**Python 21 Card Trick Documentation**

**Introduction**

Having never used Python before being set a project to solve the card trick 21 felt quite a large step for me. However, I was excited to step up to the challenge. After reading through all the learning resources from lectures I realised I knew more than I originally thought.

We were asked to create a python command line application to recreate the card trick “21”. A game where the dealer would deal out 21 cards putting 7 cards in 3 columns side by side. The player would then select a card but not tell the dealer. Instead the player tells the dealer which column it is in. The dealer picks up the cards column by column sandwiching the column the player chose in between the other two columns. The dealer then re-deals the cards and asks the player where the card is now. After going through this process another two times the dealer picks out the 11th card which is, the players card.

When re-creating the game in python, I had to abide to the following rules:

* The player cannot tell the program what their card is as this would defeat the object of the trick.
* The program cannot use any external python libraries.
* All code must be referenced.

**Creating the game**

**Link to game:** https://gitlab.cs.cf.ac.uk/c1673107/21-Card-Trick-Python

**Core functionality**

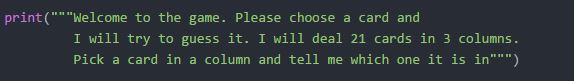
Firstly, the game prints out some welcome text informing the user on how to play and what is going to happen. There is then a 3 second break until the next few functions load. A function creates a deck of 52 random cards and a second function takes out 21 cards from this deck. A function is then run to present 3 columns of 7 cards to the user and presents an input field so they can type which column their card is in. A for loop then repeats this process and a final function takes the 11th cards and presents it to the player. Playing the game is quite simple, as the intro text explains. All the user needs to do is enter 1,2 or 3 as per the card they choose and which column it moves to. The card is automatically guessed at the end to save the user from having to ask for their card to be presented.

**Going into detail**

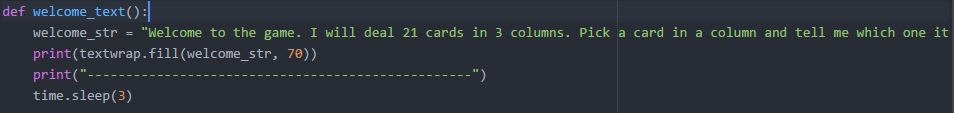
The game uses mainly a lot of the content I have learnt in the lectures. It also however incorporates further content from which I have found through multiple learning resources online; these sources are explicitly referenced in the comments of the code and at the end of this document.

**Printing the welcome text**

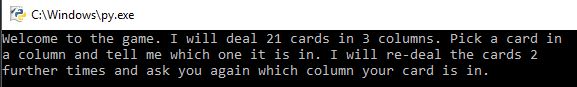
I thought it would be a good idea to inform the player on how to play and give an idea of how the game works. From what I had learnt in lectures I originally just printed using triple quotes as shown below:



However, what this did was when run in the command prompt, lines 2 and 3 were indented differently to the top line. I had to manually use different lines as to abide by the PEP guidelines of maximum 79 characters. After research, I found on stack overflow a way of using text wrapping. For example:



Here I do not have to put my string on different lines the import ‘textwrap’ from the standard library does it for me. Here I print out the string ‘welcome\_str’ in lines of maximum character length 70 as shown below:



**Creating a deck of 52 random cards**

Originally the first thing I did in the game was create a deck of 52 cards. I defined a function, ‘create\_shuffled\_deck’ which creates 2 lists, ‘cards\_value’ and’ cards\_suit’. The next step was to combine the 2 lists and add them to another list called ‘deck’. To do this I used my knowledge of for loops and lists from class notes so that for the entire list ‘cards\_value’, each string from ‘card\_suit’ would be concatenated to each string in ‘card\_value’. This is done by using the method ‘append’. I also used indexes to access each card suit. Again, using my class notes on imports, I used the standard library and the random import to then shuffle this deck of 52 cards.

