Programming Assignment 1
Vigènere Cipher

Michael McAlpin

May 31, 2022

1 Vigènere Cipher

In this assignment you'll write a program that encrypts the alphabetic letters in a file using the Vigènere cipher. Your program will take two command line parameters containing the names of the file storing the encryption key and the file to be encrypted. The program must generate output to the console (terminal) screen as specified below.

1.1 Command line parameters

1. Your program **must compile** and **run** from the **terminal command line**.
2. Input the required file names as command line parameters. Your program may NOT prompt the user to enter the file names. The first parameter must be the name of the encryption key file, as described below. The second parameter must be the name of the file to be encrypted, as also described below. The sample run command near the end of this document contains an example of how the parameters will be entered.
3. Your program should open the two files, echo the processed input to the screen, make the necessary calculations, and then output the ciphertext to the console (terminal) screen in the format described below.

Note
If the plaintext file to be encrypted doesn't have the proper number (512) of alphabetic

characters, pad the last block as necessary with the lowercase letter **x**. Make sure that all the input characters are lower case only.

1.2 Formats

1.2.1 Encryption Key File Formats

The encryption key is plain text that may contain upper and lower case letters, numbers, and other text. The input must be stripped of all non-alphabetic characters. Please note that the input text must be converted to contiguous lower case letters to simplify the encryption process.

1.2.2 Encryption Plaintext File Formats

The file to be encrypted can be any valid text file with no more than 512 letters in it. (Thus, it is safe to store all characters in the file in a character array of size 512, including any pad characters.) Please note that the input text file will also generally have punctuation, numbers, special characters, and whitespace in it, which should be ignored. You should also ignore whether a letter is uppercase or lowercase in the input file. In order to simplify the encryption, all letters should be converted to **lower case** letters. In the event the plaintext input file is ***less than 512*** characters, pad the input file with a lowercase **x** until the 512 character input buffer is full.

1.2.3 Output Format

The program must output the following to the console (terminal) screen, also known as `stdout`:

1. Echo the lowercase alphabetic text derived from the input key file.
2. Echo the lowercase alphabetic text derived from the input plaintext file.
    - Remember to pad with **x** if the processed plaintext is less than 512 characters.
3. Ciphertext output produced from encrypting the input key file against the input plaintext file.

The output portion of each input file should consist of only lowercase letters in rows of exactly 80 letters per row, except for the last row, which may possibly have fewer. These characters should correspond to the ciphertext produced by encrypting all the letters in the input file. Please note that only the alphabetic letters in the input plaintext file will be encrypted. All other characters should be ignored.

1.2.4 Program Execution

The program, **pa01** expects two inputs at the command line. The first parameter is the name of the key file, an 8 bit ASCII string terminated by a a newline character. The second parameter is the name of the plaintext file, an 8 bit ASCII string terminated by a a newline character. Note that

these input files may contain non-alphabetic characters (numbers, punctuation, white spaces, etc.). The valid inputs, as discussed previously, may be any alphabetic character, and should be converted to lower case.

1.2.5 Program execution - example

Note that the commands below are also outlined later on in this document for either **c**, **c++**, or **Java**. Also note that the first parameter is the *key* filename and the second parameter is the *plaintext* filename.

```
systemPrompt$ gcc -o pa01 pa01.c
systemPrompt$ ./pa01 kX.txt pX.txt
```

1.3 Submission instructions

You must submit this assignment in **Webcourses** as a source file upload. Note that all submissions will be via Webcourses. The submitted programs will be **tested** and **graded** on **Eustis**.

1.3.1 Code Requirements

- Header - the following *Header Usage instructions/comments* comment block should be at the beginning of the source file.

```
/*============================================================================
 |   Assignment:  pa01 - Encrypting a plaintext file using the Vigenere cipher
 |
 |       Author:  Your name here
 |     Language:  c, c++, Java, go, python
 |
 |   To Compile:  javac pa01.java
 |                gcc -o pa01 pa01.c
 |                g++ -o pa01 pa01.cpp
 |                go build pa01.go
 |                python pa01.py
 |
 |   To Execute:  java   -> java pa01 kX.txt pX.txt
 |           or   c++    -> ./pa01 kX.txt pX.txt
 |           or   c      -> ./pa01 kX.txt pX.txt
 |           or   go     -> ./pa01 kX.txt pX.txt
 |           or   python -> python pa01.py kX.txt pX.txt
 |                          where kX.txt is the keytext file
 |                            and pX.txt is plaintext file
 |
 |         Note:  All input files are simple 8 bit ASCII input
 |
 |        Class:  CIS3360 - Security in Computing - Summer 2022
 |   Instructor:  McAlpin
 |     Due Date:  per assignment
 |
 +============================================================================*/
```

- The following *Academic Integrity Statement* comment block should be at the end of the source file.

```
/*============================================================================
 |     I [your name] ([your NID]) affirm that this program is
 | entirely my own work and that I have neither developed my code together with
 | any another person, nor copied any code from any other person, nor permitted
 | my code to be copied  or otherwise used by any other person, nor have I
 | copied, modified, or otherwise used programs created by others. I acknowledge
 | that any violation of the above terms will be treated as academic dishonesty.
 +============================================================================*/
```

1.4 Program Notes and Hints

Your program must read in an input plaintext file that may contain uppercase letters, lowercase letters and non-letter characters. Your program must distinguish between these three groups so that only the letters get encrypted. All non-letter characters in the file are simply skipped and not counted as part of the plaintext. Please note that although both upper case and lower case letters will be encrypted, your program should convert an upper case input letter to the corresponding lower case letter, i.e., it should convert an **A** to an **a**, and so on for the whole alphabet.
One possible breakdown to solve this problem is as follows:

1. Write a section of code or function that reads only the upper and lower case letters in the input file into an char array of size 512, storing only the appropriate lowercase letters in the character array.

2. Write a section of code or function that takes as input the array from section 1 and the encryption key and produces an array of ciphertext storing only lowercase letters.

3. Write a section of code or function that takes as input the array storing the ciphertext and outputs it to the screen in the format specified. Additional functions or code will be needed to echo the input key and plaintext files.

1.5 Sample inputs and outputs

1.5.1 Sample Key File

"I think and think for months and years. Ninety-nine times, the conclusion is false. The hundredth time I am right." - Albert Einstein "Imagination is more important than knowledge. For knowledge is limited, whereas imagination embraces the entire world, stimulating progress, giving birth to evolution." - Albert Einstein[1]

1.5.2 Sample Plaintext File

"Fall in love with some activity, and do it! Nobody ever figures out what life is all about, and it doesn't matter. Explore the world. Nearly everything is really interesting if you go into it deeply enough. Work as hard and as much as you want to on the things you like to do the best. Don't think about what you want to be, but what you want to do. Keep up some kind of a minimum with other things so that society doesn't stop you from doing anything at all." - Richard Feynman[2]

1.5.3 Sample Ciphertext Output File

```
ntstvxlbyxdqgrxcdqopmpnigbyrdugvbasumqgrzxzrmynyiuchvhbsbzvnwnsldptisbnnqumwwvxa
zxuafkmxlqpwpvvmlmjgkplammrrgrvxzmgpazuzwzqpzcriamxyefdvbctjbuylczxgceehhkttqpva
czlzkyorwhszpatlnsfcqueezfuyefmassampvxdwervqhcxcvemwquiyshvwlvuvobuoosruvnhacoe
shcknneussxfcgoaeblwndiadtbghrmrzzdjaardpfdbiyqieazczabruwglxzflagnwucgjlnkwqvml
ddzwwgawaicbfyikvflamvgmegzobnrbxrepzvuaezqnqytunnqflkfpjlobfjmloqxkqqexkhkltiba
dbclohkltibadbfpifjfqbatebobxpfjxdfkxqflkbjyoxzbpqebbkqfobtloiapqfjrixqfkdmoldob
ppdfsfkdyfoqeqlbslirqflkxiyboqbf
```

[1] This is k2.txt in the Programming Assignment 1 ZIP file.
[2] This is p2.txt in the Programming Assignment 1 ZIP file.

1.6 Grading

Scoring will be based on the following rubric:

Table 1.1: Grading Rubric

Deduction	Description
-100	Cannot compile on *eustis*
-100	Your program does not successfully compile from the command line with one of these commands: C program: `prompt$gcc -o pa01 pa01.c` C++ program: `prompt$g++ -o pa01 pa01.cpp` Go program: `prompt$go build pa01.go` Python program: `prompt$python pa01.py` Java program: `prompt$javac pa01.java` Note: *If you are submitting a Java program, the class file must be named "pa01.java" and the class name must be "pa01".*
-100	Cannot read input files specified on command line
-100	Cannot write output to `stdout`
- 90	Your program does not run from the command line without error or produces no output.
- 70	The program compiles, runs, and outputs the key and input file, but crashes thereafter or produces no encryption output.
- 50	Runs without crashing but produces **no meaningful** encryption output
- 20	Fails to produce valid all lowercase alphabetic **key**
- 20	Fails to produce valid all lowercase alphabetic **plain text**
- 20	Fails to produce valid lowercase alphabetic **key**,**plaintext**, **ciphertext** - formatted to 80 columns per line
- 25	Does not have *Header Usage instructions/comments* statement
- 25	Does not have *Academic Integrity* statement
Start with 100 points and deduct per the schedule above	