Card Game

ไพ่หนึ่งสำรับมีไพ่ 52 ใบ ประกอบด้วยไพ่ 4 ชุด โพดำ (Spade) โพแดง (Heart) ข้าวหลามตัด (Diamond) และดอกจิก (Club) แต่ละชุดมี 13 ใบประกอบด้วยตัวเลข 2 ถึง 10, Jack, Queen, King และ Ace

โปรแกรมของการบ้านนี้เป็นโปรแกรมเล่นเกมง่าย ๆ เกมหนึ่ง โปรแกรมเมอร์ที่เขียนงานนี้ตัดสินใจ แทนไพ่ 1 ใบด้วยสตริงที่ขึ้นต้นด้วย **|** ตามด้วย *ค่าไพ่* ตามด้วย*ชุดไพ่* แล้วปิดท้ายด้วย **|** โดยที่

- ค่าไพ่ เป็นตัวเลขหรือตัวอักษรหนึ่งตัว ได้แก่ A, 2, 3, 4, 5, 6, 7, 8, 9, T (แทน 10), J, Q และ K
- ชุดไพ่ เป็นตัวอักษรหนึ่งตัว ได้แก่ S, H, D และ C

สตริง **"|2H||4S||TD||AC|"** จึงแทนชุดไพ่ 4 ใบคือ 2 โพแดง (2H), 4 โพดำ (4S), 10 หลามตัด (TD) และ Ace ดอกจิก (AC)

การเล่นเกม

เกมเริ่มด้วยการตัดและกรีดไพ่สักพัก แล้วแจกไพ่ให้ผู้เล่น 2 คน คนละ 5 ใบ และวางไพ่หนึ่งใบไว้บนโต๊ะ เกมดำเนินไปโดยผู้เล่นทั้งสองสลับกันเลือกไพ่ในมือวางต่อท้ายไพ่บนโต๊ะ เรียงไปเรื่อย ๆ ดังนี้

- ไพ่ที่นำมาต่อได้ ต้องเป็นไพ่ที่มีค่าไพ่หรือชุดไพ่เหมือนกับไพ่ใบสุดท้ายบนโต๊ะ
- ถ้าไม่มีไพ่ใดในมือที่เลือกมาวางต่อท้ายไพ่บนโต๊ะได้ตามเงื่อนไขข้างต้น ผู้เล่นต้องจั่วไพ่ใบใหม่มาจากกองกลาง 1 ใบมาถือเพิ่ม
- ผู้เล่นใดที่ไพ่ในมือหมดก่อน เป็นผู้ชนะ
- ถ้าไพ่กองกลางหมด ผู้เล่นที่ถือไพ่เป็นจำนวนน้อยกว่าเป็นผู้ชนะ ถ้าถือจำนวนเท่ากัน เกมก็เสมอ

สิ่งที่ต้องทำ

ดูโปรแกรมในหน้าเกือบสุดท้าย แล้วไม่ต้องกังวล เพราะยังไม่ได้เรียนหลายคำสั่งของภาษาที่ใช้ในโปรแกรม สิ่งที่ต้องทำคือ เขียนรายละเอียดของฟังก์ชันที่เว้นว่างไว้ โดยใช้แค่คำสั่งพื้นฐานของสตริงกับลิสต์ก็พอ ดังนี้ (ชม**วิดีทัศน์ตัวอย่าง**การเล่นเกมของโปรแกรมเมื่อทำเสร็จสมบูรณ์)

```
def peek_kth_card(cards, k):
    # cards เป็นสตริงเก็บไพ่หลายใบ (อาจจะใบเดียวก็ได้) เช่น "|2H||4S||TD||AC|"
    # k     เป็นจำนวนเต็ม ระบุตำแหน่งไพ่ที่สนใจใน cards (ใบซ้ายสุดคือตำแหน่งที่ 1)

    # ต้องทำ: ตั้งค่าให้ตัวแปร the_kth_card เก็บสตริงที่แทนไพ่ใบที่ k ใน cards
    # ตัวอย่าง: cards เก็บ "|2H||4S||TD||AC|", k เก็บ 2
    #          ได้ the_kth_card เก็บ "|4S|"

    return the_kth_card
```

```
def remove_kth_card(cards, k):
    # cards เป็นสตริงเก็บไพ่หลายใบ (อาจจะใบเดียวก็ได้) เช่น "|2H||4S||TD||AC|"
    # k     เป็นจำนวนเต็ม ระบุตำแหน่งไพ่ที่สนใจใน cards (ใบซ้ายสุดคือตำแหน่งที่ 1)

    # ต้องทำ: ตั้งค่าให้ตัวแปร new_cards ให้เหมือน cards แต่ไพ่ใบที่ k เดิมของ cards ถูกลบทิ้ง
    # ตัวอย่าง: cards เก็บ "|2H||4S||TD||AC|", k เก็บ 2
    #          ได้ new_cards เก็บ "|2H||TD||AC|"

    return new_cards
```

```
def deal_n_cards(deck, n):
    # deck   เป็นสตริงเก็บไพ่หลายใบ (อาจจะใบเดียวก็ได้) เช่น "|2H||4S||TD||AC||AD||AS|"
    # n      เป็นจำนวนเต็ม ระบุจำนวนไพ่ที่ต้องการแจกออกจาก deck (n ใบซ้าย)
    # ต้องทำ: ตั้งค่าให้ตัวแปร cards เก็บสตริงที่แทนไพ่จำนวน n ใบทางซ้ายของ deck (ในลำดับเดิมที่อยู่ใน deck)
    #         ตั้งค่าให้ตัวแปร new_deck ให้เหมือน deck แต่ไพ่จำนวน n ใบทางซ้ายที่ k เดิมของ deck ถูกลบทิ้ง
    #         หมายเหตุ: ไม่ต้องสนใจกรณีที่ n มีค่ามากกว่าจำนวนไพ่ใน deck
    # ตัวอย่าง: deck เก็บ "|2H||4S||TD||AC||AD||AS|", n เก็บ 4
    #          ได้ cards เก็บ "|2H||4S||TD||AC|" และ new_deck เก็บ "|AD||AS|"

    return cards, new_deck
```

```
def cut(deck, m):
    # deck   เป็นสตริงเก็บไพ่หลายใบ เช่น "|2H||4S||TD||AC||3H||4H||5H||6H||7H||8H|"
    # m      เป็นจำนวนเต็ม ระบุจำนวนไพ่ที่สนใจทางซ้ายของ deck (m มีค่าได้ตั้งแต่ 0 ถึงจำนวนไพ่ใน deck)
    # ต้องทำ: ตั้งค่าให้ตัวแปร new_deck เก็บสตริงจากการย้ายไพ่ m ใบทางซ้ายของ deck มาต่อท้ายทางขวาของ deck
    # ตัวอย่าง: deck เก็บ "|2H||4S||TD||AC||3H||4H||5H||6H||7H||8H|", m เก็บ 4
    #          ได้ new_deck เก็บ "|3H||4H||5H||6H||7H||8H||2H||4S||TD||AC|"

    return new_deck
```

```
def shuffle(deck):
    # deck เป็นสตริงเก็บไพ่หลายใบ เช่น "|2H||3H||4H||5H||6H||7H||8H||9H||TH||JH||QH||KH||AH|"
    # ต้องทำ: ตั้งค่าให้ตัวแปร new_deck เก็บสตริงจากการนำไพ่ครึ่งซ้ายและครึ่งขวาของ deck มาวางสลับกันทีละใบ
    # ตัวอย่าง: deck เก็บ "|2H||3H||4H||5H||6H||7H||8H||9H||TH||JH||QH||KH||AH|"

    #          ได้ new_deck เก็บ "|2H||9H||3H||TH||4H||JH||5H||QH||6H||KH||7H||AH||8H|"
    # หมายเหตุ: ในกรณีที่จำนวนไพ่ใน deck เป็นจำนวนคี่ ให้ครึ่งซ้ายมีจำนวนมากกว่าครึ่งขวา (ดูตัวอย่าง)

    return new_deck
```

```
def show_table_cards(cards, m):
    # cards เป็นสตริงเก็บไพ่หลายใบ (อาจจะใบเดียวก็ได้) เช่น "|2H||4S||TD||AC|"
    # m      เป็นจำนวนเต็ม ระบุจำนวนไพ่ที่สนใจทางขวาของ cards ที่จะนำมาแสดง
    # ต้องทำ: นำไพ่ใน cards ทางขวา m ใบมาแสดง (ถ้ามีน้อยกว่าก็แสดงเท่าที่มี)
            ในกรณีที่มีมากกว่า m ใบ ต้องแสดง .... ทางซ้ายด้วย ดูรายละเอียดของรูปแบบการแสดงในตัวอย่าง
            ฟังก์ชันนี้ไม่มีการคืนค่าใด ๆ
```

ตัวอย่าง	ผลที่แสดง
cards เก็บ "|2H||3H||4H||5H|", m เก็บ 5	---------- **Table: |2H||3H||4H||5H|** ----------
cards เก็บ "|2H||3H||4H||5H|", m เก็บ 4	---------- **Table: |2H||3H||4H||5H|** ----------
cards เก็บ "|2H||3H||4H||5H|", m เก็บ 3	---------- **Table: ....|3H||4H||5H|** ----------
cards เก็บ "|2H||3H||4H||5H|", m เก็บ 2	---------- **Table: ....|4H||5H|** ----------

```
# Prog-03: Card Game
# # 6???????21 Name ?

import time
import random

def generate_deck(n_cards, n_shuffles):
  print('Shuffle', end='')
  deck = ''
  for suit in 'CDHS':
    for face in 'A23456789TJQK':
      deck += '|' + face + suit + '|'
  for i in range(n_shuffles):
    deck = cut(deck, random.randint(0,n_cards))
    deck = shuffle(deck)
    time.sleep(0.1)
    print('.', end='')
  print()
  return deck[:4*n_cards]

def play(n_cards):
  print('Start a card game.')
  deck = generate_deck(n_cards, 20)

  p1, deck = deal_n_cards(deck, 5)
  p2, deck = deal_n_cards(deck, 5)
  players = [p1, p2]

  table_cards, deck = deal_n_cards(deck, 1)
  fail = False
  turn = 0

  while True:
    show_table_cards(table_cards, 10)
    show_player_cards(players[turn], turn+1)
    k = select_card_number(players[turn])
    valid = (k != 0)
    if valid:
      cards = players[turn]
      card = peek_kth_card(cards, k)
      valid = eq_suit_or_value(card,
                        table_cards[-4:])
      if valid:
        table_cards += card
        players[turn] = remove_kth_card(cards,k)
        fail = False
    if not valid:
      print('  ** Invalid **')
      if len(deck) == 0:
        if fail: break
        fail = True
      if len(deck) > 0:
        print('  >> get a new card')
        card, deck = deal_n_cards(deck, 1)
        players[turn] = card + players[turn]

    show_player_cards(players[turn], turn+1)
    if len(players[turn]) == 0: break
    turn = (turn + 1) % len(players)

  if len(deck) == 0:
    print('\n** No more cards **')
  print('*****************')
  if len(deck) == 0 and \
     len(players[0]) == len(players[1]):
      print('Draw!!!')
  elif len(players[0]) < len(players[1]):
    print('Player # 1 win!!!')
  else:
    print('Player # 2 win!!!')
```

ใส่เลขประจำตัว ชื่อ นามสกุล

download code นี้ได้

```
def eq_suit_or_value(card1, card2):
  return card1[1] == card2[1] or \
         card1[2] == card2[2]

def show_player_cards(cards, k):
  print('  Player #', k, ':', cards)

def input_int(prompt):
  while True:
    try:
      return int(input(prompt))
    except:
      pass

def select_card_number(cards):
  n = len(cards)//4
  k = input_int('  Select card # (1-'+
           str(n)+') : ')
  if not(1 <= k <= n): k = 0
  return k

#---------------------------------------
def peek_kth_card(cards, k):
    
  return the_kth_card
#---------------------------------------
def remove_kth_card(cards, k):
    
  return new_cards
#---------------------------------------
def deal_n_cards(deck, n):
    
  return cards, new_deck
#---------------------------------------
def cut(deck, m):
    
  return new_deck
#---------------------------------------
def shuffle(deck):
    
  return new_deck
#---------------------------------------
def show_table_cards(cards, m):
    

#---------------------------------------
play(51)
```

โปรแกรมที่ส่ง ห้ามเปลี่ยนโค้ดส่วนที่เป็นสีแดงโดยเด็ดขาด เปลี่ยนได้เฉพาะส่วนที่มีพื้นหลังเป็นสีเขียวเท่านั้น

ข้อแนะนำ

คำสั่งหลัก ๆ ที่ใช้การทำงานในแต่ละฟังก์ชันคือ การใช้แนวคิดของ `index` และ `slice` ของสตริง บางฟังก์ชันอาจจะทำได้ง่ายขึ้น ถ้าใช้สตริง `split` ให้กลายเป็นลิสต์ แล้วก็ใช้ `index` และ `slice` ของลิสต์ช่วย

มีอยู่หนึ่งคำสั่งที่คู่กับ `split` ที่ยังไม่ได้นำเสนอในบทนี้ (อยู่ในบทลิสต์) คือ `join` มีวิธีใช้คือ

สตริง.join(ลิสต์ที่เก็บสตริง)

ผลที่ได้คือการนำสตริงแต่ละตัวในลิสต์ มาต่อ ๆ กัน โดยมี สตริง คั่นสตริงแต่ละตัวที่นำมาจากลิสต์ เช่น

```
s = 'aa,b,c,zzz'
x = s.split(',')  # ได้ x เก็บ ['aa', 'b', 'c', 'zzz']
t = '##'.join(x)  # ได้ t เก็บ 'aa##b##c##zzz'
```

วิธีทำการบ้านนี้

ดู[วิดีทัศน์]นี้

โบนัส

ถ้าได้เรียนคำสั่งในบทถัด ๆ ไป จะพบว่า คำสั่งที่ให้เขียนในแต่ละฟังก์ชันในการบ้านนี้ จะตรงไปตรงมา ไม่ซับซ้อน
ดังนั้น หากใครไม่ใช้คำสั่งจำพวก `if, for, while` ของบทถัด ๆ ไป จะให้คะแนนโบนัสอีก `20%` ของคะแนนที่ได้