Question 2.2:

(i) How does your code perform when the input corpus is really large in size, say a billion sentences?

This is a naive algorithm, so a better algorithm is needed which scales better.

For that purpose we can use the N gram model. It predicts the next word/letter based on the last N words(using probability of previous words).

There are many other possible methods to accomplish this as well.

(ii) What are the parameters that determine how quickly you can predict the next character?

In my algorithm, the size of the text, If it is too big to be stored in the memory then the computation will be slower since I am using a list.

A lazy iterator can solve this problem.

Also a larger corpus will be slower but much more accurate.

(iii) What would you change in your code if you were asked to predict the next character in real-time as the user types his word?

Ill use a larger corpus and use the same idea of predicting the next word based on the probability of the occurrences of the words in the corpus.

Question 2.3:

Extend your code above to do whole word prediction.

For example, if the corpus is the same as above, when the user next types "is", then the engine should predict "lazy" and "long" as the options.

I can solve this problem very easily but unable to due to shortage of time. Will use the same idea with a larger corpus and use lazy iterators to save memory.