Electrical & Computer Engineering Department, University of Maryland, College Park

Spring 2007 **ENEE114** Dr. Gang Qu

Project 4: A Simple Cryptography Engine

Posted: Wednesday April 25, 2007.
Described: Thursday April 26, 2007.
Due: 11:59PM, Monday May 7, 2007.

Project Objective:

1. master the process of completing a programming project.
2. master command line argument and file I/O.
3. get familiar with string and array operations.
4. learn to use dynamic memory allocation.

Project Description:

In this project, you will implement three basic cryptography procedures for shift cipher. Cryptography refers to the use of **encryption** (or encoder) to convert plain text information into unintelligible gibberish (cipher text) for security purposes. A corresponding **decryption** algorithm (or decoder) will convert the cipher text back to the plain text with the correct decryption key. A **code breaker** tries to decode the cipher text without the key.

Shift cipher is one of the simplest cryptography systems. It encrypts the plain text message by shifting each character for fixed number of characters down the relevant alphabet. This fixed number is called the encryption key. For example, consider the English alphabet, when the encryption key is 2, *a, b, c, …, x, y, z* will be shifted to *c, d, e, …, z, a, b*, respectively. A shift cipher with encryption key 2 will convert the plain text "*project*" into cipher text "*rtqlgev*". The decoder in this case will be shifting each letter 2 position to the left, that is, replacing *a, b, c, …, x,y,z* in the cipher text by *y, z, a, …, v, w, x,* respectively. It suffices for a code breaker in this case to guess the value of the encryption/decryption key.

You need to write a single program to perform these three basic cryptography procedures at the user's choice. Assuming that *a.out* is the executable of your program:

a.out 1 input output key	encodes *input* file with *key* and write cipher text to *output*
a.out 2 input output key	decodes *input* file with *key* and write plain text to *output*
a.out 3 input output	breaks the cipher text *input* and write plain text to *output*

The code breaker procedure tries all the possible key values and matches the words in the corresponding plain text with the words in a dictionary "mydictional.txt". The key that produces the maximal number of matches will be considered as the key and will be used to generate the plain text file *output*.

Input/Output

Both plain text and cipher text will consist of characters from the following 72-letter alphabet:

0-25: lower case letters a-z
26-51: upper case letters A-Z
52-61: digits 0-9
62-71: 10 punctuations ; : , . " " ( ) \n (new line) and (single space)

The input file will be the plain text when user option is 1, and will be the cipher text when user option is 2 or 3. The output file will be the corresponding cipher text (for user option 1) and plain text (for user options 2 or 3). The key will be an integer between 0 and 71.

When user option is 3, your program needs (1) to try decoding with each of the possible key values between 0 and 71, (2) to match the words in the corresponding plain text with the words in a dictionary "mydictional.txt", and (3) to find the key that produces the maximal number of matches and use this key to generate the plain text file *output*. Your program should also print out on the display the number of matched words for each key:

```
Key     Matches
0       1
1       5
2       0
.
.
.
26      56
.
.
.
70      3
71      2

Correct Key is 26
```

The size of the input file and the dictionary is unknown and you need to use dynamic memory allocation commands if you want to store them in your program.

Project Requirements:

1. You must program using C under GLUE UNIX system and name your program **p4.c**.
2. Your program must be properly documented.
3. Submit your program **p4.c** electronically before the due time.
4. **IMPORTANT:** Your program's output, both to the output file and to the computer monitor, should be exactly the same as that produced by the master program.
5. **IMPORTANT:** Input file is critical for this project, please use ***cp*** command in UNIX to copy the input files from class webpage to your directory.

Grading Criteria:

Correctness:	80%
Good coding style:	10%
Proper documentation:	10%
Late submission penalty:	-40% for the first 24 hours No submission will be accepted after the first 24 hours.