ECE 471 Project 1 – Cracking Ciphers

Table of Contents

[Overview of Methods 4](#_Toc475219473)

[Shift Cipher 4](#_Toc475219474)

[*Method - Detailed Steps* 4](#_Toc475219475)

[*Interesting Observations* 4](#_Toc475219476)

[*Final Results* 4](#_Toc475219477)

[Substitution Cipher 4](#_Toc475219478)

[*Method - Detailed Steps* 4](#_Toc475219479)

[*Interesting Observations* 4](#_Toc475219480)

[*Final Results* 4](#_Toc475219481)

[Vigenere Cipher 4](#_Toc475219482)

[*Method - Detailed Steps* 4](#_Toc475219483)

[*Interesting Observations* 4](#_Toc475219484)

[*Final Results* 4](#_Toc475219485)

[Permutation Cipher (Columnar Transposition) 4](#_Toc475219486)

[*Method - Detailed Steps* 4](#_Toc475219487)

[*Interesting Observations* 4](#_Toc475219488)

[*Final Results* 4](#_Toc475219489)

[One-Time Pad 4](#_Toc475219490)

[*Method - Detailed Steps* 4](#_Toc475219491)

[*Interesting Observations* 4](#_Toc475219492)

[*Final Results* 4](#_Toc475219493)

[What We Learned 4](#_Toc475219494)

[Appendix 4](#_Toc475219495)

[How to Use this Program 4](#_Toc475219496)

[Explanation of Files (Initial Files Created) 6](#_Toc475219497)

[*createProbabilityData.m* 6](#_Toc475219498)

[*decryptVigenere.m* 6](#_Toc475219499)

[*findBigrams.m* 6](#_Toc475219500)

[*findTrigrams.m* 6](#_Toc475219501)

[*gcdList.m* 6](#_Toc475219502)

[*icTest.m* 6](#_Toc475219503)

[*initial.m* 7](#_Toc475219504)

[*kasiskiTest.m* 7](#_Toc475219505)

[*loadFile.m* 7](#_Toc475219506)

[*monoFrequency.m* 7](#_Toc475219507)

[*plotBi.m* 7](#_Toc475219508)

[*plotIC.m* 8](#_Toc475219509)

[*plotMono.m* 8](#_Toc475219510)

[*shiftTest.m* 8](#_Toc475219511)

[*trigramIntervals.m* 8](#_Toc475219512)

[*unVigenere.m* 8](#_Toc475219513)

[*unshift.m* 8](#_Toc475219514)

# Overview of Methods

## Shift Cipher

### Method - Detailed Steps

### Interesting Observations

### Final Results

## Substitution Cipher

### Method - Detailed Steps

### Interesting Observations

### Final Results

## Vigenere Cipher

### Method - Detailed Steps

### Interesting Observations

### Final Results

## Permutation Cipher (Columnar Transposition)

### Method - Detailed Steps

### Interesting Observations

### Final Results

## One-Time Pad

### Method - Detailed Steps

### Interesting Observations

### Final Results

# What We Learned

# Appendix

## How to Use this Program

To start using this program, run Matlab.

In the command line, use the loadFile function and enter in the address of the file you wish to decrypt.

>> loadFile(‘C:/Users/user/Desktop/ciphertexts/cipher4.txt’)

After the file has been loaded, run the initial function to determine which cipher is used on the ciphertext.

>> initial

The initial function will print to the command line the type of cipher and relevant information of the cipher.

Vigenere cipher with length 9

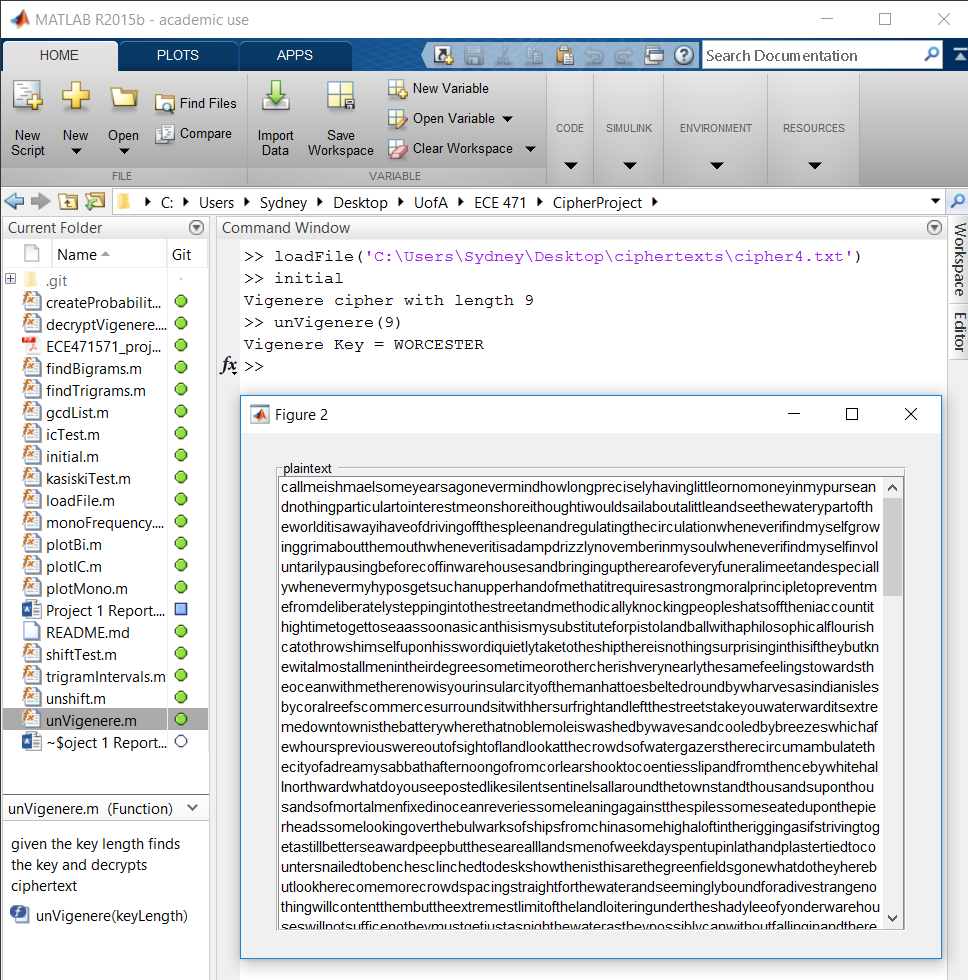
Available decryption techniques are shift and Vigenere. In order to decrypt, simply run the corresponding ‘un’ cipher function.

>> unVigenere(9)

Vigenere Key = WORCESTER

The program will then print out relevant information to the decrypted cipher, and in a separate window the program will print the plaintext of the cipher.

Screenshot of usage:



Demonstration of How to Use This Program

## Explanation of Files (Initial Files Created)

### createProbabilityData.m

Inputs: None

Outputs: None

Description: This function places probability data taken from practicalcryptography.com into the global variables monoProbability, biProbability, triProbability and wordProbability.

Usage: The user does not use this program. It is internally used by other functions.

### decryptVigenere.m

Inputs: key – string (this is the key to the cipher)

Outputs: Window display with plaintext

Description: This function decodes the cipher given the key.

Usage: The user does not use this program. It is internally used by other functions.

### findBigrams.m

Inputs: None

Outputs: bigrams - 3 column matrix where the first column and second column are the number values for each letter (0 is A, 1 is B, etc.) and the third column is the relative frequency. The columns are sorted according to decreasing relative frequency.

Description: This function will output the normalized relative frequencies of all the bigrams in a given ciphertext file.

Usage: The user will run the command findBigrams in the Matlab command line.

### findTrigrams.m

Inputs: None

Outputs: trigrams - 4 column matrix where the first column, second column and third column are the number values for each letter (0 is A, 1 is B, etc.) and the third column is the relative frequency. The columns are sorted according to decreasing relative frequency.

Description: This function will output the normalized relative frequencies of all the trigrams in a given ciphertext file.

Usage: The user will run the command findTrigrams in the Matlab command line.

### gcdList.m

Inputs: n – array of numbers

Outputs: v – the gcd of a list of numbers

Description: This function will take in an array of numbers and output the gcd of that array of numbers.

Usage: The user does not use this program. It is internally used by other functions.

### icTest.m

Inputs: numberOfRows – key length of the cipher

Outputs: ic – index of coincidence value

Description: This function calculates the index of coincidence for a cipher given the key length.

Usage: The user does not use this program. It is internally used by other functions.

### initial.m

Inputs: None

Outputs: None

Description: This function will determine what the type of cipher used is after a file has been loaded using the loadFile function.

Usage: The user will run the command initial in the Matlab command line.

### kasiskiTest.m

Inputs: None

Outputs: None

Description: This function will determine the top five most frequent trigrams in the given ciphertext and print out their locations.

Usage: The user will run the command kasiskiTest in the Matlab command line.

### loadFile.m

Inputs: filename – the directory path to the file containing the ciphertext

Outputs: None

Description: This function enters in the ciphertext in the file into a global variable named cipherText and also computes the frequency tables for monograms and bigrams using the findBigrams and monoFrequency(cipherText) functions.

Usage: The user will run the command loadFile(‘file\_address’) command. This is the first command run before any other function is used.

### monoFrequency.m

Inputs: cipherText – all the ciphertext contained in the input file

Outputs: sortedFrequencyTable – table that contains the frequency for individual letters

Description: This function calculates the frequency for each individual letter in the given ciphertext and creates a table that has each letter and its relative frequency.

Usage: The user does not use this program. It is internally used by other functions.

### plotBi.m

Inputs: numberOfBigrams – the number of bigrams the user wishes to plot

includeBigramFrequency – true if user wishes to plot English language frequency of bigrams against the frequency of the bigrams in the ciphertext

Outputs: Plot of the frequency of the bigrams

Description: This function plots the frequency of the most common bigrams in the ciphertext, and it also has the option to plot this frequency against the frequency of the most common bigrams in the English language.

Usage: The user will run the command plotBi(numBigrams, true/false) in the Matlab command line.

### plotIC.m

Inputs: maxValue – the max number to find the index of coincidence for

Outputs: None

Description: This function will plot the index of coincidences from 1 to maxValue; a blue line indicates where the index of coincidence should be for a Vigenere cipher, the red line indicates where random is. This is a manual way for the user to determine the index of coincidence.

Usage: The user will run the command plotIC(value) in the Matlab command line.

### plotMono.m

Inputs: includeLetterFrequency – true if user wishes to plot English language frequency of monograms against the frequency of the monograms in the ciphertext

Outputs: Plot of the frequency of the monograms

Description: This function plots the frequency of all the monograms in the ciphertext, and it also has the option to plot this frequency against the frequency of monograms in the English language.

Usage: The user will run the command plotMono(true/false) in the Matlab command line.

### shiftTest.m

Inputs: None

Outputs: score – the highest sum value the function could find

shift – the shift value that produced score

Description: This function will determine the shift value that produces a sum of around 0.065.

Usage: The user does not use this program. It is internally used by other functions.

### trigramIntervals.m

Inputs: trigram – array of all the counts of all the trigrams in the ciphertext

Outputs: intervals – list of intervals of locations for the trigrams

Description: This program creates a list of the locations of trigrams in the given ciphertext.

Usage: The user does not use this program. It is internally used by other functions.

### unVigenere.m

Inputs: keyLength – length of the key determined by the initial function

Outputs: Window containing the decrypted plaintext, the key

Description: This function decodes the given ciphertext given the length of the key

Usage: The user will run the command unVigenere(keyLength) after running the initial command in the Matlab command line.

### unshift.m

Inputs: shiftAmount – the value of the shift amount determined by the initial function

Outputs: Window containing the decrypted plaintext

Description: This function decodes the given ciphertext given the shiftAmount.

Usage: The user will run the command unshift(shiftAmount) after running the initial command in the Matlab command line.