Lab 3

Substitution Cipher

Server Team

Karl Miller, Paul Shriner, Christian Messmer

# Compilation

To compile and run the program, simply run *make* in the directory that the zip file is extracted to.

A screen shot of a computer

Description automatically generated

# Module Overview

The Cipher.h/Cipher.c module contains all functions pertaining to the creation of a cipher as well as the encryption of a string.

FlexString.h/FlexString.c is used to dynamically create a string size to be able to accept a string of any size.

Colors.h simply contains VTE commands to color the text of the program.

main.c runs the program and uses the modules to run a basic encryption program using the randomized substitution cipher.

# Header Files

## Cipher.h

#ifndef Cipher\_h

#define Cipher\_h

/\*

    Class: ECET 4640-002

    Assignment: Lab Assignment 3

    Authors: Christian Messmer, Karl Miller, Paul Shriner

    Cipher.h: Function prototypes for Cipher.c

\*/

/\*\*

    @brief Generates a random cipher.

    Modular, intended to be reusable.

    @details The array is first filled with the characters between start and end. Then the array is traversed. Each element is swapped with some random other element. Each element is swapped at least once.

    @param cipher The cipher to fill.

    @param start The character to start.

    @param end The character the cipher will end on (inclusive).

    @attention Cipher is at least end-start in length.

    @attention Mutates: Fills cipher randomly with characters between start and end

\*/

void GenerateCipher(char \*cipher, char start, char end);

/\*\*

    @brief Prints the cipher.

    Modular, intended to be reusable.

    @details Prints the cipher in a series of columns describing what each character will be transformed into.

    Uses colors.h.

    @param cipher The cipher to print.

    @param start The character started on cipher

    @param length The length of the cipher

\*/

void PrintCipher(char \*cipher, char start, char length);

/\*\*

    @brief Encrypts the given string

    Modular, intended to be reusable.

    @details Replaces the string in place, mutating it. Anything out of bounds of the cipher will not be encrypted and will stay as its original character.

    @attention mutatates: encypts the string in place, destroying the original characters

    @param string String to ecrypt.

    @param length Length of `string`

    @param cipher Cipher to use for encrypting the string. Must be (start-end)+1 in size.

    @param start The first character the cipher uses

    @param end The last character the ciper uses

\*/

void EncryptString(char \*string, int length, char \*cipher, char start, char end);

#endif

## FlexString.h

#ifndef FlexString\_h

#define FlexString\_h

/\*

    Class: ECET 4640-002

    Assignment: Lab Assignment 3

    Authors: Christian Messmer, Karl Miller, Paul Shriner

    FlexString.h: Function prototypes for FlexString.c

\*/

/\*

    FlexString is used to create a heap-allocated string that resizes as necessary for longer input strings.

\*/

#include <stdio.h>  // for reading input

#include <stdlib.h> // for memory allocation

/\*\*

    Fills a malloced string with input, reallocating if necessary.

    Called by FlexString

    @param string A char \*\*; it points to the string that will be populated.

    @param capacity A pointer to the capacity desired of the string. If 0 or less, it will be set to 11.

    @param file A pointer to a file. Can be stdin if reading from the console.

    @attention Precondition: String has already been malloced.

    @attention Mutates: May reallocate and change value pointed to by string.

    @attention Mutates: May change value pointed to by capacity.

    @return The number of characters read (not including new line, and null-terminator)

\*/

int FlexString\_Read(char \*\*string, size\_t \*capacity, FILE \*file);

/\*\*

    @brief Creates and fills a malloced string. Then calls FlexString\_Read to read from input, resizing as necessary.

    Called by main.

    @param string A pointer to a char\*, referencing the string that will be populated.

    @param capacity A pointer to the capacity desired of the string. If 0 or less, it will be set to 11.

    @param file A pointer to a file. Can be stdin if reading from the console.

    @attention Mutates: May reallocate and change value pointed to by string.

    @attention Mutates: May change value pointed to by capacity.

    @return The size of characters read, excluding the null-terminator and new-line (if present)

\*/

int FlexString(char \*\*string, size\_t \*capacity, FILE \*file);

#endif

## colors.h

/\*

    Class: ECET 4640-002

    Assignment: Lab Assignment 3

    Authors: Christian Messmer, Karl Miller, Paul Shriner

    colors.h: Define color macros for use with printing text to the console

    Acknowledgements/Credits:

        1. https://www.man7.org/linux/man-pages/man4/console\_codes.4.html

\*/

#ifndef colors\_h

#define colors\_h

/\*

    Karl's magic color macros.

    These use Virtual Terminal escape sequences to trigger color changes on the console when printed.

    See 1 in Acknowledgements/Credits for more information.

\*/

#define COLOR\_RED "\e[38;2;255;75;75m"

#define COLOR\_BLUE "\e[38;2;0;240;240m"

#define COLOR\_GREEN "\e[38;2;0;240;0m"

#define COLOR\_YELLOW "\e[38;2;255;255;0m"

#define COLOR\_GRAY "\e[38;2;224;224;224m"

#define COLOR\_BOLD "\e[1m"

#define COLOR\_RESET "\e[0m"

#endif

# Contributions

Karl re-used and modified the color codes and flex string from previous assignments. He contributed to writing this document.

All group members worked together and simultaneously using VSCode LiveShare to write the initial version of the cipher.

Christian created first draft of document and make the encryption code.

Paul adjusted formatting of the output and adjusted colors to look nicer and implemented make clean in the make file.