Scientific Computing with MATLAB
Assignment 1

Paul Gribble

Psychology 9040A, Fall 2016

Due Date: Sunday October 2, 23:59:59 EST

Mr. Robot

You have received a transmission of encrypted text in the form of a list of integers [1]:

```
91,43,11,65,22,29,68,5,22,79,10,14,13,65,68,53,17,10,22,4,89,6,23,65,23,0,68,21,11,22,74,67
```

Your task is to report the decrypted message and the key used to encrypt it.

You have been given some information about how the message has been encoded.

- each character of the plaintext has been converted to an 8-bit ASCII byte (an 8-bit integer)
- then each byte has been XOR'd with a value (another 8-bit integer) taken from a secret key
- in MATLAB the `bitxor()` function performs this XOR operation
- the XOR function is symmetric: `bitxor(plaintextBytes,keyBytes)` gives you `ciphertextBytes`; `bitxor(ciphertextBytes,keyBytes)` returns the `plaintextBytes`
- the secret key is 4 bytes, and is repeated 8 times to match the length of the plaintext
- the secret key is made up of only lowercase characters chosen from 'a' through 'z'
- the first three bytes of the plaintext are: `34,68,111`

Here is an example showing how a message is encrypted using this method:

```
% encryption
%
plaintext = 'hello!';
key = 'me';
plaintext_bytes = int8(plaintext);
key_bytes = int8(key);
ciphertext = bitxor(plaintext_bytes,repmat(key_bytes,1,3));
disp(ciphertext)

    5    0    1    9    2   68
```

The plaintext and keys are each converted from a character string into an array of 8-bit integers using the `int8()` function. The ciphertext is then generated using the `bitxor()` function. Note that the second argument to `bitxor()` is the 8-bit integer representation of the key, repeated 3 times. This repetition is achieved using the `repmat()` function. The key is repeated 3 times because the plaintext is 6 bytes long and the key is only 2 bytes. Repeating the key 3 times means that the 6-byte plaintext is XOR'd with a 6-byte key (2-bytes repeated 3 times).

[1] http://www.gribblelab.org/scicomp/assignments/ciphertext.txt

Here is how the ciphertext can be decrypted:

```
% decryption
%
ciphertext = [5,0,1,9,2,68];
key = 'me';
key_bytes = int8(key);
plaintext_bytes_recovered = bitxor(int8(ciphertext),repmat(key_bytes,1,3));
plaintext = char(plaintext_bytes);
disp(plaintext)

hello!
```

The key line of code here is the one involving `bitxor()`, where we XOR the 8-bit integer representation of the ciphertext with the 8-bit integer representation of the key (repeated 3 times). We then use the `char()` function to convert the 8-bit integer representation of the plaintext (the ASCII codes) into actual human readable characters.

Hints

- you know the key is four characters long, and each character can be anything from ‘a’ to ‘z’
- this means there are $26^4 = 456,976$ possible keys
- computers are fast so you could try decrypting the ciphertext with each of them
- you know that the first three bytes of the plaintext are `34,68,111`
- remember: the `bitxor()` function is symmetric
- this should help you narrow your search

some code fragments that might be useful:

```
az = int8('a'):int8('z')
```

```
allkeys = combvec(az,az,az,az);
size(allkeys)
```

Sep 20, 2016, 3:23pm