Big Data Programming 1: 2022 (30 Points)

Assignment 3: Python programming

Due date: 22.Sept.2020

How to submit:

The excersice must only be done in Python programming language using version 3.

Exercise 3 must be submitted by each individual in zip format with the name convention : matriculationNumber-fullName-Excercise-3.zip

The zip file must contain

1. the algorithm / pseudocode for exercise 1 in pdf format with the naming convention 'Excercise-3-algorithm.pdf'
2. the source code for Exercise 1 in python (.py) file format (if modules / packages are used then zip all python files)
3. the source code for Exercise 2 in python (.py) file format (if modules / packages are used then zip all python files)

Exercise 1: Card Game in Python(20 Points)

Game Mechanics: The game is a card game between two entities. The game can be played between two humans or a human and a machine. Choose a set of characters(fictional or real) and their characteristics. Characteristics are represented by a specific numeric value indicating the strength of the characteristic (strength may be indicated by negative or positive numeric value). Each card contains only one character and its characteristics.

Characteristics must be same across all characters but can vary in strength. No characteristic across characters can share the same value.

The two entities will be called Player 1 and Player 2 hence.

Distribute the cards equally among player 1 and player 2 face down such that the players cannot see the characters they have been given. Simulate a dice throw by both player 1 and player 2 where the highest number starts first

Constraints : Each player gets two special spells (God and Resurrect) . These spells are not associated to the characters but is assoicated to the player itself

Each round is played as below:

Player 1 : chooses the first card from the deck and can : play a charactertistic he wishes to challenge player 2 with . Player 2 chooses the first card in his deck and compares the charactertistic. The round is won by the Player whose characteristic weighs more and gets 1 point. The 2 cards are then kept faced down on a different deck called 'outdated'(the cards in the outdated deck are schuflled in a random order after each card is

placed). In round 2 , The player who won the previous round begins by choosing the next card in his deck and the round proceeds as above.

Spells:

In each round a player can decide to play the God spell or the Resurrect spell along with challenging with a characteristic.

God Spell: The player who starts the round must choose the next card and characteristic on his deck. After choosing he can play the God spell, forcing the opponenet player to play any card that he chooses. The game proceeds with the comparision of the the charactestic on the choosen card.

Resurrect Spell: Before choosing the next card and characteristic, the player who starts the round can choose Resurrect spell. Choosing Resurrect will create a random number and the card at the that random number position from the 'outdated' deck will be choosen. The card present at the random number at the 'outdated' deck will be added back to the top of the player deck. The player must then play this card.

When Player 1 plays the God spell , Player 2 cannot play the God spell as he is forced to choose the card, however player 2 can play the Resurrect spell and add a new card to the top of his deck. Player 1 can then force the resurrected card to be choosen or can go with his earlier choice.

God and Resurrect spell can only be played once by each Player.

The Game ends when all cards of one / both players have been played.

The game is won by the player with maximum points.

Please note : the game mechanics is not perfect and can imbalence the game , but for the purpose of this exercise that can be overlooked.

Exercise 2: Pattern Matching (10 Points)

Write a function that takes two string S1 and S2 at its input and returns a boolean denoting whether S2 matches / contains S1. (avoid using python regex or in build python regex matching features)

S1 is a sequence of: a-z - which stands for itself

S2 is a sequence of any number of the following:
a-z - which stands for itself
.(dot) - which matches 1 occurrence of any character
*(star)- which matches 0 or more occurrences of the previous single character

Examples:
S1 = "aba" , S2 = "*ab" => false
S1 = "aa" , S2 = "a*" => true
S1 = "ab" , S2 = ".*" => true
S1 = "ab", S2 = "." => false
S1 = "aab" , S2 = "c*a*b" => true S1 = "aaa" , S2 = "a*" => true