# Pontoon (21 or bust!)

MAKING OUR DIGITAL FUTURE

## INTRODUCTION

Pontoon is a great card game. You can play it against the computer or your friends.

The rules are:

1) Each person has the option to take another card, or stay where they are
2) The person closest to scoring 21 wins
3) If you go over, you lose!

```
digilocaladmin@digilocal-HP-255-G7-Notebook-PC:~$
Your card is the King of Clubs
Computer draws the Jack of Clubs
You scored 10
Computer scored 10
Select (s)tick or (t)wist
>t
Your card is the 8 of Diamonds
Computer draws the 3 of Diamonds
You scored 18
Computer scored 13
Select (s)tick or (t)wist
>s
Stick.
Computer draws the 8 of Hearts
Sorry, you lose.
digilocaladmin@digilocal-HP-255-G7-Notebook-PC:~$
```

Let's get started!

## Step One: The core game

Open VS Code and start a new file.

We want to shuffle our deck of cards before playing, so we need to tell the computer how to pick random things.

```
1   import random
```

Next we need a deck of cards. We could just type them all in, but that's a bit boring. Let's get the computer to create a new deck for us.

# Pontoon (21 or bust!)

First we need a list of all the cards.

```
1   import random
2
3   cards = ['Ace', 2, 3, 4, 5, 6, 7, 8, 9, 'Jack', 'Queen', 'King']
```

This list actually contains two data types; strings, and integers. Many software languages won't let you do this, but Python will!

Next we need the four suits.

```
4   suits = ('Hearts', 'Clubs', 'Spades', 'Diamonds')
```

Notice these lists have ( ) brackets. That means they are tuples, they are immutable, can't be changed. Which is ok, because we don't need to change them.

Now we build our deck of cards. We do this by creating a new list with two entries for each card, the suit and the card. First we create an empty list, this time we use [ ] to tell python it's a list.

```
5   deck = []
```

To build our deck of cards, we need to go through each suit and add all the numbered cards. For this we need a nested loop like this:

```
7   for suit in suits:
8       for card in cards:
```

And for each card, we add a new `[card, suit]` list to our `deck[]`

```
9           deck.append([suit, card])
```

We can test this by adding a print command:

# Pontoon (21 or bust!)

```
1   import random
2
3   cards = ('Ace', 2, 3, 4, 5, 6, 7, 8, 9, 'Jack', 'Queen', 'King')
4   suits = ('Hearts', 'Clubs', 'Spades', 'Diamonds')
5   deck = []
6
7   for suit in suits:
8       for card in cards:
9           deck.append([suit, card])
10
11  print(deck)
```

Test your code!

```
digilocaladmin@digilocal-MP-255-G7-Notebook-PC:~$ /usr/bin/python3 /home/digilocaladmin/Barton-Hill/John/pontoon2.py
[['Hearts', 'Ace'], ['Hearts', 2], ['Hearts', 3], ['Hearts', 4], ['Hearts', 5], ['Hearts', 6], ['Hearts', 7], ['Hearts', 8], ['Hearts', 9], ['Hearts', 'Jack'], ['Hearts', 'Queen'], ['Hearts', 'King'], ['Clubs', 'Ace'], ['Clubs', 2], ['Clubs', 3], ['Clubs', 4], ['Clubs', 5], ['Clubs', 6], ['Clubs', 7], ['Clubs', 8], ['Clubs', 9], ['Clubs', 'Jack'], ['Clubs', 'Queen'], ['Clubs', 'King'], ['Spades', 'Ace'], ['Spades', 2], ['Spades', 3], ['Spades', 4], ['Spades', 5], ['Spades', 6], ['Spades', 7], ['Spades', 8], ['Spades', 9], ['Spades', 'Jack'], ['Spades', 'Queen'], ['Spades', 'King'], ['Diamonds', 'Ace'], ['Diamonds', 2], ['Diamonds', 3], ['Diamonds', 4], ['Diamonds', 5], ['Diamonds', 6], ['Diamonds', 7], ['Diamonds', 8], ['Diamonds', 9], ['Diamonds', 'Jack'], ['Diamonds', 'Queen'], ['Diamonds', 'King']]
digilocaladmin@digilocal-MP-255-G7-Notebook-PC:~$
```

## Step Two: Pick a card, any card!

We want to pick random cards from our deck. However, each time we run the code we'll get the same deck of cards, exactly. We need to shuffle them.

Fortunately, python has an easy shuffle command for this very purpose!

```
7   for suit in suits:
8       for card in cards:
9           deck.append([suit, card])
10
11  random.shuffle(deck)
12
13  print(deck)
```

Test your code!

# Pontoon (21 or bust!)

```
digilocaladmin@digilocal-HP-255-G7-Notebook-PC:~$
[['Hearts', 9], ['Clubs', 'Queen'], ['Diamonds', 9
iamonds', 'King'], ['Spades', 6], ['Diamonds', 8],
n'], ['Hearts', 'Jack'], ['Diamonds', 2], ['Diamon
, 2], ['Clubs', 9], ['Diamonds', 'Queen'], ['Clubs
'Spades', 8], ['Diamonds', 5], ['Spades', 7], ['He
digilocaladmin@digilocal-HP-255-G7-Notebook-PC:~$
```

- Delete or `#comment` out your print statement. We know the list is randomised, we don't want to give the game away!

- Now we need to pick a card. We could just read the first entry. However, we are going to want to pick several cards, and we don't want to pick the same card twice.

- We can use the command `.pop()` to select an item from a list, and then delete that item from the list.

```
13    my_card = deck.pop()
```

- Let's check it worked with a new print command. We're not going to print the whole deck, just the card we picked.

```
14    print('Your card is the', my_card[1], 'of', my_card[0])
```

- Remember, each element in our list `deck` has two data values, the card, and the suit. So `my_card[0]` is the first item in the data pair, the card value, and `my_card[1]` is the second item in the data pair, the suit. Your program should look like this (we've added some comments in green to explain each step).

# Pontoon (21 or bust!)

```
1   import random
2   # define our variables
3   cards = ('Ace', 2, 3, 4, 5, 6, 7, 8, 9, 'Jack', 'Queen', 'King')
4   suits = ('Hearts', 'Clubs', 'Spades', 'Diamonds')
5   deck = []
6   # build our deck of cards
7   for suit in suits:
8       for card in cards:
9           deck.append([suit, card])
10  # shuffle the deck
11  random.shuffle(deck)
12  # pick a card and remove it from deck
13  my_card = deck.pop()
14  print('Your card is the', my_card[1], 'of', my_card[0])
```


```
digilocaladmin@digilocal-HP-255-G7-Notebook-PC:~$
Your card is the 3 of Diamonds
digilocaladmin@digilocal-HP-255-G7-Notebook-PC:~$
Your card is the 9 of Diamonds
digilocaladmin@digilocal-HP-255-G7-Notebook-PC:~$
Your card is the Ace of Clubs
digilocaladmin@digilocal-HP-255-G7-Notebook-PC:~$
```

# Pontoon (21 or bust!)

## Step Three: Scoring!

Just after we create our deck, let's create a variable to hold our score.

```
6   my_score = 0
```

Before we can keep score, we have a problem with our list of cards. We can add the number cards easily (2, 3, 4, 5, etc). But we also have cards like 'Ace', 'Jack', 'Queen', and 'King'. We need to tell our programme how to handle these.

Aces can be high or low. To keep things simple, let's play Aces high!

```
16  if my_card[1] == 'Ace':
17      my_score += 11
```

Our other picture cards are all worth 10

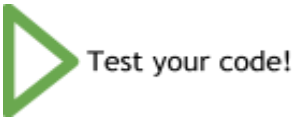
```
18  elif my_card[0] == 'Jack' or my_card[0] == 'Queen'or my_card[0] == 'King':
19      my_score += 10
```

Now we can add our score from the other cards.

```
20  else:
21      my_score += int(my_card[1])
```

Add a print statement to test your code.

```
22  print('You scored', my_score)
```




```
digilocaladmin@digilocal-HP-255-G7-Notebook-PC:~$
Your card is the Jack of Clubs
You scored 10
digilocaladmin@digilocal-HP-255-G7-Notebook-PC:~$
```

# Pontoon (21 or bust!)

## Step Four: Winning, or losing!

- We want to keep track of our score to see how close to 21 we can get!

- We'll create a new variable to control this. We'll set this to `True` or `False` and control our main repeat loop.

```
7   playing = True
```

- Now all our card selection & evaluation needs to go inside this new loop. Remember, lines 16 - 24 below you already have, don't type them out again!

```
15  while playing:
16      my_card = deck.pop()
17      print('Your card is the', my_card[1], 'of', my_card[0])
18      if my_card[1] == 'Ace':
19          my_score += 11
20      elif my_card[1] == 'Jack' or my_card[1] == 'Queen'or my_card[1] == 'King':
21          my_score += 10
22      else:
23          my_score += int(my_card[1])
24      print('You scored', my_score)
```

- If we've scored over 21 we lose

```
25      if my_score > 21:
26          print('Bust! You lose!')
27          playing = False
```

- Setting playing = False means the programme will exit the loop.

- If we scored 21, we win automatically. Rember, the \ before the ' in You're is so Python know's is a punctuation symbol, not the end of our print string.

```
28      elif my_score == 21:
29          print('Perfect Score. You\'re a winner!')
```

- If we aren't bust, do we want another card?

# Pontoon (21 or bust!)

- This is a little trickier as we're asking the user for an input. They might type (s) to stick with what they have, or (t) to twist and take another card, or they might try and type something else in to fool the programme.

- We can check for this with a local variable as below. We set our choice to 'z' as the default wrong answer. Then while we don't have a correct answer we can ask the user what their choice is.

```
30      else:
31          again = 'z'
32          while again == 'z':
```

- Here we use the .lower in case they type S, or T.

```
33              again = input('Select (s)tick or (t)wist\n>').lower()
```

- If they (s)tick with the score they have, we set playing = False and the game will exit our main loop.

```
34              if again == 's':
35                  print('Stick.')
36                  playing = False
```

- Else, if they (t) we pass - basically do nothing, they then pop another card..

```
37              elif again == 't':
38                  pass
```

- Or they typed some rubbish, in which case we go round again. We don't really care what they typed, so we don't need to check what it was.

```
39              else:
40                  again = 'z'
```

# Pontoon (21 or bust!)

## Step Five: Play against the computer!

Modify the line with your blank score variable to look like this .

```
6   my_score = computer_score = 0
```

We're now creating two variables and setting them both to `0`.

Now after you pick a card, we need to let the computer pick a card.

```
26      #computer's turn
27      computer_card = deck.pop()
28      print('Computer draws the', computer_card[1], 'of', computer_card[0])
29      #work out computer's score
30      if computer_card[1] == 'Ace':
31          computer_score += 11
32      elif computer_card[1] == 'Jack' or computer_card[1] == 'Queen' or computer_card[1] == 'King':
33          computer_score += 10
34      else:
35          computer_score += int(computer_card[1])
36      print('You scored', my_score)
37      print('Computer scored', computer_score)
```

We now work out who won! You could quit the game on the first go if you drew a higher card than the computer. So we need to let the computer keep playing once you quit to see if it can beat you.

Right at the end of our programme add this code for the computer to draw another card.

```
55                      again = 'z'
56  if computer_score < 21:
57      computer_card = deck.pop()
58      print('Computer draws the', computer_card[1], 'of', computer_card[0])
59      #work out computer's score
60      if computer_card[1] == 'Ace':
61          computer_score += 11
62      elif computer_card[1] == 'Jack' or computer_card[1] == 'Queen' or computer_card[1] == 'King':
63          computer_score += 10
64      else:
65          computer_score += int(computer_card[1])
```

# Pontoon (21 or bust!)

Next we need to check who the final winner is.

```
66  if my_score > computer_score and my_score <22:
67      print('You are a winner!')
68  elif my_score < 22 and computer_score > 21:
69      print('You are a winner')
70  else:
71      print('Sorry, you lose.')
```

Test your code!

```
digilocaladmin@digilocal-HP-255-G7-Notebook-PC:~$
Your card is the King of Clubs
Computer draws the Jack of Clubs
You scored 10
Computer scored 10
Select (s)tick or (t)wist
>t
Your card is the 8 of Diamonds
Computer draws the 3 of Diamonds
You scored 18
Computer scored 13
Select (s)tick or (t)wist
>s
Stick.
Computer draws the 8 of Hearts
Sorry, you lose.
digilocaladmin@digilocal-HP-255-G7-Notebook-PC:~$
```

## Challenge time!

**Can you add a two player option?**

**How about best of three**
**(you'll need to create a new deck for each game)?**