Fall 2016 CIS 3362 Week One Assignment Solutions

For questions 1 - 3 decode each message. The techniques used to encrypt the messages are either similar to methods used in the day 1 activity or the affine cipher. In your write-up, explain the process you used to decrypt and include any code you might have used as an aid. Please do not use websites that automatically solve ciphers as most of your grade will be based on your description of the decryption process and original code you include in your write up.

1) nombizdsxqwocckqocscqyyngryvocywopex

Solution

A utility program was written to try each shift and the mirror cipher. This program is included at the end of this document. When a shift of +16 was applied to the ciphertext, the message:

```
DECRYPTING MESSAGES IS GOOD WHOLESOME FUN
```

is obtained. Note that this means that the encryption key was a shift of 10. We can verify this by noting that the first character with plaintext 'n' (value 13) when shifted back 10 characters yields 13 - 10 = 3, or character 'd'.

2) nriilinriililmgsvdzoodslrhgsvuzrivhglugsvnzoo

Solution

The use of the same utility program yields that this cipher was encrypted using the mirror system, so there was no key per se. The encryption function was f(x) = 25 - x, where x is the numeric value of the letter from 0 to 25, inclusive. Here is the plaintext:

```
MIRROR MIRROR ON THE WALL WHO IS THE FAIREST OF THEM ALL.
```

3) zinkpmttggimukhhzecykccqukfkaxgnhkfvkpixkpxefqgwzeazechz
kfkqftedkpixhmxdeduedhzkqcceudykdhepdihdiveufkqtcedakhzk
qcceudykdhecidtgwixhzidknkxakdhiphzkaimxckuxqfkvmhfihmxd
ciykhzeduedihzkxweckgimwidhvkkteuevtkpixpedqdaeqtqef

Solution

Neither the shift cipher or mirror system can be used to decrypt this, looking at the output of the utility program written for questions 1 and 2. It stands to reason, based on the directions given, that an affine cipher was used to encrypt this ciphertext.

If one feels very comfortable coding, he/she can try all 312 keys, but instead of printing all output to the screen, only print the output that corresponds to a valid word in the first k characters, for k ranging from 3 to 15. (This way, there's less to visually sift through.)

If one doesn't feel like writing much code, he/she can just print out all 312 possible messages and sift through all the output visually. In the included solution this is what was done. The text of the program is included at the end of the solutions.

After two minutes of reading through the file storing the output, I was able to find the following keys for decryption a = 21, b = 2, yielding the following plaintext:

```
Hopefully you get this message decrypted before Friday which is
the deadline for turning in the assignment. If not, no big deal
since the assignment is only worth one percent of the course grade.
But do turn something in otherwise you won't be eligible for
financial aid.
```

Another possible technique would have been to look for common letter frequencies and make guesses for 'e' and 't', setting up equations to solve for the a and b for decryption which would correspond to those guesses.

Included at the end of the homework is a short program that calculates letter frequencies. We see that in this ciphertext, k appears 29 times and e appears 21 times. It might stand to reason that k maps to the plaintext e and e maps to the plaintext t. Setting up these equations we get:

$f(10) = 10a + b \bmod 26 = 4$ (our guess is k → e)
$f(4) = 4a + b \bmod 26 = 19$ (our guess is e → t)

Subtracting, we get $6a \equiv -15 \pmod{26}$, or $6a \equiv 11 \pmod{26}$. This equation has no solutions because 2 divides evenly into 6 and 26, but not 11.

So, at least one of those guesses is wrong. Let's try mapping e to s instead of t for the second equation. 's' is also a common letter in English and the frequency of other letters in the cipher text was very close to that of 'e'. Now we get:

$f(10) = 10a + b \bmod 26 = 4$ (our guess is k → e)
$f(4) = 4a + b \bmod 26 = 18$ (our guess is e → s)

This time, we find that $6a \equiv -14 \pmod{26}$, so $6a \equiv 12 \pmod{26}$. It follows that a = 2 or 15. This leads to a dead end.

Next we see that there are 18 h's. Maybe h maps to a plaintext of t. This gives us:

$f(10) = 10a + b \bmod 26 = 4$ (our guess is k → e)
$f(7) = 7a + b \bmod 26 = 19$ (our guess is h → t)

Subtracting, we get $3a \equiv -15 \bmod 26$, so $a \equiv -5 \bmod 25$, so $a \equiv 21 \bmod 26$. Plugging in a = 21 to the second equation, we get $7(21) + b = 19 \bmod 26$. Solving for b we get $b \equiv (19 - 147) \bmod 26$, so $b \equiv -128 \bmod 26$, $b \equiv 2 \bmod 26$.

So, in avoiding sifting through all of the data, the cryptanalysis route may take some trial and error.

4) Using the affine cipher with the encryption keys a = 7 and b = 12, encrypt the following plaintext:

```
thequickbrownfoxjumpedoverthelazydog
```

Note: If you write a program to perform the encryption, please include the text of that program in your write-up.

Solution

A simple edit can be made to any utility program for #3 to just print out the given affine operation only for a = 7 and b = 12. Here is the corresponding ciphertext:

```
Pjouwqaetbgkzvgrxwsnohgdobpjolmfyhgc
```

Program for Questions 1, 2

```
#include <stdio.h>
#include <string.h>

int main() {

    char cipher[1000];
    scanf("%s", cipher);

    int i, j;

    // Try each shift.
    for (i=0; i<26; i++) {
        printf("%d\t", i);
        for (j=0; j<strlen(cipher); j++)
            printf("%c", (cipher[j]-'a'+i)%26 + 'a');
        printf("\n");
    }
    printf("\n");

    // Try mirror.
    for (i=0; i<strlen(cipher); i++)
        printf("%c", ('z'-cipher[i]) + 'a');
    printf("\n");
    return 0;
}
```

Program for Question 3

```
#include <stdio.h>
#include <string.h>

const int ALIST[] = {1,3,5,7,9,11,15,17,19,21,23,25};

int main() {

    char cipher[1000];
    scanf("%s", cipher);

    int a, b, i, j;

    // Try each key.
    for (i=0; i<12; i++) {
      for (b=0; b<26; b++) {
        printf("%d %d\t", ALIST[i], b);
        for (j=0; j<strlen(cipher); j++)
          printf("%c", (ALIST[i]*(cipher[j]-'a')+b)%26 + 'a');
        printf("\n");
      }
    }
    printf("\n");
    return 0;
}
```

Program to get frequency and solve question 4

```
#include <stdio.h>
#include <string.h>

int main() {
    char cipher[1000];
    scanf("%s", cipher);
    int freq[26], i;
    for (i=0; i<26; i++) freq[i] = 0;

    for (i=0; i<strlen(cipher); i++) {
        freq[cipher[i]-'a']++;
        printf("%c", (7*(cipher[i]-'a')+12)%26 + 'a');
    }
    printf("\n");

    for (i=0; i<26; i++)
        printf("%c %d\n", 'a'+i, freq[i]);

    return 0;
}
```