

# **COMPUTER PROGRAMMING IN JAVA**

COLUMBIA UNIVERSITY HIGH SCHOOL SCIENCE HONORS PROGRAM

## Intro to Cryptology: Code Breaking Assignment 2008 May 03 Sat

We have obtained a large body of encoded messages, and we wish to crack them. Here's what we know:

1. The first group of documents was encrypted with GAMMA level security, the lowest and easiest to break level of security. We know that GAMMA level security uses a shift cipher. For some of these documents, we know the shift value, but for others we don't.
2. The second group of documents was encrypted with BETA level security, the second level of security, and tougher than GAMMA. We have learned that BETA level security entails a "keyed" and "reversed" Caesar cipher; see below for details. For some of these documents we have parts of the key; this should be helpful in frequency analysis.
3. The last group of documents was encrypted with ALPHA level security, the highest and toughest level. Messages protected with ALPHA level security uses a Vigenère cipher. We know nothing about these documents.

It is known that the last group of documents (the ALPHA protected documents) contains a secret passphrase. Cracking the ALPHA documents directly is difficult (but feasible); fortunately, we have other means at our disposal. Some of the GAMMA documents provide clues to the keys that were used to encrypt the BETA documents, and some of the BETA documents in turn provide hints about the key used to encrypt the ALPHA documents. Thus, if we can crack each level of security in turn, we will have many more hints with which to perform cryptanalysis of the ALPHA documents.

Your team's mission is to extract the secret passphrase from the ALPHA documents. Once extracted, communicate it to me and I will be able to confirm or deny its correctness.

I recommend that your team follow these assignments:

- **COORDINATOR:** One member of the team focuses on decrypting the GAMMA documents with the known keys, and extracting the relevant clues from the decrypted documents. This member should also keep track of all the clues the team uncovers, as well as trying to figure out the hints.
- **FREQUENCY ANALYZER:** One member of the team writes a frequency analyzer: given a String, outputs the frequency of each letter.

- **ALPHABET GENERATOR:** One member of the team writes an alphabet generator for BETA documents. That is, given some key, write a method which populates an array of size 26 with all the unique letters (in order) from the key plus the alphabet in reverse.
- **CRYPTANALYSIS:** This position is **optional**. Implement the cryptanalysis technique of *Kasiski examination* for breaking the ALPHA documents. Alternatively, the last member of your team can either help the others with organizing the information, extracting plaintext, solving the hints, or help implement some of the programs.

To facilitate your mission, you have been given the following tools:

- A Java program that can encrypt or decrypt GAMMA documents given the shift value.

To decrypt a GAMMA document, type:

```
java Cipher -d gamma <some number> <name of file>
```

If you wish to encrypt your own documents, replace the “-d” with “-e”

- The beginnings of a FrequencyAnalyzer class. The class currently only reads in a file and prints out its contents. To aid your cryptanalysis, you should write the code that counts the frequency of each letter in the text and print the percentage occurrence.
- The beginnings of a KeyedReversedCaesarAlphabet class. Specifically, you are to write code that populates the plainAlpha and cipherAlpha String arrays.

The plainAlpha array should contain the letters “a” to “z” in order; so plainAlpha[0] = “a”, plainAlpha[1] = “b”, plainAlpha[24] = “y”, plainAlpha[25] = “z”, etc.

The cipherAlpha array should contain the unique letters from the key, in order, followed by the remainder of the alphabet, in reverse, from “z” to “a”. For example, if my key was “cannon”, the cipherAlpha array should contain:

```
c a n o z y x w v u t s r q p m l k j i h g f e d b
```

Once you have finished writing it, test it by trying to decrypt some of the BETA documents:

```
java Cipher -d beta <some key> <name of file>
```

Again, to decrypt, replace “-d” with “-e”.

Finally, here are some hints:

1. All text is in lower case.
2. Not all of the documents are relevant, and sometimes the clue might be hidden in the middle of a large body of text, so look carefully, and be sure to decrypt *all* the ciphertexts!
3. You can cast chars to String by “adding” an empty string to it:

```
char someChar = 'a';  
String someCharAsString = someChar + ""; // someCharAsString is now "a"
```

4. We know that the documents in the third GAMMA block use a shift of 1. That is, if you do the following:

```
java Cipher -d gamma 1 ciphertext/gamma/g-03-01.q
```

You should be able to read the decrypted message. Try the same with g-03-02.q, g-03-03.q, ..., g-03-06.q

5. To download the package, execute the following commands in CUNIX:

```
wget http://www.cs.columbia.edu/~mwc2110/shp/intro-crypto/package.zip  
unzip package.zip
```

(Thanks to CS1004 for the inspiration to this assignment)