Break the substitution cipher (A1, B1): [10 marks]

Each of you will be given a separate text file. The file will consist the ciphertext C (all capital letters, no space or punctuations marks), most frequent three characters, and some words that are present in the plain text P.

Your task is to decode the ciphertext i.e. find the plaintext P (contains only English **lowercase** alphabets) , find the key (Mapping between characters) and report the accuracy. In short, the output of the program will be:

1. The plaintext message in lower case letter (Even if you are not able to break the cipher completely, provide output as far you are able to decode, leave the remaining as they are present in the ciphertext C)

2. The mapping between P and C (for example: a = D, b = X, c = G,……)

3. Again encode the P to C and mention the accuracy. (like how many percent of characters are matched properly)

[bonus] If you can make your program dynamic, like for any given file it is able to complete the above-mentioned tasks.

Break the transposition cipher (A2, B2): [10 marks]

Each of you will be given a separate text file. The file will consist the ciphertext C (all capital letters, no space or punctuations marks and padded with XX… if necessary), and some words that are present in the plain text P.

Your task is to decode the ciphertext i.e. find the plaintext P (contains only English **lowercase** alphabets) , find the key (the order of columns) and report the accuracy. In short, the output of the program will be:

1. The plaintext message in lower case letter (Even if you are not able to break the cipher completely, provide output as far you are able to decode, leave the remaining as they are present in the ciphertext C)

2. The length and ordering of key (for example, if column size is 6, a sample ordering might be 5, 1, 2, 4, 3, 6.)

3. Again encode the P to C and mention the accuracy. (like how many percent of characters are matched properly)

[bonus] If you can make your program dynamic, like for any given file it is able to complete the above-mentioned tasks.

Implementing the encryption and decryption of DES(ALL) [15 Marks]

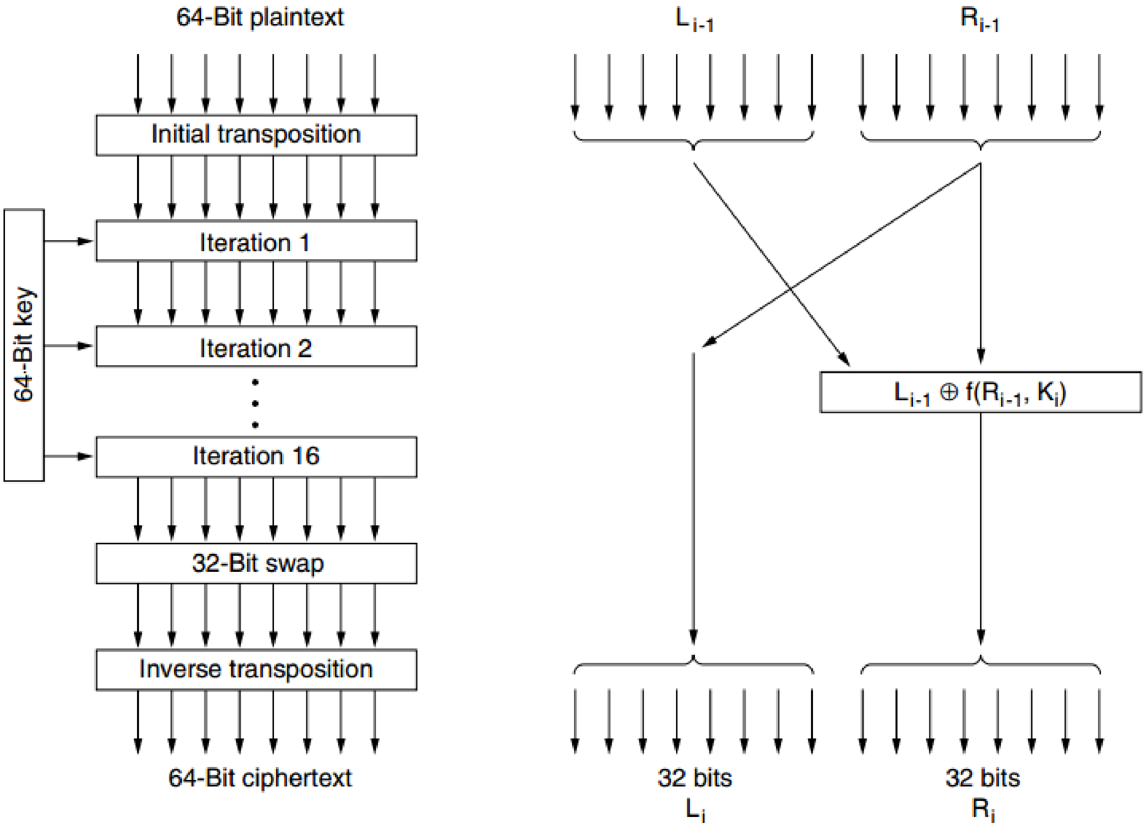
You need to implement the encryption and decryption part of a basic DES algorithm.

Input of the program: a sample plaintext string, a key (8 character long)

Output: Ciphered and deciphered string

Input	**Output**
Key: megabuck **Plaintext:** Hello world	**Ciphered:** ÿb\x00X®Ô\x7fCÅòJAsêöQ **Deciphered:** Hello world
Key: cse_buet **Plaintext:** Attack_at_dawn!	**Ciphered:** Î|'â£²èèÿ¼Ø8çþh\x97 **Deciphered:** Attack_at_dawn!

Encryption:



1. Group the plaintext by 64 bits (8 characters, consider space, punctuations marks, numbers also).

2. Run the encryption on this 64 bit chunk.

3. Use the PI matrix to transpose the 64 bit data

4. Run 16 iterations on these 64 bits (details later)

5. After iteration stops, swap the left-most 32 bits and rightmost 32 bits.

6. Use PI_1 to finally transpose these 64 bits before sending.

Iteration steps:

1. At iteration i, leftmost 32 bits L(i) = R(i-1) and rightmost 32 bits, R(i) = bitwise XOR of the left input and a function of the right input and the key for this stage, *Ki*

2. **Keys at each round:**

    a. In each of the 16 iterations, a different key is used.

    b. Before the algorithm starts, a 56-bit transposition is applied to the key according to CP_1 array.

    c. Just before each iteration, the key is partitioned into two 28-bit units, each of which is rotated left by a number of bits dependent on the iteration number according to SHIFT array

    d. *Ki* is derived from this rotated key by applying yet another 56-bit transposition to it according to CP_2 array. A different 48-bit subset of the 56 bits is extracted and permuted on each round

3. **Function at each iteration:**

    a. First, a 48-bit number, *e*, is constructed by expanding the 32-bit $R_i - 1$ according to E array

    b. *e* and *Ki* are XORed together

    c. Sample 32 bits from the result according to PI_2 arrays.

    d. Finally, these 32 bits are passed through a P-box. P box simulated as P array.

Decryption:

Reverse operation of encryption. **But use the keys and arrays in appropriate order.**

Remarks:

1. Try to do as far as possible. Marks will be given upon the incrementation of the algorithm (ex: how many steps have you implemented)

2. The ciphered text might not be always a valid character. Use appropriate print format to show it.

3. You cannot import and use any library functions for cryptographic use.