MTP Assignment

OTP is used for perfect secrecy and here the onus is to choose the key of size equal to maximum of all the input plaintexts.
First step is to obtain several cipher texts which are encrypted using the same OTP. Next step is to XOR every pair of cipher text obtained in step 1 and check for spaces. The redundancy in form of ascii and the english language statistics helps us in identifying the key.
Now, we need to write the crack program to do cryptanalysis and prove that MTP is insecure since the common key for all ciphers can be retrieved back easily with some tricks.

The main trick is that-
When a space character is XORed with any alphabet character [‘a’-’z’ or ‘A’ to ‘Z’], then that particular character at that given position in the plaintext might be either a letter or a space.

If we XOR each one of the ciphertexts with one another, we end up getting XOR og plaintexts with the keystream [being same in MTP] getting cancelled out.

When XORing a space with any of the upper/lowercase letters, a number greater than or equal to 65 is obtained.

Please acknowledge that the key is being retrieved from the space attack..since some partial errors are there in the key, denoted by '*'.
In the crack code I am able to get the exact key [with a few obscure characters, because of the nondeterministic randomization]
So when we are going to use the retrieved key for cryptanalysis xor with star character is creating some unidentified random ascii #characters. So, there is partial recovery of the plaintexts.

Not only space attacks, but we can also use the frequency of alphabets in english language to make smart estimation of the ciphertexts.
Here, I propose the use of modified Transposition cipher to match the frequency of a particular encrypted character with that of the english letter alphabets. So, most frequently occurring characters like “e,t,a,o,i” will be mapped to a particular character in encrypted file; and using Reverse engineering or backtracking we can get entire plaintext with full accuracy.
Due to a shortage of time, it has not been implemented in the code.