Homework Number: 1
Name: Jose Luis Tejada
ECN Login: tejada
Due Date: Thursday 1/23/2020 at 4:29PM

Code:

```
def cryptBreak(ciphertextFile,key_bv):
    BLOCKSIZE = 16
    NUM_BYTES_BLOCK = BLOCKSIZE // 8

    #Read original file, Convert file to bitvector block by block
    fp = open(ciphertextFile)
    cipher_bv = BitVector(hexstring=fp.read())

    #Passphrase
    pass_phrase = "Hopes and dreams of a million years"
    pass_phrase_bv = BitVector(bitlist=[0]*BLOCKSIZE) #Generate Bitvector of size
blocksize
    for byte in range(0,len(pass_phrase) // NUM_BYTES_BLOCK):
        partial_string = pass_phrase[byte*NUM_BYTES_BLOCK:(byte+1)*NUM_BYTES_BLOCK]
        pass_phrase_bv ^= BitVector(textstring=partial_string)


    #For each block in bitvector perform incremental xoring
    plaintext_bv = BitVector(size=0) #Holds original message
    previous_cipher_block = pass_phrase_bv  #Previous bitblock is passphrase for 1st
iteration
    for i in range(0, (len(cipher_bv) // BLOCKSIZE)):
        current_cipher_block = cipher_bv[i*BLOCKSIZE:(i+1)*BLOCKSIZE] #Obtain one
block of ciphertext
        temp = current_cipher_block.deep_copy()
        current_cipher_block ^=  previous_cipher_block
        previous_cipher_block = temp
        current_cipher_block ^= key_bv
        plaintext_bv += current_cipher_block

    plaintext = plaintext_bv.get_text_from_bitvector()

    return plaintext
```

Plaintext Quote:
It is my belief that nearly any invented quotation, played with confidence, stands a good chance to deceive.

- Mark Twain

Encryption Key:
25202_{10} -> 0110001001110010_2

Explanation:

In order to recover the plaintext from the given encrypted text, there are three main steps that we must undertake. First, from utilizing the 'encrypt for fun' program as a reference, we must first obtain the initial bit vector that is used to perform incremental XOR-ing with the

encrypted string, that is the passphrase bit vector. We must do this incrementally, with each segment of the requisite block size being converted to a bit vector and appending each bit vector block to then use it as the previous bit vector for decryption. Once done this we then obtain a bit vector from the cipher text one block at a time. We make a copy of this bitvector, we perform XOR-ing with the previous bitvector block with this cipher block, and then set the previous cipher block to be the copied block. Finally, we XOR the result from the previous operation with the attempted encryption key and then append this final bitvector. Once this is done for all blocks, we can then convert this bitvector to text and determine whether it has the desired message.