Part 1: I first approached this problem using the grammatical structure of language. I tried to come up with a way to identify the part of speech most likely associated with each word, and then "train" a probabilistic model on the ordering of different parts of speech. I was not able to figure out a way to map words to their likely part of speech without using additional data, so I decided to design a model that would focus on the orderings of words themselves, instead of the parts of speech associated with them. In my model, I consider words to be the nodes in a directed graph with weighted edges. If word **a** had an edge pointing to word **b** it represented that **b** follows directly after **a** in a sentence. The weight of the edge represented the number of times that word **b** followed word **a** in the training data. Note: in my implementation, I considered other non-word tokens, such as symbols to be nodes too, and I included the start of the sentence to be a node as well. I had two directed graphs: one for correct sentences, and another for incorrect sentences. In my code, I implemented these directed graphs with matrices.

The idea behind using a directed graph was to have a way of calculating the likelihood of getting from one word from another. I also hoped that by using a graph, my model could take into account the likelihood of certain paths in the graph, which would correspond to common sentence and clause structures. After the model was "trained," I calculated the probability that sentences from test.rand.txt were correct using a naïve Bayes classifier. I used Laplace smoothing to account for the fact that some words could appear in test.rand.txt that were not in the training data. Let S be the event that you get a sequence of word transitions $W_1, W_2, ...W_n$. Let C be the event that a sequence is a correct English sentence.

$$P(S|C) = \prod_{i=1}^{n} P(W_i|C)$$

$$P(S) = P(S|C) * P(C) + P(S|C^c) * P(C^c) \text{ by Law of Total Probability}$$

Note that $P(C^c) = P(C) = 1/2$, so those terms cancel when using Bayes theorem:

$$P(C|S) = \frac{P(S|C)}{P(S|C) + P(S|C^c)}$$

From my own testing, I found that my model was not very accurate. One possible problem with this model is that it focuses on the consecutive relationship between words, whereas in natural language, the context can sometimes be a better predictor of correctness. For example, if "tuxedo" was often followed by "is" and "tardigrade" was often preceded by "is" in the training data, my model would have marked the sentence "tuxedo is tardigrade" as most likely correct. If I were to do create this model again, I would experiment with methods to contextualize words. Another possible problem with this model could be an incorrect assumption of independence that I made to be able to use the naïve Bayes classifier.

Part 2:

My approach to ruin English was to choose a random word **z** in a correct sentence and alter **z** based on the probability of **z** following the preceding word **x**. I iterated through all possible outgoing edges of **x** $\{W_1, W_2...\}$, and found the difference between the probability that this transition would occur in a correct sentence and an incorrect sentence. Difference in probability = $P(W_i|C^c) - P(W_i|C)$. I replaced **z** with the word with the highest difference in probability. If this equation came up with the original word, I modified the spelling of the word.