Laboratory: Classic Ciphers

Classic Ciphers are encryption methods used to convey secret messages for civil and military purposes. They have a long history before the advent of computers and the creation of modern cryptography.

Classic ciphers usually work on preprocessed text without punctuation marks and white spaces. In what follows, we disregard the existence of digits and numbers. Furthermore, a common practice is to use a dictionary that translates important words into code words. Feel free to add more code words.

1. Write a function that takes a text file (specified by a file name) as input, changes all letters into capital letters and translates important words into code words. Use the text of the anglo-soviet agreement from 1920 as your test file. You do not need to worry about punctuation signs that might get lost.

Original	Code Word
Russia	Vermont
Russian	Vermontian
Soviet	Syrup
Government	Battleship
Great	Bloated
United	Greater
Kingdom	Hampshire
Britain	Illinois
British	Illinoisian
London	Milwaukee
Socialist	Maple
Republic	Flask
agents	salesmen

2. Write a function preprocess that takes two file names as input. The function reads the first file and writes only the letters in the alphabet in capitalized form into the second file. All non-letter symbols are to be lost with the exception of newlines.

A substitution code takes all letters (at this point, the file only consists of capital letters) and replaces all letters with another. In order for the process to be reversible, the mapping from one letter to the one substituting it needs to be a bijection. Another way of looking at this is to

create a permutation of the letters, where all the letters in the alphabet are arranged in a different order. Here is a way that we can create such a permutation. Take a long string such as: “IT RAINS A LOT IN MILWAUKEE IN THE WINTER, BUT SOMETIMES IT SNOWS.” This will serve as our key. We append to this string all the letters in the alphabet to obtain:

“ITRAINSALOTINMILWAUKEEINTHEWINTERBUTSOMETIMESITSNOWSABCDEFGHIJKLMNOPQRSTUVWXYZ”

We then run through the string from left to right and remove all letters that we have seen before. For our example, this gives

“ITRANSLOMWUKEHBCDFGJPQVXYZ”

We now create a dictionary, where the first letter in the alphabet has as value the first letter in the word, etc.

This gives `{'A': 'I', 'B': 'T', 'C': 'R', 'D': 'A', 'E': 'N', 'F': 'S', 'G': 'L', 'H': 'O', 'I': 'M', 'J': 'W', 'K': 'U', 'L': 'K', 'M': 'E', 'N': 'H', 'O': 'B', 'P': 'C', 'Q': 'D', 'R': 'F', 'S': 'G', 'T': 'J', 'U': 'P', 'V': 'Q', 'W': 'V', 'X': 'X', 'Y': 'Y', 'Z': 'Z'}`

3. Implement a function that takes a string as input, adds all at the letters in the alphabet to the string and then removes all duplicate letters from a string.

4. Implement a function that creates a dictionary for the substitution.

5. Implement a function that takes a dictionary and then creates the opposite dictionary, where the values are the keys and vice versa.

For example, the reverse of the dictionary is:
`{'I': 'A', 'T': 'B', 'R': 'C', 'A': 'D', 'N': 'E', 'S': 'F', 'L': 'G', 'O': 'H', 'M': 'I', 'W': 'J', 'U': 'K', 'K': 'L', 'E': 'M', 'H': 'N', 'B': 'O', 'C': 'P', 'D': 'Q', 'F': 'R', 'G': 'S', 'J': 'T', 'P': 'U', 'Q': 'V', 'V': 'W', 'X': 'X', 'Y': 'Y', 'Z': 'Z'}`

6. Implement a function that takes opens a text file, reads it line by line, removes all characters that are not in the alphabet, changes everything to capital letters, and then applies the dictionary in order to create the substitution code in a different file. The function should have the key and the two filenames as parameters.

7. Implement a function that takes the file with the cipher text and decodes it by applying the reverse dictionary on each of its letters.