Dataset: [stories.zip](https://bits-pilani.instructure.com/courses/1482/files/275927?wrap=1)[Download stories.zip](https://bits-pilani.instructure.com/courses/1482/files/275927/download?download_frd=1)

The dataset has several small stories in text file. Consider each story as separate document.

**Term - Frequency Inverse Document Frequency**

1) Remove Stopwords (1 Mark)

2) Remove the punctuations. the special characters and convert the text to lower case.        (2 Mark)

3) create bigrams for the entire dataset and list down 10 most frequent bigrams. ( 2 Marks )

4) You have to implement TF-IDF the Algorithm from scratch.   ( 4 Mark )

5) Use the above-implemented algorithm and the values to calculate TF-IDF (using TF IDF formula) on the preprocessed data for unigrams and list down the top 10 words which have the highest TF-IDF Value. (2 Marks)

Refer to the following link to create POS labeled data: https://www.geeksforgeeks.org/part-speech-tagging-stop-words-using-nltk-python/

**Perform Part of Speech Tagging using the Viterbi Algorithm,**

6) Label the cleaned Tf-IDF dataset ( obtained after performing step 1 and step 2 )   ( 2 Mark )

7) Split the Train and the Test Dataset                      (1 Mark)

8) Implement the Viterbi Algorithm ( you can use Library) to get the Part of Speech Tagging.        ( 3 Marks)

9) Calculate the Accuracy and F1 score. ( Number of Predicted Correct Tag in the test set / Total number of Data points in the test set)   (2 Marks)

**Topic Modelling**

**Use cleaned Tf-IDF dataset ( obtained after performing step 1 and step 2 )**

10) Using the LDA algorithm create the Topics (10) for the Corpus             (3 Marks)

11) List down the 10 words in each of the Topics Extracted.           (2 Marks)

**Note: You may ignore if you find the words in the topic are not highly related.**