Encryption

A block cipher method called chaining can be used to make a much more secure ciphertext message. In this problem you will use the cipher block chaining method to encrypt a message. You can use any programming language that you like.

Here are the steps you should follow:

1. Convert the ascii key to a binary (10101010) representation.
2. Convert the ascii text to binary representation.
3. Break the binary text from number 2 up into some larger blocks (you will use 12 bits as your block size, but you theoretically could use anything). If the last block is less than 12, you should add alternating 1's and 0's to it until you have 12. Finally, if you added some padding, you should append a final block (of size 12) that contains the number of bits you added. I.e. If I added 4 bits of padding, I would append the block '000000000100' as the final block to be encoded.
4. Each of the above (12 bit) blocks will now be encrypted, following this process:
    - Take the first block to be encrypted, reverse it, Xor it with the first 12 bits of the key.(store as first block of encrypted output)
    - Take the next block and Xor it with the encrypted output of the first block. Reverse the new block, xor it with the first 12 bits of the key. (add to encrypted output stream)
    - repeat until all the blocks are encrypted.

Your code should take an input file with the ASCII key on the first line, the text to be encoded on the remaining lines, convert them to their binary representation, and use the cipher block chaining method as described above to 'encrypt' them. The resultant bitstream should then be output.

Example: ABC is my key ABCDE is the stuff I want to encrypt

This input:

ABC

ABCDE

Should produce this output:

011010010110111010100110000001100001100100101110001001011101

Hints:

```
    ABCDE in binary:

    0100000101000010010000110100010001000101

    After padding is added (to get last block up to 12 bits):

    0100000101000010010000110100010001000010110101010

    After the block is added that tells us how much padding was added (12-bit block):

    0100000101000010010000110100010001000101101010100000000001000

    Key in binary (first 12 bits): 010000010100
    First 12 bit block (reversed): 001010000010
    Xor                  : 011010010110 #first 12 bits of encrypted output

    2nd 12 bit block         : 001001000011
    Output of last Xor       : 011010010110
    Xor                  : 010011010101

    Key in binary (first 12 bits): 010000010100
    Xor'ed output (reversed)     : 101010110010
    Xor                  : 111010100110 #next 12 bits of encrypted output

    2nd 12 bit block         : 010001000100
    Output of last Xor       : 111010100110
    Xor                  : 101011100010

    And so on...
```

To pass off

Demonstrate that your code works. Maybe by showing that the given input produces the expected output. Upload your code as well.