Key in decimal = 91 57 53 119 74 76 52 80 105 69 41 55 117 94 80 45 81 40 37 94 45 95 50 53 52 100 104 49 70 64 64 110 110 69 49 50 56 101 82 111

The length of the key was found using the idea where we shifted the ciphertext and got matches. There were huge matches every 40th number which lead to finding the key length to be 40. Since ciphetext2 was not a plain text, I decrypted the first four characters of the ciphertext with all the available combination of possible printable ascii characters. By doing that, the possible file format was reduced to MPEG-1 Layer 3 file (key being 102, 62), GZIP (key being 100, 53), flash (key being 33, 115, 59) and ZIP (key being 91, 57, 53, 119). And then, the frequency analysis of each decrypted characters using the key characters corresponding encrypted characters was used to find the correct format. That part has been commented out in my program since it takes too long. Frequency analysis seemed like a good idea since all of the key character needed to have the similar structure of the frequency. The same logic was used to find the rest of the key characters. By knowing the most used decrypted characters from the first four key characters (20 or 22). I only had to look at either one or two possible choices of the key characters each. I first decrypted the file using all the key characters that corresponded to 20 being the most decrypted characters since three out of the first four key characters had 20 as most used decrypted characters. By doing that, I was able to create a zip file and was able to view the format of the file. I was able to tell that I was on the right track since the format of the file looked correct but couple words were spelled incorrectly. By counting which key characters were used, there were only couple keys that needed to be switched to the other key character.

This was all commented out to make the the code run faster if you want to test it out you can un-comment it out but as the moment we hard coded the key and key length in for efficiency and speed