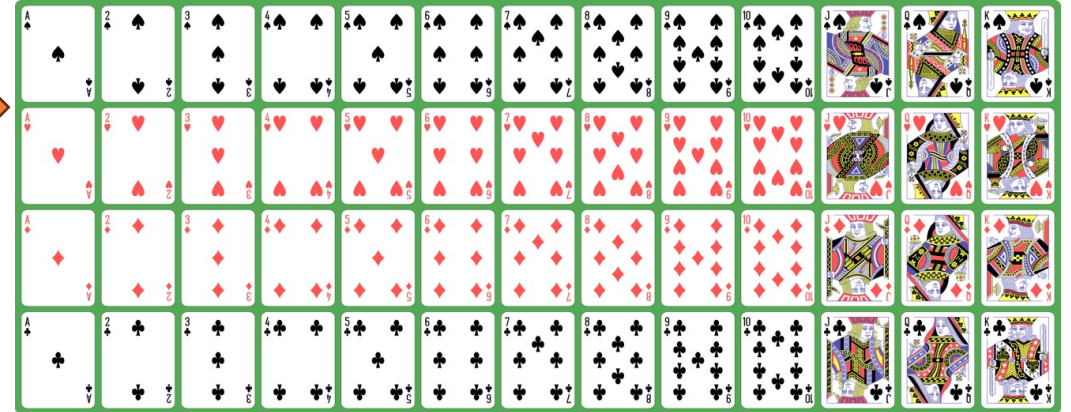
## 카드 게임 클래스 구성



Card 객체



1벌의 카드: `deck = list(Card객체)`

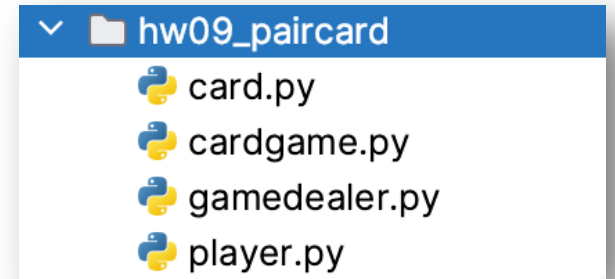


총 52장의 Card 객체 저장

# 카드 게임 클래스 설계

## ■ 구현 클래스: 각각 별도의 파일로 작성

- Card 클래스: [card.py](#)
  - 한 장의 카드를 나타내기 위한 클래스
  - suit와 number의 값을 가짐
- Player 클래스: [player.py](#)
  - 자신이 가지고 있는 카드 관리
    - 두 개의 리스트를 가짐 (holding\_card\_list, open\_card\_list)
  - 두 장의 동일한 카드를 제거하는 기능
    - 번호가 동일한 경우, holding\_card\_list에서 open\_card\_list로 이동: 테이블에 공개하는 기능
- GameDealer 클래스: [gamedealer.py](#)
  - 1벌의 카드(deck) 생성 기능: 리스트로 구현
  - 각 Player들에게 카드를 나누어 주는 기능
    - 자신이 가진 deck에서 제거하여 다른 선수들에게 제공
- 게임 동작 파일: [cardgame.py](#)
  - 각 클래스의 객체 생성 및 게임 진행



# Card 클래스

## ■ Card 클래스 구현

- 두 개의 변수(suit, number)
- `def __str__(self)` 기능 구현
  - 클래스를 문자열로 리턴

```
class Card:
    def __init__(self, card_suit, card_number):
        self.suit = card_suit
        self.number = card_number

if __name__ == '__main__':
    card = Card('♠', 10)
    print(card)
```

<\_\_main\_\_.Card object at 0x110499a30>



```
class Card:
    def __init__(self, card_suit, card_number):
        self.suit = card_suit
        self.number = card_number

    def __str__(self):
        '''
        객체를 문자열로 변환
        '''
        return f'({self.suit},{self.number:>2})'

if __name__ == '__main__':
    card = Card('♠', 10)
    print(card)
```

(♠,10)

# \_\_repr\_\_()과 \_\_str\_\_() 비교

## ■ 두 함수 모두 객체를 문자열로 변환

- \_\_repr\_\_()
  - 객체를 문자열로 표현하는 함수
  - 공식적인 방법(official)
  - 파이썬 인터프리터에서 확인
- \_\_str\_\_()
  - 객체를 문자열로 표현하는 함수
  - 비공식적 문자열 표현 (informal)
  - print(객체)에서 사용
- 클래스 내부에 \_\_str\_\_()이 없으면
  - \_\_repr\_\_()이 호출됨

## repr()과 str() 비교

```
class Card:
    def __init__(self, card_suit, card_number):
        self.suit = card_suit
        self.number = card_number
```

```
def __repr__(self):
    ...
    객체를 공식적인 문자열로 변환(인터프리터에서 객체 표현에 사용)
    ...
    return f'[{self.suit},{self.number:>2}]'
```

```
def __str__(self):
    ...
    객체를 문자열로 변환: print(객체)에 사용
    :return:
    ...
    return f'({self.suit},{self.number:>2})'
```

```
card = Card('♠', 10)
card # 파이썬 인터프리터에서 객체 출력: __repr__()
```

```
[♠,10]
```

```
print(card) # print(객체)는 __str__() 호출
```

```
(♠,10)
```

# GameDealer 클래스 설계 및 구현

## ■ GameDealer 클래스 기능

- GameDealer 객체는 card\_suits, card\_number를 이용하여 Card 객체 생성 및 리스트(deck)에 저장
  - card\_suits = ["♠", "♥", "♣", "♦"]
  - card\_numbers = ["A", "2", "3", "4", "5", "6", "7", "8", "9", "10", "J", "Q", "K"]

```
from card import Card
import random

class GameDealer:

    def __init__(self):
        self.deck = list()
        self.suit_number = 13

    def make_deck(self):
        card_suits = ["♠", "♥", "♣", "♦"]
        card_numbers = ["A", "2", "3", "4", "5", "6", "7", "8", "9", "10", "J", "Q", "K"]

        . . .
```

# GameDealer 클래스 설계 및 구현

## ■ GameDealer 클래스 구현

- deck의 내용을 출력하는 함수 구현
  - 반복문을 이용하여 deck에 있는 Card 객체를 하나씩 출력 (한 라인에 13장씩 출력)
    - `print(card)` 형태: Card 클래스의 `__str__()`호출
- 리스트 deck의 값을 랜덤하게 섞음
  - `random.shuffle(리스트)` 함수 호출
  - 랜덤하게 섞인 deck 내용 출력: 위의 deck 내용 출력함수 재사용

[GameDealer] 초기 카드 생성

-----  
[GameDealer] 딜러가 가진 카드 수: 52

(♠, A) (♠, 2) (♠, 3) (♠, 4) (♠, 5) (♠, 6) (♠, 7) (♠, 8) (♠, 9) (♠,10) (♠, J) (♠, Q) (♠, K)  
(♥, A) (♥, 2) (♥, 3) (♥, 4) (♥, 5) (♥, 6) (♥, 7) (♥, 8) (♥, 9) (♥,10) (♥, J) (♥, Q) (♥, K)  
(♣, A) (♣, 2) (♣, 3) (♣, 4) (♣, 5) (♣, 6) (♣, 7) (♣, 8) (♣, 9) (♣,10) (♣, J) (♣, Q) (♣, K)  
(♦, A) (♦, 2) (♦, 3) (♦, 4) (♦, 5) (♦, 6) (♦, 7) (♦, 8) (♦, 9) (♦,10) (♦, J) (♦, Q) (♦, K)

[GameDealer]카드 랜덤하게 섞기

-----  
[GameDealer] 딜러가 가진 카드 수: 52

(♠,10) (♣, 8) (♦, 8) (♠, 6) (♦, 3) (♦, 6) (♥, 4) (♥, 8) (♠, 4) (♠, 2) (♦, Q) (♦, A) (♥,10)  
(♠, J) (♥, A) (♥, J) (♣, Q) (♥, 5) (♣, 3) (♦,10) (♦, 5) (♥, 6) (♠, 5) (♦, J) (♥, 3) (♣, K)  
(♣,10) (♠, 8) (♥, Q) (♦, 2) (♠, A) (♣, 2) (♦, 7) (♠, 3) (♣, 6) (♣, J) (♥, K) (♠, K) (♦, 4)  
(♠, Q) (♣, 9) (♣, 4) (♥, 9) (♦, 9) (♦, K) (♥, 7) (♣, 5) (♠, A) (♣, 7) (♥, 2) (♠, 7) (♠, 9)

# Player 클래스

## ■ Player 클래스: `player.py`

- 자신이 가지고 있는 카드 관리
  - 두 개의 리스트를 가짐 (`holding_card_list`, `open_card_list`)
- 두 장의 동일한 카드를 제거하는 기능
  - 번호가 동일한 경우, `holding_card_list`에서 `open_card_list`로 이동: 테이블에 공개하는 기능
- 두 개의 리스트를 출력하는 기능

```
class Player:
    def __init__(self, name):
        self.name = name
        self.holding_card_list = list()
        self.open_card_list = list()

    def add_card_list(self, card_list):
        pass

    def display_two_card_lists(self):
        pass

    def check_one_pair_card(self):
        pass
```



# cardgame.py

## ■ 기능

- 전체 카드 게임을 조율하는 역할
- 각 클래스를 import 하고 각 클래스의 메소드를 호출해서 게임을 진행
- Player 객체 생성 및 동작
- GameDealer의 객체 생성 및 동작

```
from card import Card
from player import Player
from gamedealer import GameDealer

def play_game():
    # 두 명의 player 객체 생성
    player1 = Player("홍부")
    player2 = Player("놀부")

    dealer = GameDealer()
    . . .

if __name__ == '__main__':
    play_game()
```

# 카드 게임: 1단계

## ■ GameDealer 클래스 구현

- 각 Player에게 초기 10장씩 나누어 줌
  - deck[]에서 10장씩 각 Player에게 줌
  - deck[]에서는 총 20장의 카드가 삭제됨
- GameDealer가 가지고 있는 카드 목록(deck)을 화면에 출력
  - 32장 남음

## ■ Player 클래스 구현

- 자신이 가지고 있는 두 개의 리스트(holding\_card\_list, open\_card\_list)를 출력하는 함수
  - 리스트의 항목들을 하나씩 가져와서 print(card) 형태로 출력
- GameDealer가 전달해 준 10장의 카드를 자신의 holding\_card\_list에 저장
- Player는 자신이 가진 카드 목록을 화면에 출력

# 카드 게임: 1단계

```
=====
카드 나누어 주기: 10장
```

```
-----
[GameDealer] 딜러가 가진 카드 수: 32
```

```
(♠,10) (♣, 8) (♦, 8) (♠, 6) (♦, 3) (♦, 6) (♥, 4) (♥, 8) (♠, 4) (♠, 2) (♦, Q) (♦, A) (♥,10)
(♠, J) (♥, A) (♥, J) (♣, Q) (♥, 5) (♣, 3) (♦,10) (♦, 5) (♥, 6) (♠, 5) (♦, J) (♥, 3) (♣, K)
(♣,10) (♠, 8) (♥, Q) (♦, 2) (♠, A) (♣, 2)
```

```
=====
[홍부] Open card list: 0
```

```
[홍부] Holding card list: 10
```

```
(♠, 9) (♥, 2) (♣, A) (♥, 7) (♦, 9) (♣, 4) (♠, Q) (♠, K) (♣, J) (♠, 3)
```

```
=====
[놀부] Open card list: 0
```

```
[놀부] Holding card list: 10
```

```
(♠, 7) (♣, 7) (♣, 5) (♦, K) (♥, 9) (♣, 9) (♦, 4) (♥, K) (♣, 6) (♦, 7)
```

# 카드 게임: 2단계

## ■ Player 객체

- 번호가 같은 한 쌍의 카드 제거
  - 각 Player가 가지고 있는 카드 목록(holding\_card\_list)에서 번호가 같은 카드를 찾음
  - 찾은 카드는 holding\_card\_list에서 open\_card\_list로 이동시킴
  - 두 리스트의 현황을 화면에 출력

[2]단계: 다음 단계 진행을 위해 Enter 키를 누르세요!

=====

[홍부: 숫자가 같은 한쌍의 카드 검사]

=====

[홍부] Open card list: 2

(♠, 9) (♦, 9)

[홍부] Holding card list: 8

(♥, 2) (♣, A) (♥, 7) (♣, 4) (♠, Q) (♠, K) (♣, J) (♠, 3)

=====

[놀부: 숫자가 같은 한쌍의 카드 검사]

=====

[놀부] Open card list: 6

(♠, 7) (♣, 7) (♦, K) (♥, K) (♥, 9) (♣, 9)

[놀부] Holding card list: 4

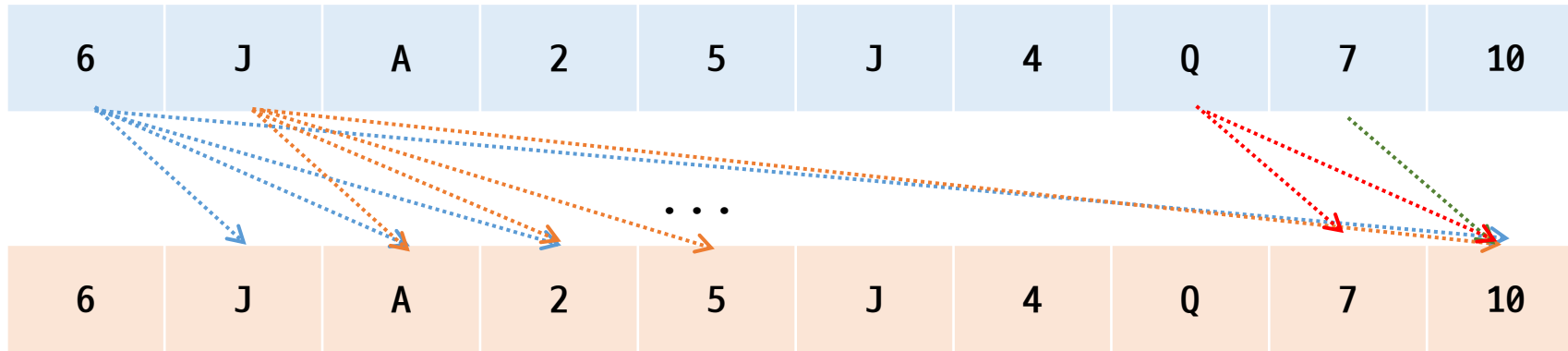
(♣, 5) (♦, 4) (♣, 6) (♦, 7)

# 두 장의 같은 번호 검사

## ■ 순차 탐색 방법

$[(\clubsuit, 6), (\heartsuit, J), (\spadesuit, A), (\clubsuit, 2), (\heartsuit, 5), (\clubsuit, J), (\diamondsuit, 4), (\heartsuit, Q), (\heartsuit, 7), (\heartsuit, 10)]$

$i=0$



$i=1$   $[(\clubsuit, 6), (\heartsuit, J), (\spadesuit, A), (\clubsuit, 2), (\heartsuit, 5), (\clubsuit, J), (\diamondsuit, 4), (\heartsuit, Q), (\heartsuit, 7), (\heartsuit, 10)]$

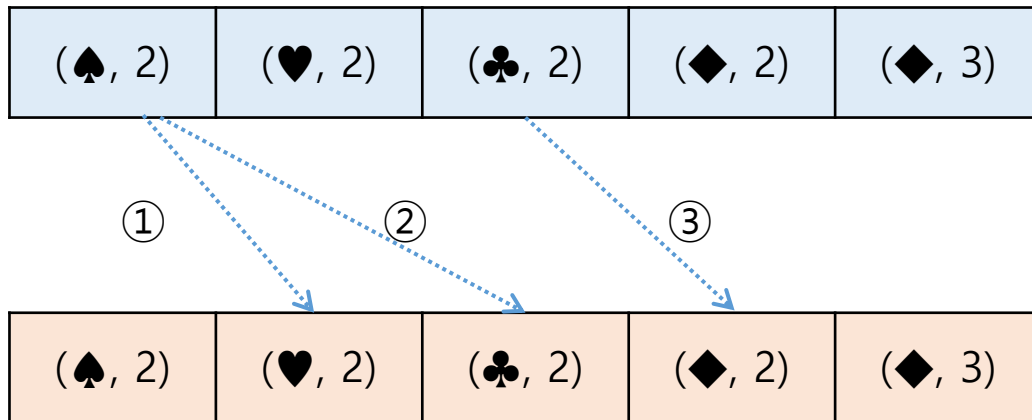
$j=i+1$   $[(\clubsuit, 6), (\heartsuit, J), (\spadesuit, A), (\clubsuit, 2), (\heartsuit, 5), (\clubsuit, J), (\diamondsuit, 4), (\heartsuit, Q), (\heartsuit, 7), (\heartsuit, 10)]$

# 순차 검색 과정

## ■ 검색 과정

- 중복된 파일이 2쌍이 존재하는 경우
  - self.holding\_card\_list = [(♠, 2), (♥, 2), (♣, 2), (♦, 2), (♦, 3)]

self.holding\_card\_list



① self.open\_card\_list = [(♠, 2) (♥, 2)] 에 추가

② self.open\_card\_list에 저장하기 전에  
리스트에 이미 (♠, 2)가 존재함  
→ self.open\_card\_list에 저장하지 않음

③ (♣, 2) (♦, 2)는 self.open\_card\_list에 모두  
존재하지 않음→ 둘 다 추가  
self.open\_card\_list = [(♠, 2), (♥, 2), (♣, 2), (♦, 2)]

# 카드 게임: 반복 수행

## ■ GameDealer 기능

- 각 Player에게 4장씩 카드를 나누어 줌

## ■ Player 기능

- GameDealer가 나누어 준 카드를 holding\_card\_list에 추가
- 5단계 기능을 반복: 번호가 같은 한 쌍의 카드 제거

## ■ 게임 종료 조건

- GameDealer가 가진 카드(deck)의 수가 0이거나
- 각 Player의 holding\_card\_list의 수가 0이면 게임 종료

## 3단계 실행 결과

[3]단계: 다음 단계 진행을 위해 Enter 키를 누르세요!

```
=====
카드 나누어 주기: 4장
-----
```

[GameDealer] 딜러가 가진 카드 수: 24

(♠,10) (♣, 8) (♦, 8) (♠, 6) (♦, 3) (♦, 6) (♥, 4) (♥, 8) (♠, 4) (♠, 2) (♦, Q) (♦, A) (♥,10)  
(♠, J) (♥, A) (♥, J) (♣, Q) (♥, 5) (♣, 3) (♦,10) (♦, 5) (♥, 6) (♠, 5) (♦, J)

```
=====
[홍부: 숫자가 같은 한쌍의 카드 검사]
=====
```

[홍부] Open card list: 6

(♠, 9) (♦, 9) (♥, 2) (♣, 2) (♠, K) (♣, K)

[홍부] Holding card list: 8

(♣, A) (♥, 7) (♣, 4) (♠, Q) (♣, J) (♠, 3) (♦, 2) (♠, 8)

```
=====
[놀부: 숫자가 같은 한쌍의 카드 검사]
=====
```

[놀부] Open card list: 6

(♠, 7) (♣, 7) (♦, K) (♥, K) (♥, 9) (♣, 9)

[놀부] Holding card list: 8

(♣, 5) (♦, 4) (♣, 6) (♦, 7) (♠, A) (♥, Q) (♣,10) (♥, 3)



## 4단계 실행 결과

[4]단계: 다음 단계 진행을 위해 Enter 키를 누르세요!

=====

카드 나누어 주기: 4장

-----

[GameDealer] 딜러가 가진 카드 수: 16

(♠,10) (♣, 8) (♦, 8) (♠, 6) (♦, 3) (♦, 6) (♥, 4) (♥, 8) (♠, 4) (♠, 2) (♦, Q) (♦, A) (♥,10)  
(♠, J) (♥, A) (♥, J)

=====

[흥부: 숫자가 같은 한쌍의 카드 검사]

=====

[흥부] Open card list: 8

(♠, 9) (♦, 9) (♥, 2) (♣, 2) (♠, K) (♣, K) (♣, J) (♦, J)

[흥부] Holding card list: 10

(♣, A) (♥, 7) (♣, 4) (♠, Q) (♠, 3) (♦, 2) (♠, 8) (♥, 6) (♦,10) (♥, 5)

=====

[놀부: 숫자가 같은 한쌍의 카드 검사]

=====

[놀부] Open card list: 12

(♠, 7) (♣, 7) (♦, K) (♥, K) (♥, 9) (♣, 9) (♣, 5) (♠, 5) (♥, Q) (♣, Q) (♥, 3) (♣, 3)

[놀부] Holding card list: 6

(♦, 4) (♣, 6) (♦, 7) (♠, A) (♣,10) (♦, 5)

## 5단계 실행 결과

[5]단계: 다음 단계 진행을 위해 Enter 키를 누르세요!

```
=====
카드 나누어 주기: 4장
-----
```

[GameDealer] 딜러가 가진 카드 수: 8

(♠,10) (♣, 8) (♦, 8) (♠, 6) (♦, 3) (♦, 6) (♥, 4) (♥, 8)

```
=====
[흥부: 숫자가 같은 한쌍의 카드 검사]
=====
```

[흥부] Open card list: 14

(♠, 9) (♦, 9) (♥, 2) (♣, 2) (♠, K) (♣, K) (♣, J) (♦, J) (♣, A) (♦, A) (♦, 2) (♠, 2) (♥, J) (♠, J)

[흥부] Holding card list: 8

(♥, 7) (♣, 4) (♠, Q) (♠, 3) (♠, 8) (♥, 6) (♦,10) (♥, 5)

```
=====
[놀부: 숫자가 같은 한쌍의 카드 검사]
=====
```

[놀부] Open card list: 18

(♠, 7) (♣, 7) (♦, K) (♥, K) (♥, 9) (♣, 9) (♣, 5) (♠, 5) (♥, Q) (♣, Q) (♥, 3) (♣, 3) (♦, 4) (♠, 4) (♠, A) (♥, A) (♣,10) (♥,10)

[놀부] Holding card list: 4

(♣, 6) (♦, 7) (♦, 5) (♦, Q)

## 6단계 실행 결과

[6]단계: 다음 단계 진행을 위해 Enter 키를 누르세요!

=====

카드 나누어 주기: 4장

-----

[GameDealer] 딜러가 가진 카드 수: 0

=====

[홍부: 숫자가 같은 한쌍의 카드 검사]

=====

[홍부] Open card list: 18

(♠, 9) (♦, 9) (♥, 2) (♣, 2) (♠, K) (♣, K) (♣, J) (♦, J) (♣, A) (♦, A) (♦, 2) (♠, 2) (♥, J) (♠, J) (♠, 8) (♥, 8) (♥, 6) (♦, 6)

[홍부] Holding card list: 8

(♥, 7) (♣, 4) (♠, Q) (♠, 3) (♦, 10) (♥, 5) (♠, 6) (♣, 8)

=====

[놀부: 숫자가 같은 한쌍의 카드 검사]

=====

[놀부] Open card list: 18

(♠, 7) (♣, 7) (♦, K) (♥, K) (♥, 9) (♣, 9) (♣, 5) (♠, 5) (♥, Q) (♣, Q) (♥, 3) (♣, 3) (♦, 4) (♠, 4) (♠, A) (♥, A) (♣, 10) (♥, 10)

[놀부] Holding card list: 8

(♣, 6) (♦, 7) (♦, 5) (♦, Q) (♥, 4) (♦, 3) (♦, 8) (♠, 10)

# Questions?