SCIE1000 Project, Semester 1 2011

Topic 1: Cryptography

1 Introduction to Cryptography:

1.1 Science

Cryptography is the science of using mathematics to hide information. Cryptography allows us to store sensitive information, or to transmit it over insecure networks (such as the internet) so that it can only be read by the intended recipient. *Encryption* is the process of converting readable data (called the *plaintext*) into a form which hides its content, called the *ciphertext*. *Decryption* is the reverse process, with a ciphertext converted back into the corresponding plaintext.

A *cipher* is a mathematical function used in the encryption and decryption processes. Most ciphers use a *secret key* when encrypting, and different keys will typically encrypt a given plaintext into different ciphertexts. The key is usually only known by the person who encrypts the data, and the intended recipient. The secrecy of the key ensures that even if an eavesdropper were to intercept the transmitted data, they would be unable to decrypt it. In general the security of encrypted data is dependent on two factors: the strength of the cipher, and the secrecy of the key.

The most common type of cipher is the *substitution cipher*, where we use the key to encrypt the plaintext one letter at a time, replacing each letter by another letter. In this project, we shall only consider plaintexts and ciphertexts written using the 26 lower-case English characters. We will ignore spaces and punctuation (since these can usually be determined from the context), and will assume that the plaintext is written in English.

The process of trying to derive the meaning of the ciphertext without knowing the key is known as *cryptanalysis*. This is often called “breaking” or “cracking” the cipher. The cipher we will consider in this assignment is the *General Mixed Alphabet* (GMA), and a few of its special cases. The GMA is one of the more commonly used ciphers from history, due to its ease, simplicity and large number of keys. As we shall see in this project, however, it is relatively easy to crack this cipher.

1.2 Mathematics

It is usual to study cryptography in terms of *numbers* rather than letters. To do so, we need to use *modular arithmetic*. Modular arithmetic is concerned with finding the remainder of integer division with respect to some other given integer n, called the *modulus*. When you divide any integer by n, the remainder is always an integer between 0 and $n-1$ inclusive. Hence we say that every integer is *equivalent* to a value between 0 and $n-1$, modulo n. Mathematically, we write *equivalent* as $\equiv$.

For example, the elements of the integers modulo (often shortened to “mod”) 26 are $0, 1, \ldots, 24, 25$. Then 31 is equivalent to 5 modulo 26, written $31 \equiv 5 \bmod 26$, because when 31 is divided by 26 the remainder is 5. Similarly, we also have $57 \equiv 5 \bmod 26$.

In the following table, all integers within a column are equivalent modulo 26, because they all have the same remainder when divided by 26. Every integer, no matter how large or small, can be written in exactly one of the columns. In this project, we will always be working modulo 26.

-26	-25	-24	...	-3	-2	-1
0	**1**	**2**	...	**23**	**24**	**25**
26	27	28	...	49	50	51
52	53	54	...	75	76	77

Due to the ‘wrap-around’ property of modular arithmetic, basic operations such as addition, subtraction and multiplication behave slightly differently to standard operations. Essentially, the operation can be performed as usual, and then the remainder calculated at the end. Alternately, remainders can be calculated at any time during the calculations, if they make the working easier.

Here are some examples to illustrate how modular addition, subtraction and multiplication work.

Example (modulo 26)	Calculations
$12 + 5 \equiv \mathbf{17}$	
$25 + 1 \equiv \mathbf{0}$	$25 + 1 = 26 = 1 \times 26 + \mathbf{0}$
$2 - 5 \equiv \mathbf{23}$	$2 - 5 = -3 = -1 \times 26 + \mathbf{23}$
$38 + 109 \equiv \mathbf{12} + \mathbf{5} \equiv 17$	$109 = 4 \times 26 + \mathbf{5}$
$46 + 32 \equiv \mathbf{20} + \mathbf{6} \equiv 26 \equiv \mathbf{0}$	$26 = 1 \times 26 + \mathbf{0}$
$2 \times 12 \equiv \mathbf{24}$	$24 = 0 \times 26 + \mathbf{24}$
$21 \times 6 = 126 \equiv \mathbf{22}$	$126 = 4 \times 26 + \mathbf{22}$
$30 \times 27 = 810 \equiv \mathbf{4}$	$810 = 31 \times 26 + \mathbf{4}$
$30 \times 27 \equiv \mathbf{4} \times \mathbf{1} = 4$	$30 \equiv \mathbf{4}$ and $27 \equiv \mathbf{1}$

Another important concept to consider when working modulo n is the *multiplicative inverse* of a given integer. For an integer x, its multiplicative inverse modulo n (if one exists), denoted x^{-1}, is the number such that $x \times x^{-1} \equiv 1$ modulo n. For example, the multiplicative inverse of 5 modulo 26 is 21, because $5 \times 21 \equiv 1$ modulo 26 (because $5 \times 21 = 105 = 4 \times 26 + 1 \equiv 1$ modulo 26).

(It is important to note that in modular arithmetic, a^{-1} **does not mean** $1/a$. In fact, we have not defined division at all.)

Not all numbers have a multiplicative inverse modulo n. In general, a number will only have an inverse if it does not share any common factors with the modulus n (apart from the common factor 1). Since 26 has the factors 2 and 13, this means that even numbers, and the number 13, do not have an inverse modulo 26.

Multiplicative inverses are useful in solving for x equations of the form $a \times x \equiv b$ modulo n. First, calculate the inverse of a, if it exists, and then multiply both sides of the equation by a^{-1}, giving $a^{-1} \times a \times x = 1 \times x \equiv a^{-1} \times b$ modulo n. For example, to solve $23 \times x \equiv 2$ modulo 26, we proceed as follows. First, note that the multiplicative inverse of 23 is 17 (mod 26), because $23 \times 17 = 390 = 26 \times 15 + 1 \equiv 1$. Then,

$$\begin{aligned}
\text{since} \quad 23^{-1} &\equiv 17 \text{ modulo } 26, \\
23 \times x &\equiv 2 \text{ modulo } 26 \quad \text{means that} \\
17 \times 23 \times x &\equiv 17 \times 2 \text{ modulo } 26. \\
\text{Then since} \quad 17 \times 23 &\equiv 1 \text{ modulo } 26, \\
x &\equiv 34 \text{ modulo } 26, \text{ so} \\
x &\equiv 8 \text{ modulo } 26.
\end{aligned}$$

1.3 Python

In Python, $a \bmod n$ can be calculated using the command: `a % n`. To plot a bar graph (or histogram) where the x value for each bar is in the array `bins` and the height of each bar is in the array `heights`, use the following commands:

```
figure()
bar(bins, counts)
show()
```

1.4 Questions:

(0) (0 marks) A number of the following questions vary between students. Let T be the two digit number corresponding to the **last two digits of your student number**, so T is a number between 00 and 99 (inclusive). For example, if your number were 42136702, then $T = 02$. (It is important that you use the correct value of T for your student number; if you do not, you will lose a large number of marks. If you have any questions, please ask a tutor.)

1. Solve the following modular arithmetic problems by hand, showing all working:

(a) (1 mark) Evaluate:

(i) $100 \bmod 26$

(ii) $126 \bmod 26$

(iii) $13 \bmod 26$

(iv) $-5 \bmod 26$

(b) (1 mark) Solve:

(i) $5 + 10 \bmod 26$

(ii) $13 - 16 \bmod 26$

(iii) $32 + 46 \bmod 26$

(iv) $26 + 52 + 78 + 104 + 130 \bmod 26$

(c) (1 mark) In each of the following cases state whether the multiplicative inverse exists, and find the inverse whenever it exists.

(i) $3 \bmod 26$

(ii) $1 \bmod 26$

(iii) $16 \bmod 26$

(iv) $19 \bmod 26$

(v) $11 \bmod 26$

(d) (1 mark) Evaluate

(i) $25 \times 9 - 16 \times 22 \bmod 26$

(ii) $9 \times 11 - 3 \times 5 \bmod 26$

(e) (2 marks) Find the values of a which satisfy each of the following expressions.

(i) $5a \equiv 23 \bmod 26$

(ii) $3a \equiv 1 \bmod 26$

2. Write down your student number in full, then answer the following questions:

(a) (1 mark) Let T be the sum of the digits in your student number, modulo 26, and let N be your student number, modulo 26. Find T and N.

(b) (1 mark) Which (if any) of T and/or N have multiplicative inverses modulo 26? Justify your answer briefly.

(c) (1 mark) Find the **first integer larger** than your student number for which the corresponding values of both T and N have multiplicative inverses modulo 26. Again, justify your answer briefly.

2 The Caesar Cipher

2.1 Science

The *Caesar cipher* is a famous, ancient cipher. It has existed in various forms since at least the time of Ancient Rome, and it was still used regularly in Europe up until the time of the Renaissance. The name comes from its most famous version, which was created by Julius Caesar. In the original form, each plaintext letter was encrypted to a ciphertext letter according to the following table.

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

That is, an ‘a’ in the plaintext is encrypted to ‘d’ in the ciphertext, and so on. This is equivalent to taking the standard English alphabet and shifting it to the right by three characters (wrapping around at the end).

In the more general form of the cipher, encryption occurs by shifting each plaintext character to the right by n characters, with $0 \leq n \leq 25$. Of course, a shift of $n = 0$ offers no security at all (since the ciphertext is then exactly the same as the plaintext), whereas there is no point considering a shift $n > 25$ as this is equivalent to shifting the alphabet by $n \bmod 26$. The first letter of the cipher alphabet (‘d’ in the original form) is the key of the cipher.

Example

We can encrypt Julius Caesar’s famous saying “Veni, Vidi, Vici” using a Caesar cipher with a key of ‘f’ as follows:

- Re-write the text to be enciphered to just use lower-case English characters. This is the plaintext: `venividivici`.
- Use the key to create the cipher table:

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

- For each character in the plaintext, substitute it with its cipher equivalent. This is now the ciphertext: `ajsnaninanhn`.

Note that we do not re-insert spaces, punctuation or capitals into the ciphertext: this can make the ciphertext easier to crack as it provides the cryptanalyst with more information (for example, if there is a word containing a single letter, then it is probably either “a” or “I”). When decrypting the message, we perform the reverse procedure: we take each character in the ciphertext and find its plaintext equivalent by **subtracting** the value of the key.

2.2 Mathematics

When considering this type of cipher from a mathematical perspective, we transform our plaintext from letters to numbers, by replacing ‘a’ with 0, ‘b’ with 1, and so on, as shown in the following table.

Letter	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o	p	q	r	s	t	u	v	w	x	y	z
Number	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25

For the key, we use the number of places n that each letter should be shifted to the right; this is equivalent to using the letter with which the cipher alphabet starts. We can then define our encryption function $E_n(x)$ that takes in the plaintext number x and returns the ciphertext number as follows:

$$E_n(x) = (x + n) \bmod 26 \tag{1}$$

Similarly, the decryption function (which takes in the ciphertext number y) is:

$$D_n(y) = (y - n) \bmod 26 \tag{2}$$

After encryption/decryption, we can convert the numbers back into letters. This mathematical process may seem pointless when working by hand, but it is much more easily implemented on a computer.

2.3 Questions:

3. The Blackboard site contains files “Caesar_plain_Q3.txt” and “Caesar_keys_Q3.txt”. These files contain (respectively) 100 pieces of plaintext and 100 keys for the Caesar cipher, numbered from 00 to 99. Use the plaintext/key with number T, where T is the value you calculated in Question 0.

(a) (1 mark) Write out a table showing how plaintext letters and ciphertext letters correspond. Use your table to encrypt the given plaintext.

(b) (1 mark) Convert the plaintext and key to numbers using the transformation described above, and encrypt the text. Verify that your answer matches the ciphertext in Part (a).

4. (2 marks) The Blackboard site contains files “Caesar_cipher_Q4.txt” and “Caesar_keys_Q4.txt”. These files contain (respectively) 100 pieces of ciphertext and 100 keys for the Caesar cipher, numbered from 00 to 99. Each piece of ciphertext has been encrypted using the corresponding key. Use the ciphertext/key with number T, where T is the value you calculated in Question 0. Decrypt the ciphertext. Your answer must include the plaintext, with appropriate spaces and punctuation inserted.

5. **(a)** (0 marks) Write a Python program that encrypts or decrypts text using the Caesar cipher. Your program must:

- Prompt the user to enter the following:
  - the name of a file that contains some text;
  - whether they wish to encrypt or decrypt the text; and
  - a key for the Caesar cipher, which will be an integer between 0 and 25 (inclusive).
- Print the encryption or decryption (as required) of the text in the file.
- Have variables with meaningful names, and be appropriately commented.

Hint(s):

- You may assume that all input values are valid (for example, the key is a valid number between 0 and 25 inclusive).
- The number of entries in an array `arr` is given by the command `size(arr)`.
- The SCIE1000 Blackboard site contains a file called “fileIO.py” which contains some useful commands. To use these commands, place a copy of that file in the same folder as the programs that you are writing, and import it into your program using the command

  ```
  from fileIO import *
  ```

  after the other import statements. For this question, the following commands which are defined in “fileIO.py” may be useful:

  `FileToIntArray`: This command reads the name of a file from the keyboard, then converts all of the text in that file to corresponding numbers, and places them in an array.

  `IntArrayToText`: This command takes an array of integers (assumed to be modulo 26) and converts it to an array of corresponding text.

- As an example, here is a sample program using those two commands:

```
from __future__ import division
from pylab import *
from fileIO import *

intArray = FileToIntArray()
print "The text, converted to numbers, is:"
print intArray
```

```
text = IntArrayToText(intArray)
print "The text in the file, printed as letters, is:"
print text
```

The following output is obtained when the program is run using a file called "vvv" that contains the text "Veni, Vidi, Vici".

```
Enter the file name: vvv
The text, converted to numbers, is:
[21 4 13  8 21  8  3  8  21  8  2  8]
The text in the file, printed as letters, is:
venividivici
```

- Note that you must **not** change the contents of fileIO.py under *any* circumstances. We suggest that you **do not** read the code in fileIO.py, as it uses some programming constructs that you have not learned.

(b) (4 marks) Print and submit a copy of your program. Marks will be awarded for:

- $*$ Adherence to the program specifications given above
- $*$ Appropriate programming style, structure and logic
- $*$ Appropriate print statements, with helpful text explanations of the output
- $*$ Use of comments and meaningful variable names

(c) (2 marks) Test your program in the following ways. You must include a printed copy of the output from your program in your submission; in each case, verify the output against your hand calculations. (Hint: in order to encrypt or decrypt your text, you will need to copy and paste the appropriate pieces of ciphertext and plaintext into files in the same folder as your Python program.)

(i) Use your program to repeat Question 3(b).

(ii) Use your program to repeat Question 4.

(d) (1 mark) The SCIE1000 Blackboard site contains a file called "fileIO.py", which you should have already copied into this folder. Use your program to encrypt that Python program file, using a Caesar cipher with a key of 10. Submit a copy of the encrypted text.

All questions above this line must be completed as part of your initial project submission.

---

3 The Affine Cipher

3.1 Science

The Caesar cipher is very insecure: there are only 26 possible keys, so testing every possible key is an easy way of cracking it. However, this cipher has been used until relatively recent times; for example, the Russian army used the Caesar cipher in 1915 because anything more secure was too complicated for most of the soldiers to understand! (Of course, the results were not good for the Russian army.)

The Caesar cipher is in fact a special case of another type of cipher, called an *Affine cipher*. With this cipher, the key consists of two numbers, say a and b. To be a valid key, the value a must have a multiplicative inverse modulo 26, and b can be any integer modulo 26. Then the Affine encryption function $A(x)$ (to encrypt a plaintext letter x) is defined by:

$$A(x) = ax + b \bmod 26 \tag{3}$$

where x is the plaintext number to be encrypted. Similarly, the decryption function (which operates on the ciphertext number y) is:

$$B(y) = a^{-1}(y - b) \bmod 26 \tag{4}$$

where a^{-1} is the multiplicative inverse of a, modulo 26.

Example

We can encrypt Julius Caesar's famous saying "Veni, Vidi, Vici" using an Affine cipher with a key of $(a, b) = (5, 17)$ as follows:

- Re-write the text to be enciphered to just use lower-case English characters. This is the plaintext: `venividivici`.
- Convert the plaintext to corresponding numbers:
  21 4 13 8 21 8 3 8 21 8 2 8
- For each number in the plaintext, multiply it by $a = 5$, then add $b = 17$, and finally take the answer modulo 26. For example, to encrypt the plaintext letter 'v', which corresponds to 21, the calculation is:

  $$(5 \times 21 + 17) \bmod 26 = 122 \bmod 26 \equiv 18.$$

  Thus the encryption of 21 is 18, so the encryption of the letter 'v' is the letter 's'. Following through for each letter in turn gives the following ciphertext: `slefsfgfsfbf`.

Once again, we do not re-insert spaces, punctuation or capitals into the ciphertext.

3.2 Questions:

6. **(a)** (1 mark) Explain why it is correct to describe the Caesar cipher as a "special case of the Affine cipher".

(b) (2 marks) What is the total number of valid keys for the Affine cipher? Show all working.

(c) (2 marks) Show that the decryption function B for the Affine cipher correctly decrypts text that has been encrypted with the encryption function A.

(d) (3 marks) The Blackboard site contains files "Affine_plain_Q6.txt" and "Affine_keys_Q6.txt". These files contain (respectively) 100 pieces of plaintext and 100 keys for the Affine cipher, numbered from 00 to 99. Use the plaintext/key with number T, where T is the value you calculated in Question 0. Encrypt the **first 12 letters** of the plaintext. Write your answer using letters, not numbers.

7. The Blackboard site contains files "Affine_cipher_Q7.txt" and "Affine_keys_Q7.txt". These files contain (respectively) 100 pieces of ciphertext and 100 keys for the Affine cipher, numbered from 00 to 99. Each piece of ciphertext has been encrypted using the corresponding key. Use the ciphertext/key with number T, where T is the value you calculated in Question 0.

(a) (1 mark) If the encryption key is (a, b), find a^{-1}. (The values of a and b are given in the key file.)

(b) (3 marks) Decrypt the **first 10 letters** of the ciphertext. Your answer must include the plaintext, with appropriate spaces and punctuation inserted.

4 General Mixed Alphabet cipher

4.1 Science

The Affine cipher is also very insecure: once again, checking every possible key is an easy way of cracking it. However, the Affine cipher is itself a special case of another type of cipher, called a *General Mixed Alphabet*

(GMA), which has many more possible keys. In fact, there are so many keys that it is not possible to check them all, even on very fast computers. In a GMA, rather than encrypting the plaintext using a mathematical equation, encryption proceeds as follows:

- First a secret key is chosen. The key is a sequence (or table) of 26 letters, showing the ciphertext letter to which each plaintext letter is encrypted. Each letter of the alphabet must occur exactly once as a plaintext and once as a ciphertext in the key.
- The plaintext is encrypted one letter at a time. Each occurrence of the letter 'a' in the plaintext is encrypted to the first letter in the key, each occurrence of 'b' to the second letter in the key, and so on.

Example

Earlier we encrypted Julius Caesar's famous saying "Veni, Vidi, Vici" using a Caesar cipher. Now we can encrypt it again, using a GMA.

- Re-write the text to be enciphered to just use lower-case English characters. This is the plaintext: `venividivici`.
- Choose a key and use it to create the cipher table. The key is the second row of the following table.

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

- For each character in the plaintext, substitute it with its cipher equivalent. This is now the ciphertext: `tujitimitihi`.

Note that once again, we do not re-insert spaces, punctuation or capitals into the ciphertext. When decrypting the message, we perform the reverse procedure: we take each character in the ciphertext and find its plaintext equivalent, using the cipher table. For example, if the ciphertext "stuhsuksf" had been encrypted using the above key, then the corresponding plaintext is "avecaesar", or "Ave, Caesar" with spaces and punctuation inserted.

4.2 Mathematics

Because each letter of the alphabet occurs exactly once in the key for a GMA, this key is an *ordered arrangement* of the alphabet. Mathematically, this is called a *permutation* of the alphabet. Given n items, there are $n!$ (pronounced "n-factorial") distinct permutations of those items.

4.3 Questions:

8. **(a)** (2 marks) What is the (approximate) total number of keys available for a GMA with 26 letters in the alphabet? Show all working.

(b) (2 marks) In the example above, the plaintext character 'i' encrypted to 'i' in the ciphertext. Consider the following variation to the standard GMA: "When choosing a key, ensure that no letter encrypts to itself". Would you expect this variation to make a GMA more secure, less secure, or to have no impact on the security? Explain your answer carefully.

(c) (2 marks) The key for a GMA is a permutation of the 26 letters, which is hard to remember. In practice, GMAs typically employ a secret keyword, as follows.

$*$ Write the secret word across the page, but not repeating any letter that has already been written.

$*$ Working from the start of the alphabet, under the keyword, write each letter that has not yet been written on the page. If you reach the end of the keyword, start writing on a new line.

∗ You should now have all 26 letters written exactly once each. Read these letters column by column; this forms the key for the GMA.

Find the GMA key that corresponds to your family name. Show all working.

(d) (1 mark) Using the key from Part (c), encrypt the phrase “Veni, Vidi, Vici”.

9. (2 marks) The Blackboard site contains files “GMA_plain_Q9.txt” and “GMA_keys_Q9.txt”. These files contain (respectively) 100 pieces of plaintext and 100 keys for the GMA, numbered from 00 to 99. Use the plaintext/key with number T, where T is the value you calculated in Question 0. Encrypt the plaintext.

10. (2 marks) The Blackboard site contains files “GMA_cipher_Q10.txt” and “GMA_keys_Q10.txt”. These files contain (respectively) 100 pieces of ciphertext and 100 keys for the GMA, numbered from 00 to 99. Each piece of ciphertext has been encrypted using the corresponding key. Use the ciphertext/key with number T, where T is the value you calculated in Question 0. Decrypt the ciphertext. Your answer must include the plaintext, with appropriate spaces and punctuation inserted.

5 Frequency Analysis

5.1 Science

Despite the enormous number of keys in a GMA, it has also been proven to be insecure: as early as the 9th century, the Arab scholar Al-Kindi was using *frequency analysis* to crack this type of cipher. The basis of frequency analysis is that — no matter which language is being used — not all letters occur equally often. In English text, for example, the most common letter is ‘e’ (which makes up approximately 12.702% of all letters) and the least common letter is ‘z’ (with a frequency of 0.074%). Compare this to a frequency of $1/26 \approx 3.85\%$, which is what we would expect if each letter had the same frequency. See Figure 1 for the relative frequencies of characters in English.

Letter	Frequency	Letter	Frequency
e	12.702%	**m**	2.406%
t	9.056%	**w**	2.360%
a	8.167%	**f**	2.228%
o	7.507%	**g**	2.015%
i	6.966%	**y**	1.974%
n	6.749%	**p**	1.929%
s	6.327%	**b**	1.492%
h	6.094%	**v**	0.978%
r	5.987%	**k**	0.772%
d	4.253%	**j**	0.153%
l	4.025%	**x**	0.150%
c	2.782%	**q**	0.095%
u	2.758%	**z**	0.074%

Figure 1: Relative frequencies of English characters, in descending order of frequency.

Thus, the basis of frequency analysis is that if we find the frequency of each letter in the ciphertext, the most common letter probably corresponds to the letter ‘e’, the second most common letter corresponds to ‘t’, and so on. Note that it is usually not this simple, as the frequencies in Table 1 are *averages* for text written in English; specific texts will probably have different frequencies. For example, the French author Georges Perec wrote the 200-page novel *La Disparation* in 1969 without using a single letter ‘e’ (which is a common letter in French as well); Gilbert Adair then translated this into English (entitled *A Void*), again avoiding usage of ‘e’.

As such, you might need to experiment with the top few most common letters in different orders. Frequency analysis typically works better on longer pieces of ciphertext, where the small variations do not affect the overall frequencies as much (for example, the most common letter in "Veni, Vidi, Vici" is 'i', followed by 'v').

5.2 Questions:

11. The Blackboard site contains a file "GMA_cipher_big.txt". The file contains 100 pieces of ciphertext, each encrypted using a GMA with a secret key. Use the ciphertext with number T, where T is the value you calculated in Question 0.

- **(a)** (3 marks) Find the absolute and relative frequencies of each letter within the ciphertext. (Note: absolute frequency is the count, and relative frequency is the count divided by the total length of the ciphertext).
- **(b)** (1 mark) Deduce which ciphertext letter is most likely to correspond to the plaintext letter 'e' (you may need to identify more than one letter in your answer).

12. **(a)** (0 marks) Modify your Python program from Question 5 so that in addition to implementing encryption and decryption using a Caesar cipher, it now also:

- Gives the user two initial options: encryption/decryption, or doing a frequency analysis.
- If the user chooses encryption/decryption, then:
  - · they can choose to use the Caesar cipher, Affine cipher or a GMA
  - · for a Caesar cipher or Affine cipher, the user is prompted to enter the key from the keyboard
  - · for a GMA, the program reads the 26-letter key from a file.
- If the user chooses to do a frequency analysis, then the program:
  - · plots a histogram of the absolute frequencies of the letters in the text
  - · prints a list of all English letters and their **relative frequencies** in the text, sorted from most frequent to least frequent.
- has variables with meaningful names, and is appropriately commented.

(Hint: don't forget the two Python commands described above (for reading text from a file, and for converting numbers back to text). The command `FileToIntArray` is not only useful for reading the plaintext/ciphertext, but you can also use it to read the key for a GMA from a file. In this case, the file should contain **only** the 26 letters of the key.)

(b) (14 marks) Print and submit a copy of your program. Marks will be awarded for:

- ∗ Adherence to the program specifications given above
- ∗ Appropriate programming style, structure and logic
- ∗ Appropriate print statements, with helpful text explanations of the output
- ∗ Use of comments and meaningful variable names

(c) Test your program in the following ways. (You must include a printed copy of the output from your program in your submission; in each case, verify the output against your hand calculations.)

- **(i)** (1 mark) Use your program to repeat Question 6(d).
- **(ii)** (1 mark) Use your program to repeat Question 7.
- **(iii)** (1 mark) Use your program to repeat Question 9.
- **(iv)** (1 mark) Use your program to repeat Question 10.
- **(v)** (1 mark) Use your program to repeat Question 11.

(d) (5 marks) Write a brief user guide which explains how to use the program. (Your user guide should **not** assume that the user has read this assignment question sheet.) The guide should contain all necessary information about:

- What the program does.
- What input is requested by the program, and what valid input it can take.
- What output the program gives.
- Any assumptions you have made, or any special cases.

This is a user guide, not a programmer's manual. Do not describe the algorithm you have used or internal details of the program. Instead, if someone with a basic understanding of computers (and a good understanding of the science relevant to your project topic) wanted to run your program, what would they need to know?

13. **(a)** (2 marks) Figure 1 presents "standard" relative frequencies of letters in English text. Find a large piece of electronic text (for example, on the web), save it to a file, and use your program to perform a frequency analysis on it. Submit a reference to the text (for example, the web address), the frequency histogram and the sorted table of relative frequencies.

(b) (2 marks) Compare the results you obtained in Part (a) with the data in Figure 1. In about 150 words, discuss similarities and differences, and explain why your results are similar to, or different from, the data in Figure 1.

14. (25 marks) When answering this question, if relevant you may include some graphs, diagrams, tables of values, equations or mathematical calculations (which will not be included in the word limit), but your response should be predominantly text-based. The report must be typed, written in a professional style, and be within 10% of 2000 words in length. Marks will be deducted for any report with length outside this range.

It is **not** appropriate to use other sources or references in your report; all of the discussion and recommendations should arise solely from your own work, including output from your program. The course Blackboard site contains a Criteria Sheet for this report, showing how marks will be allocated. **(The answer to this question must be submitted in hardcopy and also via Turnitin; see the project overview document.)**

A secretive political lobby group (SPLG) employs you to investigate methods by which they can keep their communications secret. They have heard that the GMA has many possible keys, is simple to understand, and is easy to use by hand, so they would like to use it to encrypt their communications. They understand that the cipher can be broken, but they want some evidence about how easy or difficult that is. They pose the following questions:

(a) How hard is it to crack some ciphertext encrypted using a GMA and an unknown key?

(b) Is there anything that can be "done" to a plaintext message that makes the corresponding ciphertext harder to crack?

Write a report for SPLG. The report must meet the following criteria:

– It must be understandable by SPLG's management, who have some knowledge of simple mathematics, but are not experts.

– It should **not** describe any of the mathematical details of the cipher, or anything about the programming you did; instead, SPLG wants you to answer their specific questions.

– It must include four sections:

1. An "Executive Summary" of around 150 words, giving a brief overview of your findings, and listing the answers to their questions. The rest of the document will expand on the content of the executive summary.

2. A detailed description of **your actual** experiences when attempting to crack ciphertext encrypted using a GMA and an unknown key.
   The Blackboard site contains a file "GMA_cipher_big.txt". The file contains 100 pieces of ciphertext, each encrypted using a GMA with a secret key. Use the ciphertext with number T, where T is the value you calculated in Question 0. Decrypt the ciphertext. Your answer should include:
   - frequency information about letters in the text, from your program
   - a detailed description of what you did, each step you followed, which letters you decrypted in which order, which choices worked and which didn't work (this discussion will probably be hundreds of words long). A large number of marks will be assigned to this discussion.
   - the key.
   - the plaintext, with spaces and punctuation inserted.

   If you cannot crack the cipher, it is still possible to receive full marks if your discussion is very good, and you explain why your approach didn't work.
3. Assume that SPLG needs to transmit a secret message about a special meeting they are holding. The message needs to be about 200 **characters** long, and it **has** to be encrypted with the GMA cipher.
   (a) Discuss some techniques which could be used to make it harder for the "enemy" to crack the message. (Note: all that you can control is the choice of key, and the content of the message.)
   (b) Use the principles you outlined in Part (a) to write a plaintext message (remember, it is about a political meeting, and should be about 200 characters long) and a key, and explain why the message would be 'hard' to crack.
4. A "Conclusions/answers" section containing brief answers/responses to their questions, based on the data and experiences you presented above.

The end