Project 3 short report

20B11806
Saw Kay Khine Oo

Part 1: Encryption and Decryption

Since this is a program that can encrypt or decrypt depending on what the user wants, we have to write the encryption function and decryption function separately.

For the encryption function, we will convert the message user input into ascii code and put it in a list. Then, we use if to determine if the code is in range(97,123). Here, 97 is the code for small a and 122 is the code for small z. We need to be careful that range(97,123) accounts for 97 to 122. The key will be input by the user. Now, we only need to convert the whole list by looping and adding the key. There is one thing to notice. After adding the key, some values in the list might be converted into a number larger than 122 which is not a to z anymore. To avoid this, we use the following line.

```
a[i]= (a[i]-97+k)%26 +97
```

Lastly, we only need to decode a list of ascii codes and convert it into a string and the encrypted message is shown.

For the decryption, everything else is the same as encryption except that instead of adding the key, we need to reverse the process, so we subtract the key to decode the encrypted message into the original message. So, the decrypted message is shown.

In the main function, the user will be asked to choose between encryption and decryption. Depending on what the user chooses, the corresponding function will be called and executed.

Part 2: Deciphering a cipher text without knowing the key

While decrypting a coded message, the key is not always known. So, we need to decide how we will know the key in order to decipher a cipher text. In this program, character e will be considered as the most frequently used character in a long message. So, if we know the encrypted code for e, we will be able to find the key and decipher the message. As usual, first, we convert the message input by the user into ascii code.

Finding the encryption key

Next, we will need to define a dictionary named freq. In this dictionary, we will store the converted ascii code along with their frequency of usage in the message. For example, if a is used 10 times in the message and b is used 9 times in the message, the dictionary freq would be {97:10,98:9} with 10 and 9 being the values of codes or keys 97 and 98 respectively. To do this, we loop over a list of ascii codes of the message and use if to check if that particular code has already been stored or not. If the code is already in the freq dictionary, we will add 1 to the value of the code. If the code is not in the freq dictionary yet, we will set the value of the code to 1 first. After we have stored all the characters and their frequency respectively, we will take only the values of the freq dictionary and store it in a new list “values” to find out the maximum frequency.

We will set the element of the values list to a variable named maxfreq. Then, we will loop through the “values” list and if the next value is greater than the current maxfreq, we will replace the maxfreq value. In this way, we can find the maximum frequency.

Knowing the maximum frequency is important, but, knowing the code that the maximum frequency represents is more important. In order to figure this out, we will use the dictionary again. We will loop through every item in the dictionary and if the value is equal to the maximum frequency, we can call out the key which will be the code we want. Assuming this code is the character e as we discussed in the beginning, we can find the encryption key by simply subtracting the code for e.

<u>Main function</u>

The only difference between part 2 and part 1 decryption is that we don't know the encryption key in part 2. Now that we know the encryption key, everything else is the same.