COMP-206
Software Systems
Assignment #2
Due: Friday February 13, 2009 on Web CT at 23:55

The assignments in this course lead you, by the end of the course, to develop a web store. As we do that we also learn how to properly engineer a program using standard available tools, like GNU and the Unix environment. This assignment continues in that theme. Every good web site needs a secure login page. A secure login page implies that information is encrypted. Our first question is just that, an encryption algorithm in C. It will be eventually used in your web site. The second question asks you to create a bash program that helps you compile complex programs, while providing you some sound automated engineering procedures.

Question 1: C Programming

A semi-advanced cryptographic method is called the Grid Caesar Cipher. This method uses an integer number and a two dimensional array to encrypt messages. This method combines Simple Caesar Cipher with a transposition matrix.

Simple Caesar Cipher asks the user for an input message and an integer number. Each character of the message is shifted by N characters, where N is the inputted integer number. The shifted characters are wrapped back to the first letter of the alphabet if the shift goes past the last letter of the alphabet. For example: if our message is ZAP and our inputted integer is 2, the resulting cipher would be BCR.

A Grid Cipher is when we have a message saved in a 2D array but we write out the message by performing a matrix transposition of the array. For example, let us say we have a 10X3 matrix containing the following information:

```
 0 1 2 3 4 5 6 7 8 9
0H I   T HERE
1MR  B I L L
2B Y E
```

The transposition output of the matrix would be:

HMB IRY E TB HI EL RL E (but without any extra spaces – the extra spaces are shown here to help you see what is going on).

The Grid Caesar Cipher takes input text and stores it into an array. It then performs the standard Simple Caesar Cipher on all the characters of the array but writes out the array in its transposition.

Write a C program, that when started displays the following menu:

MAIN MENU
==========
1. Input Text
2. Encrypt
3. Decrypt
4. Exit
Selection: __

Your program will display the above menu when it starts running. The program returns to the menu after every menu selection until the user presses 4 to quit.

When the user selects option 1, the program will ask the user for the name of a text file. It will verify that the text file exists, displaying an error message and returning to the menu if it does not. If the file is valid it will then open the file and load it's contents into a 2D-character array of size 50X50, called Matrix. Make sure the text files you create are not bigger then the size of the array. Make sure to initialize the array with blank spaces, (i.e. ' ').

Note: The Matrix array is used to store both unencrypted and encrypted information. Therefore, the file loaded into the array Matrix could be an unencrypted or an encrypted text file. What you load into the array using Option 1 will depend on what you will want to do in Options 2 and 3.

Note: None alphabetic characters will not be encrypted or decrypted. In other words, they will maintain their original value after encryption and decryption.

The user then selects option 2 or 3. Options 2 and 3 use the Matrix array and assume that the user already loaded the array by using option 1, but there is no check for this. Options 2, Encrypt, will ask for an output file name and an integer number. Whatever is stored in Matrix is Grid Caesar Ciphered into that output file and also displayed on the screen. Option 3, Decrypt, ask the user for an integer number and takes what is in Matrix and reverses the Grid Caesar Cipher operations to convert the message back into its readable form. This is displayed on the screen only.

Option 4 terminates the program.

Any other data structures you may need will be up to you.

Question 2: Bash Compiling

Bash is used to simplify operations on the computer. In this question, you will make a Bash program to help when developing software in C. This is not about MAKE files, so your Bash program should not use anything you have learned about or know about MAKE files (we will be covering make files soon in this course).

The idea here is to develop your own compiler command. Its syntax will be:

$ compile cfilename –o filename –clean –backup – archive -help

WHERE:

$	is the Unix prompt
compile	is the name of your Bash program
-o filename	is a mandatory argument that indicates the name of the executable. If –o is present with other switches then –o has precedence and is executed fist before the other switches. The other switches are executed in their order of appearance.
-clean	is optional and when present deletes all the .o files
-backup	is optional and when present *copies* all the .c and .h files into your backup directory (where ever that is, overwriting what is there without prompting the user).
-archive	is optional and when present TARs the contents of your source directory with anything and everything in it. The TAR file is then *moved* into your backup directory.
-help	shows how to use the command compile. In other words, show something similar to what I have given here.

This command operates in the following way:

- Whatever options are present the –o (lowercase) must always be present or a syntax error is displayed giving the correct syntax for how to use this command. In other words it should automatically display the –help information.
- The Bash program will compile using GCC with the –o option and it will use the cfilename to compile out to the filename executable.
    - As you have seen in class, to compile a C program you use the gcc command. The gcc command needs you to provide the file names you want to compile. For example: gcc f1.c f2.c. Notice that you are not providing the file names on the command-line. This compile script will be created with gcc and the file names already present in the script.
- The –archive and –backup commands will use directory and file names defined within the script. In the case of –archive the archive file will be called ARCHIVE.TAR. For both the parameters -archive and –backup the path to your backup directory will be already part of the Bash program.

WHAT TO HAND IN

On Web CT hand in the following:

- CIPHER.C containing the entire C program in a single file.
- Also provide one input regular file and its corresponding output encrypted file. These must both be text files.
- Your Bash file called COMPILE (in lowercase)

HOW IT WILL BE GRADED

This assignment is worth 20 points:

- Question 1 is worth 10 points
  - Option 1 (3 points)
  - Option 2 (3 points)
  - Option 3 (2 points)
  - The Main program, loop and option 4 (2 points)
- Question 2 is worth 10 points
  - Parameter processing: access and on error (2 point)
  - –clean (1 point)
  - –backup (1 point)
  - –o (2 points)
  - –archive (3 points)
  - –help (1 point)