Intro to Programming for DA

Module Title:	Intro to Programming for DA.
Assignment Type:	Individual Practical Assignment.
Project Title:	Blackjack Card Game.
Project Date:	November 2020.
Assignment Compiler:	Amilcar Aponte.
Weighting:	40%
Due Date:	20th December @ 23:59. **Late submissions will be accepted until the 25th December @ 23:59. All late submissions are subject to a penalty of 10% of the mark awarded.**
Method of Submission:	Moodle Submission. **No email submission will be accepted.**

Module Learning Outcomes Assessed

- MLO 2. Design and implement algorithms and patterns for programming, particularly in the context of data analysis.
- MLO 3. Create, document and execute basic programs for data analysis.
- MOL 4. Demonstrate competence using commonplace tools/languages of data analytics

Assignment Introduction

You have been tasked to produce a command line simplified version of the Blackjack game.

System Requirements

- The user of your program will be the player and they will play against the dealer (the computer/your program).
  - You can assume that there is only one player against the dealer.
- The player is allowed to play as many times as they want.
- The player is dealt two cards. The dealer is also dealt two cards, one up (exposed) and one down (hidden).
- The values of the cards are as follows:
  - The value of cards two through ten is their pip value (2 through 10).
  - Face cards (Jack, Queen, and King) are all worth ten.
  - Aces are worth eleven.
- Players are allowed to draw additional cards to improve their hands (hit), but ultimately they decide when to hit or when to stay.
- A hand's value is the sum of the card values.
- Once the player has completed their hand, it is the dealer's turn.
- The dealer then reveals the hidden card and must hit until the cards total up to 17 points. At 17 points or higher the dealer must stay.

- The player is betting to have a better hand than the dealer. The better hand is the hand where the sum of the card values is closer to 21 without exceeding 21. The detailed outcome of the hand follows:
  - If the player is dealt an Ace and a ten-value card (called a "blackjack"), and the dealer does not, the player wins.
  - If the player exceeds a sum of 21 ("busts"), the player loses, even if the dealer also exceeds 21.
  - If the dealer exceeds 21 ("busts") and the player does not, the player wins.
  - If the player attains a final sum higher than the dealer and does not bust, the player wins.
  - If both dealer and player receive a blackjack or any other hands with the same sum called a "push", no one wins.
- You should implement object oriented good practices, so you've been provided two classes to model the Card and the Deck of card. These two classes must be used in your solution.

Notes

- You are allowed to use other tools apart from the ones seen in class, as long as it is your own code.
- Try to keep it simple. Planning your program before start coding is part of your assessment.
- Comment your code!!
- Plagiarism will not be tolerated. All work must be your own. If you used some snippet of code from an external source, make sure that you reference it correctly inside your code.
- In any situation, the lecturer is entitled to call you in for further explanation of your code.
- **Your code must run as no debugging will be done.**

Deliverables and Deadlines

You must submit through Moodle two separate files:

- Your python source code in format .py.
- One page PDF document that includes the reasoning behind your design choices.

Marking Scheme Summary

Description	**Weighting**
Program work correctly and the rules of the game are correctly followed.	
The winner is declared correctly.	15
Each game keeps going as long as the conditions are met and it stops after both player and dealer have stayed.	20
Hands are displayed correctly for the user to know what they have.	10
The counting of the hands is correct.	10
The value of the cards is correctly handled.	10
Correct data structures have been used and principles of object orientation have been followed. Correct classes, attributes and behaviours for the objects.	5
Validations have been implemented.	10
Program is correctly commented and well structured.	10
Documentation includes sensible reasoning for the design choices.	10
TOTAL	**100**