

Project Proposal

CPE 261479

เรื่อง

Create Game Using Functional Programming

โดย

นายเจษฎา จินะกะ รหัส 620612144

เสนอ

อาจารย์ชินวัตร อิศราดิศัยกุล

Create Game Using Functional Programming

เป็น mini project ที่นำเสนอเกี่ยวกับการนำหลักความรู้ของวิชา 261497-2 Functional Programming (selected topics in computer software) การเขียนโปรแกรมเชิงฟังก์ชัน (หัวข้อเลือกสรร) นำความรู้มาใช้ประยุกต์ในการเขียน code ซึ่ง mini project นี้ ได้ทำเกี่ยวกับเกมไพ่ที่มีชื่อว่า Baccarat ซึ่งเป็นการจำลองสถานการณ์ระหว่าง 2 ฝ่าย
ในส่วนแรกของการสร้างเกมสิ่งที่ต้องใช้ก็คือ

1. Visual Studio Code

2. Python

3. Import Random

กติกาของเกม

1. แบ่งผู้เล่นเป็น 2 ฝ่าย คือ Player และ Banker

```
Please Choose your side (Player or Banker): Player
Player has cards:      3      A
Player has score of    4
Banker has cards:      3      J
Banker has score of    3
Player gets a third card: K
Banker gets a third card: 7
Player has final score of 4
Banker has final score of 0
Player wins
```

ซึ่งในการสุมการ์ดนั้นจะทำการสุม ทีละ 2 ใบ โดยมีกำหนดค่าของตัวเลขดังนี้

1. ไพ่ A จะมีค่าเท่ากับ 1
2. ไพ่ 2- 9 จะมีค่าเท่า 2-9 ตามลำดับ
3. ไพ่ J จะมีค่าเท่ากับ 0
4. ไพ่ Q จะมีค่าเท่ากับ 0
5. ไพ่ K จะมีค่าเท่ากับ 0

และจะมีกติกาในการจั่วไพ่ใบที่ 3 ดังนี้

แบ่งเป็น 2 ฝ่าย

1.ฝั่งPlayer

- ถ้าไพ่ 2 ใบแรกรวมกันเท่ากับ 1,2,3,4,5 และ 0 แต้ม ผู้เล่นจะต้องจั่วไพ่
- แต่ถ้ารวมกันแล้วเท่ากับ 6 หรือ 7 แต้ม ผู้เล่นจะอยู่ไม่มีการเรียกไพ่เพิ่ม รวมถึงการเปิดไพ่ 2 ใบแรก ที่ผู้เล่นได้ 8 และ 9 แต้ม ก็จะไม่มีการจั่วไพ่ เช่นเดียวกัน

2.ฝั่งของ Banker

- ถ้าไพ่ 2 ไพ่แรกฝั่งเจ้ามือรวมกันเท่ากับ 3 แล้วไพ่ใบที่ 3 ของฝั่งผู้เล่นมีแต้มเป็น 1,2,3,4,5,6,7,9 และ 0 ฝั่งเจ้ามือจะต้องทำการจั่วไพ่เพิ่ม ส่วนถ้าไพ่ใบที่ 3 ของผู้เล่นเท่ากับ 8 เจ้ามือไม่ต้องจั่วไพ่
- ถ้าไพ่ 2 ไพ่แรกฝั่งเจ้ามือรวมกันเท่ากับ 4 แล้วไพ่ใบที่ 3 ของฝั่งผู้เล่นมีแต้มเป็น 2,3,4,5,6, และ 7 ฝั่งเจ้ามือจะต้องทำการจั่วไพ่เพิ่ม ส่วนถ้าไพ่ใบที่ 3 ของผู้เล่นเท่ากับ 1,8,9 และ 0 เจ้ามือไม่ต้องจั่วไพ่
- ถ้าไพ่ 2 ไพ่แรกฝั่งเจ้ามือรวมกันเท่ากับ 5 แล้วไพ่ใบที่ 3 ของฝั่งผู้เล่นมีแต้มเป็น 4,5,6, และ 7 ฝั่งเจ้ามือจะต้องทำการจั่วไพ่เพิ่ม ส่วนถ้าไพ่ใบที่ 3 ของผู้เล่นเท่ากับ 1,2,3,8,9 และ 0 เจ้ามือไม่ต้องจั่วไพ่
- ถ้าไพ่ 2 ไพ่แรกฝั่งเจ้ามือรวมกันเท่ากับ 6 แล้วไพ่ใบที่ 3 ของฝั่งผู้เล่นมีแต้มเป็น 6, และ 7 ฝั่งเจ้ามือจะต้องทำการจั่วไพ่เพิ่ม ส่วนถ้าไพ่ใบที่ 3 ของผู้เล่นเท่ากับ 1,2,3,4,5,8,9 และ 0 เจ้ามือไม่ต้องจั่วไพ่
- ภาพตัวอย่างของเกม

```
Please Choose your side (Player or Banker):Player
Player has cards:      8      J
Player has score of    8
Banker has cards:      9      9
Banker has score of    8
Player has final score of 8
Banker has final score of 8
Tie
```

ตัวอย่างกรณีที่ เสมอกัน

```
Player has cards:      4      10
Player has score of    4
Banker has cards:      6      J
Banker has score of    6
Player gets a third card:      8
Player has final score of      2
Banker has final score of      6
Banker wins
```

ตัวอย่างกรณีที่ Banker Win จาก Player จั่วใบที่ 3

```
Player has cards:      4      2
Player has score of    6
Banker has cards:      A      3
Banker has score of    4
Player has final score of      6
Banker has final score of      4
Player wins
```

ตัวอย่างกรณีที่ Player win

```
Player has cards:      Q      4
Player has score of    4
Banker has cards:      9      3
Banker has score of    2
Player gets a third card: 10
Banker gets a third card: A
Player has final score of 4
Banker has final score of 3
Player wins
```

ตัวอย่างกรณีที่ Player wins จากการจั่วใบที่ 3

```
Player has cards:      A      A
Player has score of    2
Banker has cards:      5      5
Banker has score of    0
Player gets a third card: A
Banker gets a third card: 9
Player has final score of 3
Banker has final score of 9
Banker wins
```

ตัวอย่างกรณีที่ Banker wins จากการจั่วใบที่ 3

```

Player has cards:      2      8
Player has score of    0
Banker has cards:      6      K
Banker has score of    6
Player has final score of      0
Banker has final score of      6
Banker wins

```

ตัวอย่างกรณีที่ Banker wins รอบเดียว

การนำFunctional Programming มาใช้

-การใช้ Hight Order Function

นำLambda function เข้ามาช่วยในการกำหนด code ที่มีเงื่อนไข if หลายๆอันๆ

```

if (banker_score == 6 and player_third == 6 ) or \
    (banker_score == 6 and player_third == 7) or \
    (banker_score == 5 and player_third > 4) or \
    (banker_score == 5 and player_third > 5) or \
    (banker_score == 5 and player_third > 6) or \
    (banker_score == 5 and player_third == 7) or \
    (banker_score == 4 and player_third > 2 ) or \
    (banker_score == 4 and player_third > 3 ) or \
    (banker_score == 4 and player_third > 4 ) or \
    (banker_score == 4 and player_third > 5 ) or \
    (banker_score == 4 and player_third > 6 ) or \
    (banker_score == 4 and player_third > 7 ) or \
    (banker_score == 3 and player_third != 8) or \
    (banker_score == 0 ) or \
    (banker_score == 1 ) or \
    (banker_score == 2 ):
    banker_hand.append(random.choice(CARDS))
    print('Banker gets a third card:\t' + banker_hand[2])

```

จากFunction ดังกล่าวเป็นการเช็คกรณีได้ดังนี้

- 1.ถ้า Banker score เป็น 6 และ Player Score เป็น 6 หรือ 7 banker จะทำการจั่วไพ่ใบที่ 3
- 2.ถ้า Banker score เป็น 5 และ Player Score เป็น 4-7 banker จะทำการจั่วไพ่ใบที่ 3

3.ถ้า Banker score เป็น 4 และ Player Score เป็น 2-7 banker จะทำการจั่วไพ่ใบที่

4.หรือ ถ้า Banker score เป็น 0-2 banker จะทำการจั่วไพ่ใบที่ 3

จึงได้มีการใช้ lambda function มากำหนดขอบเขตเพื่อลดการใช้ if else

```
if ( banker_score == 6 and player_third in setrange(6,7)) or \
    (banker_score == 5 and player_third in setrange(4,7)) or \
    (banker_score == 4 and player_third in setrange(2,7)) or \
    (banker_score == 3 and player_third != 8) or \
    (banker_score in [0, 1, 2]):
    banker_hand.append(random.choice(CARDS))
    print('Banker gets a third card:\t' + banker_hand[2])
```

จากโค้ด ตอนแรกที่มีทั้งหมด 18 บรรทัดหากแต่เรานำfunction มาช่วยก็จะสามารถลดให้เหลือ 5บรรทัดได้

Code ก่อน:

```
if (banker_score == 6 and player_third == 6 ) or \
    (banker_score == 6 and player_third == 7) or \
    (banker_score == 5 and player_third > 4) or \
    (banker_score == 5 and player_third > 5) or \
    (banker_score == 5 and player_third > 6) or \
    (banker_score == 5 and player_third == 7) or \
    (banker_score == 4 and player_third > 2 ) or \
    (banker_score == 4 and player_third > 3 ) or \
    (banker_score == 4 and player_third > 4 ) or \
    (banker_score == 4 and player_third > 5 ) or \
    (banker_score == 4 and player_third > 6 ) or \
    (banker_score == 4 and player_third > 7 ) or \
    (banker_score == 3 and player_third != 8) or \
    (banker_score == 0 ) or \
    (banker_score == 1 ) or \
    (banker_score == 2 ):
    banker_hand.append(random.choice(CARDS))
    print('Banker gets a third card:\t' + banker_hand[2])
```

Code หลัง :

```
if ( banker_score == 6 and player_third in setrange(6,7)) or \
    (banker_score == 5 and player_third in setrange(4,7)) or \
    (banker_score == 4 and player_third in setrange(2,7)) or \
    (banker_score == 3 and player_third != 8) or \
    (banker_score in [0, 1, 2]):
    banker_hand.append(random.choice(CARDS))
    print('Banker gets a third card:\t' + banker_hand[2])
```

-และมีการนำ lambda เข้ามาใช้ในการรับค่าสองค่าแล้ว return กลับไปตามเงื่อนไขนั้นๆ

Code ก่อนใช้ lambda:

```
# Natural
if player_score > 8 and player_score < 9 or banker_score > 8 and banker_score < 9:
    if player_score != banker_score:
        if banker_score > player_score: #bankerWin
            return OUTCOME[1]
        if banker_score < player_score: #playerwin
            return OUTCOME[0]
    else:
        return OUTCOME[2]
```

Code หลังใช้ lambda:

```
# Natural
if player_score > 8 and player_score < 9 or banker_score > 8 and banker_score < 9:
    if player_score != banker_score:
        x = [lambda banker_score,player_score: OUTCOME[banker_score > player_score] if banker_score > player_score else OUTCOME[2]]
    return x
```

-การลดรูปของ if else ของ Code ในส่วนของการแสดงผลผู้ชนะ

โดยใช้ if else มาใช้กับ lambda แล้วนำไปเก็บในตัวแปร

Code ในส่วนหาก player ไม่เท่ากับ banker แล้วเช็คกรณี

```
if player_score != banker_score:
    if banker_score > player_score: #bankerWin
        return OUTCOME[1]
    if banker_score < player_score: #playerwin
        return OUTCOME[0]
else:
    return OUTCOME[2]
```

Code หลังจากใช้ lambda มาเก็บไว้ในค่า x แล้ว return x

```
# Natural
if player_score > 8 and player_score < 9 or banker_score > 8 and banker_score < 9:
    if player_score != banker_score:
        x = [lambda banker_score,player_score: OUTCOME[banker_score > player_score] if banker_score > player_score else OUTCOME[2]]
    return x
```


-การนำคำสั่งมาเก็บในfunction

เนื่องจาก code ที่ทำมาในส่วนแรกนั้นจะเป็น การเลือกฝั่งของผู้เล่นและจะมีการเริ่มเล่นในส่วน Code ของตัวเกมไม่ว่าจะเป็นการสุ่ม การคิดคะแนน ซึ่งในส่วนของcode ฝั่ง Player และ Banker นั้นมีความใกล้เคียงเหมือนกัน จึงทำการยุบทั้งหมดเป็น Function ที่ชื่อว่า play() แล้วค่อยเรียกมาใช้ในส่วนของ main()

Code ในส่วน player:

```
if(x == "Player"):
    player_hand = [
        random.choice(CARDS),
        random.choice(CARDS)
    ]
    banker_hand = [
        random.choice(CARDS),
        random.choice(CARDS)
    ]

    player_score = compute_score(player_hand)
    banker_score = compute_score(banker_hand)

    print('Player has cards:\t' + player_hand[0] + '\t' + player_hand[1])
    print('Player has score of\t' + str(player_score))
    print('Banker has cards:\t' + banker_hand[0] + '\t' + banker_hand[1])
    print('Banker has score of\t' + str(banker_score))

    # Natural
    if player_score > 8 and player_score < 9 or banker_score > 8 and banker_score < 9:
        if player_score != banker_score:
            if banker_score > player_score: #bankerwin
                return OUTCOME[1]
            if banker_score < player_score: #playerwin
                return OUTCOME[0]
        else:
            return OUTCOME[2]
    # Player has low score
    if player_score > 0 and player_score < 5:
        # Player get's a third card
        player_hand.append(random.choice(CARDS))
        player_third = compute_score([player_hand[2]])
        print('Player gets a third card:\t' + player_hand[2])

        if (banker_score == 6 and player_third == 6 ) or \
            (banker_score == 6 and player_third == 7) or \
            (banker_score == 5 and player_third > 4) or \
            (banker_score == 5 and player_third > 5) or \
            (banker_score == 5 and player_third > 6) or \
            (banker_score == 5 and player_third == 7) or \
            (banker_score == 4 and player_third > 2 ) or \
            (banker_score == 4 and player_third > 3 ) or \
            (banker_score == 4 and player_third > 4 ) or \
            (banker_score == 4 and player_third > 5 ) or \
            (banker_score == 4 and player_third > 6 ) or \
            (banker_score == 4 and player_third > 7 ) or \
            (banker_score == 3 and player_third != 8) or \
            (banker_score == 0 ) or \
```

Code ในส่วน Banker:

```
if(x == "Player"):
    player_hand = [
        random.choice(CARDS),
        random.choice(CARDS)
    ]
    banker_hand = [
        random.choice(CARDS),
        random.choice(CARDS)
    ]

    player_score = compute_score(player_hand)
    banker_score = compute_score(banker_hand)

    print('Player has cards:\t' + player_hand[0] + '\t' + player_hand[1])
    print('Player has score of\t' + str(player_score))
    print('Banker has cards:\t' + banker_hand[0] + '\t' + banker_hand[1])
    print('Banker has score of\t' + str(banker_score))

    # Natural
    if player_score > 8 and player_score < 9 or banker_score > 8 and banker_score < 9:
        if player_score != banker_score:
            if banker_score > player_score: #bankerWin
                return OUTCOME[1]
            if banker_score < player_score: #playerwin
                return OUTCOME[0]
        else:
            return OUTCOME[2]
    # Player has low score
    if player_score > 0 and player_score < 5:
        # Player get's a third card
        player_hand.append(random.choice(CARDS))
        player_third = compute_score([player_hand[2]])
        print('Player gets a third card:\t' + player_hand[2])

        if (banker_score == 6 and player_third == 6 ) or \
            (banker_score == 6 and player_third == 7) or \
            (banker_score == 5 and player_third > 4) or \
            (banker_score == 5 and player_third > 5) or \
            (banker_score == 5 and player_third > 6) or \
            (banker_score == 5 and player_third == 7) or \
            (banker_score == 4 and player_third > 2 ) or \
            (banker_score == 4 and player_third > 3 ) or \
            (banker_score == 4 and player_third > 4 ) or \
            (banker_score == 4 and player_third > 5 ) or \
            (banker_score == 4 and player_third > 6 ) or \
            (banker_score == 4 and player_third > 7 ) or \
            (banker_score == 3 and player_third != 8) or \
            (banker_score == 0 ) or \
```

Code หลังจากที่ยุบทั้ง๒ส่วน:

```
while(True):
    x = input("Please Choose your side (Player or Banker):")
    if x == "Player":
        print(play());
        y = input('Do you want to play again:')
        if y == 'y':
            continue
        elif y == 'n':
            break
    if x == "Banker":
        print(play())
        y = input('Do you want to play again:')
        if y == 'y':
            continue
        elif y == 'n':
            break
```

*Play() คือfunction ของ Banker และ Player ตัวอย่างข้างบนแต่มีการยุบให้เหลือเป็น function เดียวแล้วเรียก play() เอา

ภาคผนวก

File baccarat.py

```
# This is a python file to show how the game works
```

```
import random
```

```
CARDS = ['A', '2', '3', '4', '5', '6', '7', '8', '9', '10', 'J', 'Q', 'K']
```

```
VALUE = [1, 2, 3, 4, 5, 6, 7, 8, 9, 0, 0, 0, 0]
```

```
OUTCOME = ['Player wins', 'Banker wins', 'Tie']
```

```
def compute_score(hand):
```

```
    total_value = 0
```

```
    for card in hand:
```

```
        total_value += VALUE[CARDS.index(card)]
```

```
    return total_value % 10
```

```
def main():
```

```
    x = input("Please choose your side (Player | Banker ):")
```

```
    if(x == "Player"):
```

```
        player_hand = [
```

```
            random.choice(CARDS),
```

```
            random.choice(CARDS)
```

```
        ]
```

```
        banker_hand = [
```

```
            random.choice(CARDS),
```

```
            random.choice(CARDS)
```

]

```
player_score = compute_score(player_hand)
```

```
banker_score = compute_score(banker_hand)
```

```
print('Player has cards:\t' + player_hand[0] + '\t' +  
player_hand[1])
```

```
print('Player has score of\t' + str(player_score))
```

```
print('Banker has cards:\t' + banker_hand[0] + '\t' +  
banker_hand[1])
```

```
print('Banker has score of\t' + str(banker_score))
```

```
# Natural
```

```
if player_score > 8 and player_score < 9 or banker_score > 8 and  
banker_score < 9:
```

```
    if player_score != banker_score:
```

```
        if banker_score > player_score: #bankerWin
```

```
            return OUTCOME[1]
```

```
        if banker_score < player_score: #playerwin
```

```
            return OUTCOME[0]
```

```
    else:
```

```
        return OUTCOME[2]
```

```
# Player has low score
```

```
if player_score > 0 and player_score < 5:
```

```
    # Player get's a third card
```

```

player_hand.append(random.choice(CARDS))
player_third = compute_score([player_hand[2]])
print('Player gets a third card:\t' + player_hand[2])

if (banker_score == 6 and player_third == 6 ) or \
    (banker_score == 6 and player_third == 7) or \
    (banker_score == 5 and player_third > 4) or \
    (banker_score == 5 and player_third > 5) or \
    (banker_score == 5 and player_third > 6) or \
    (banker_score == 5 and player_third == 7) or \
    (banker_score == 4 and player_third > 2 ) or \
    (banker_score == 4 and player_third > 3 ) or \
    (banker_score == 4 and player_third > 4 ) or \
    (banker_score == 4 and player_third > 5 ) or \
    (banker_score == 4 and player_third > 6 ) or \
    (banker_score == 4 and player_third > 7 ) or \
    (banker_score == 3 and player_third != 8) or \
    (banker_score == 0 ) or \
    (banker_score == 1 ) or \
    (banker_score == 2 ):
    banker_hand.append(random.choice(CARDS))
    print('Banker gets a third card:\t' + banker_hand[2])

```

```

elif player_score > 6 and player_score < 7:

```

```
if banker_score >0 and banker_score <5:
    banker_hand.append(random.choice(CARDS))
    print('Banker gets a third card:\t' + banker_hand[2])
```

```
# Compute the scores again and return the outcome
player_score = compute_score(player_hand)
banker_score = compute_score(banker_hand)

print('Player has final score of\t' + str(player_score))
print('Banker has final score of\t' + str(banker_score))
```

```
if player_score != banker_score:
    # if banker_score > player_score: #bankerWin
    #     return OUTCOME[1]
    # if banker_score < player_score: #playerwin
    #     return OUTCOME[0]
    return OUTCOME[banker_score > player_score]
else:
    return OUTCOME[2]
```

```
elif(x == "Banker"):
    player_hand = [
        random.choice(CARDS),
        random.choice(CARDS)
    ]
```

```

banker_hand = [
    random.choice(CARDS),
    random.choice(CARDS)
]

player_score = compute_score(player_hand)
banker_score = compute_score(banker_hand)

print('Player has cards:\t' + player_hand[0] + '\t' +
player_hand[1])

print('Player has score of\t' + str(player_score))

print('Banker has cards:\t' + banker_hand[0] + '\t' +
banker_hand[1])

print('Banker has score of\t' + str(banker_score))

# Natural

if player_score > 8 and player_score < 9 or banker_score > 8 and
banker_score < 9:

    if player_score != banker_score:

        if banker_score > player_score: #bankerWin

            return OUTCOME[1]

        if banker_score < player_score: #playerwin

            return OUTCOME[0]

```



```
else:
```

```
    return OUTCOME[2]
```

```
# Player has low score
```

```
if player_score >0 and player_score < 5:
```

```
    # Player get's a third card
```

```
    player_hand.append(random.choice(CARDS))
```

```
    player_third = compute_score([player_hand[2]])
```

```
    print('Player gets a third card:\t' + player_hand[2])
```

```
#if banker needs a third card
```

```
if (banker_score == 6 and player_third == 6 ) or\
```

```
    (banker_score ==6 and player_third == 7) or\
```

```
    (banker_score == 5 and player_third > 4) or \
```

```
    (banker_score == 5 and player_third > 5) or \
```

```
    (banker_score == 5 and player_third > 6) or \
```

```
    (banker_score == 5 and player_third == 7) or \
```

```
    (banker_score == 4 and player_third > 2 ) or \
```

```
    (banker_score == 4 and player_third > 3 ) or \
```

```
    (banker_score == 4 and player_third > 4 ) or \
```

```
    (banker_score == 4 and player_third > 5 ) or \
```

```
    (banker_score == 4 and player_third > 6 ) or \
```

```
    (banker_score == 4 and player_third > 7 ) or \
```

```
    (banker_score == 3 and player_third != 8) or \
```

```
    (banker_score == 0 ) or\
```

```
(banker_score == 1 ) or\  
(banker_score == 2 ):  
    banker_hand.append(random.choice(CARDS))  
    print('Banker gets a third card:\t' + banker_hand[2])
```

```
elif player_score > 6 and player_score <7:  
    if banker_score >0 and banker_score < 5:  
        banker_hand.append(random.choice(CARDS))  
        print('Banker gets a third card:\t' + banker_hand[2])
```

```
# Compute the scores again and return the outcome  
player_score = compute_score(player_hand)  
banker_score = compute_score(banker_hand)
```

```
print('Player has final score of\t' + str(player_score))  
print('Banker has final score of\t' + str(banker_score))
```

```
if player_score != banker_score:  
    if banker_score > player_score: #bankerWin
```

```
        return OUTCOME[1]
```

```
    if banker_score < player_score: #playerwin
```

```
        return OUTCOME[0]
```

```
    else:
```

```
        return OUTCOME[2]
```

```
print(main());
```

File baccaratnew.py

```
import random
```

```
CARDS = ['A', '2', '3', '4', '5', '6', '7', '8', '9', '10', 'J', 'Q', 'K']
```

```
VALUE = [1, 2, 3, 4, 5, 6, 7, 8, 9, 0, 0, 0, 0]
```

```
OUTCOME = ['Player wins', 'Banker wins', 'Tie']
```

```
def compute_score(hand):
```

```
    total_value = 0
```

```
    for card in hand:
```

```
        total_value += VALUE[CARDS.index(card)]
```

```
    return total_value % 10
```

```
def play():
```

```
    player_hand = [
```

```
        random.choice(CARDS),
```

```
        random.choice(CARDS)]
```

```
    banker_hand = [
```

```
        random.choice(CARDS),
```

```

random.choice(CARDS)]

player_score = compute_score(player_hand)
banker_score = compute_score(banker_hand)

print('Player has cards:\t' + player_hand[0] + '\t' + player_hand[1])
print('Player has score of\t' + str(player_score))
print('Banker has cards:\t' + banker_hand[0] + '\t' + banker_hand[1])
print('Banker has score of\t' + str(banker_score))

# Natural

if player_score > 8 and player_score < 9 or banker_score > 8 and
banker_score < 9:
    if player_score != banker_score:
        x = [lambda banker_score,player_score: OUTCOME[banker_score >
player_score] if banker_score > player_score else OUTCOME[2]]
        return x

# Player has low score
if player_score > 0 and player_score < 5:
    # Player get's a third card
    player_hand.append(random.choice(CARDS))
    player_third = compute_score([player_hand[2]])
    print('Player gets a third card:\t' + player_hand[2])
    # lambda setrange

```

```

setrange = lambda start, end: range(start, end + 1)

if ( banker_score == 6 and player_third in setrange(6,7)) or \
    (banker_score == 5 and player_third in setrange(4,7)) or \
    (banker_score == 4 and player_third in setrange(2,7)) or \
    (banker_score == 3 and player_third != 8) or \
    (banker_score in setrange(0,2)):
    banker_hand.append(random.choice(CARDS))
    print('Banker gets a third card:\t' + banker_hand[2])

elif player_score > 6 and player_score <7:
    if banker_score >0 and banker_score <5:
        banker_hand.append(random.choice(CARDS))
        print('Banker gets a third card:\t' + banker_hand[2])

# Compute the scores again and return the outcome
player_score = compute_score(player_hand)
banker_score = compute_score(banker_hand)

print('Player has final score of\t' + str(player_score))
print('Banker has final score of\t' + str(banker_score))

if player_score != banker_score:
    return OUTCOME[banker_score > player_score]
else:
    return OUTCOME[2]

def main():

```

```
while(True):  
    x = input("Please Choose your side (Player or Banker):")  
    if x == "Player":  
        print(play());  
        y = input('Do you want to play again:')  
        if y == 'y':  
            continue  
        elif y == 'n':  
            break  
    if x == "Banker":  
        print(play())  
        y = input('Do you want to play again:')  
        if y == 'y':  
            continue  
        elif y == 'n':  
            break  
main()
```