650.445/600.454: Practical Cryptographic Systems August 31, 2017

Assignment 1

Instructor: Matthew Green *Due: 11:59 pm September 13, 2017*

Name: _____

The assignment must be completed individually. You are permitted to use the Internet and any printed references, though your code must be your own!

Please submit the completed assignment via Blackboard.

Problem 1: Classical encipherment and decipherment (20 points)

For this problem you will implement software that programmatically enciphers and deciphers messages using the Bellaso (Vigenère) cipher. Your programs must be written in Go. In either case, your programs should run at the command line using the command line syntax given below.

```
vigenere-encrypt <encipherment key> <plaintext filename>
vigenere-decrypt <decipherment key> <ciphertext filename>
```

The key field should consist of a string of uppercase letters up to 32 characters in length. The input file field should contain the name of a text file (of size up to a maximum of 100KB) containing plaintext (resp. ciphertext). Your program should strip out and ignore all characters in the file that are not letters ([a-zA-Z]). The program output should consist only of uppercase letters with no white space, unless you wish to add *optional* newline characters for formatting. Example usage on a Unix system is:

```
vigenere-encrypt PLEBISCITE plaintext.txt > ciphertext.txt
vigenere-decrypt PLEBISCITE ciphertext.txt > recovered_plaintext.txt
```

Problem 2: Finding key lengths (30 points)

Now that you've written a tool to compute Vigenère ciphertexts, write a second tool that uses the *index of coincidence* method to find the length of the key used in a Vigenère ciphertext. Your tool should take an enciphered file with an unknown key, and output a guess for the length of the key. Your tool should be reliable for keys of up to 20 characters in length, when run on English language texts of at least 20,000 characters. Your tool should output a number that represents the most likely guess for the key length. The required command-line syntax is:

```
vigenere-keylength <ciphertext filename>
```

Problem 3: Full cryptanalysis (45 points)

Armed with a guess for the key length used in a given ciphertext, write a tool that attempts to complete decrypt a Vigenère ciphertext using frequency analysis, or make a best guess for the decryption key *using ciphertext only*. Your tool should output the *recovered decipherment key* for the analyzed ciphertext (or a reasonably-sized list of keys if multiple candidates are discovered).

```
vigenere-cryptanalyze <ciphertext filename> <key length>
```

You may combine the answers to Part 2 and Part 3 into a single program that omits the key length argument. Remember that you do not need to perfectly decrypt *every* ciphertext, but you should be able to at least partially decrypt some English-language messages.

Problem 4: LOPs (5 points)

Imagine that we modify the Vigenère encipherment to use a single key that is at least as long as the message. Explain why this cipher might be more secure, and what conditions you would place on the keys to ensure it.