維吉尼亞密碼

計算機程式概論 作業二 書面報告

余京儒　心理四 **b04310036**

目錄

一、動機

這次的作業是延續之前的作業一，因為想說上次做的是密碼，那這次也就繼續來做密碼吧。之前作業一做的是凱薩密碼，在查了一下各種古典密碼後，決定做為凱薩密碼延伸的維吉尼亞密碼，並以物件導向撰寫且讓城市比起上次可以有更高的容錯。

二、構想解說與程式測試及規劃

維吉尼亞密碼是由一些偏移量不同的凱薩密碼所組成，一般的加密方法如下:

假設明文為 ATTACKATDAWN，金鑰為 lemon，先重複金鑰至與明文一樣長度，再根據維吉尼亞方格，將明文的第 k 個字，對應以金鑰的第 k 個字作為開頭的維吉尼亞方格中的行做轉換，如下:

明文: ATTACKATDAWN
金鑰: LEMONLEMONLE
密碼: LXFOPVEFRNHR

	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
A	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
B	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	A
C	C	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
D	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
E	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	D
F	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
G	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	F
H	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G
I	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H
J	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I
K	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J
L	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K
M	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L
N	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M
O	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N
P	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
Q	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P
R	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q
S	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R
T	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S
U	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T
V	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U
W	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V
X	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W
Y	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X
Z	Z	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

圖片來源: 維基百科

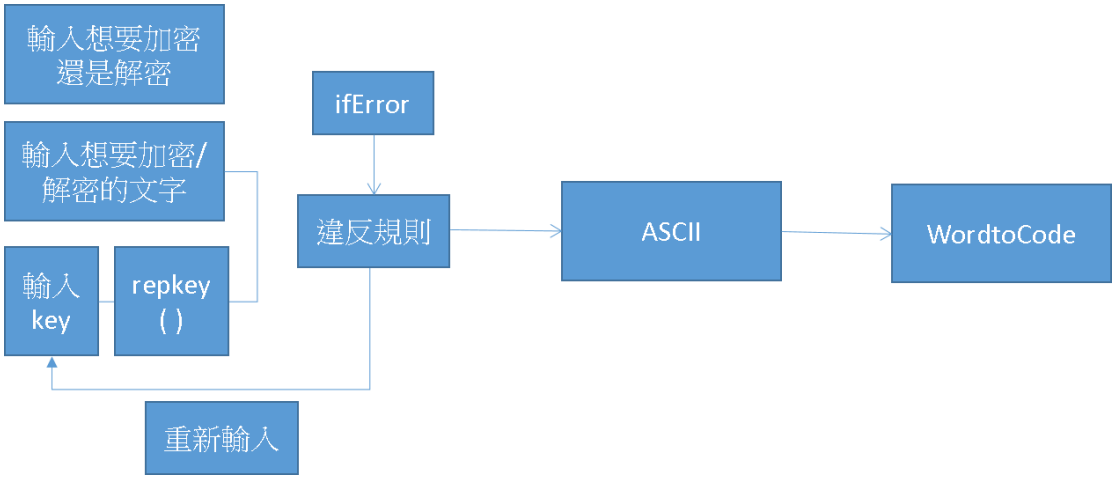
在我的程式中，並不需要真的做出維吉尼亞方格，只需將明文根據對應的金鑰的字，來增加它的位移就好。另外，我希望在程式中能有加密與解密兩種功能，然而維吉尼亞密碼的解密如果沒有金鑰的話有些困難，因此在此程式中的解密是在已知金鑰的情況下。我規劃的物件有下列四種:

1. IfError : 用來確認輸入的明文與金鑰是否都只包含英文字母，若含有其他字母則重新輸入。然而在實際寫時，因為物件的傳回一直出現問題，因此 iferror 只用來判斷輸入的明文與金鑰是否都只包含英文字母，要求重新輸入的部分還是寫在主程式。
2. Repkey: 用來將金鑰的重複到跟明文一樣長度。然而在實際寫實，一樣因為物件的傳回一直出現問題，因此最後寫成放在主程式的涵式。
3. ASCII: 用來將明文與金鑰轉換成 ASCII。因為金鑰其實為對應順序明文的位移量，因此分成兩個功能來寫。KeyToASCII( ) 是將金鑰轉為位移量後在做大小寫的處理，讓步論大寫還小寫都是一樣的位移量。而 ToASCII 則是負責將明文轉成對應的 ASCII。
4. WordtoCode: 用來將明文與其對應金鑰作位移後，將 ASCII 轉回字元，分為加密與解密兩種功能。Encrypt ()為加密，將明文加上金鑰後取除以 26 的餘數，

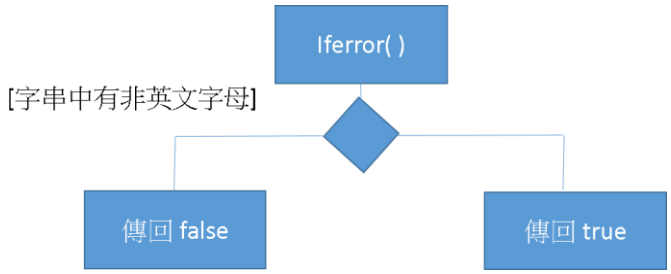
再轉回字元。Decrypt ()為解密，將明文扣除金鑰後加 26，取除以 26 的餘數，再轉回字元

三、程式流程圖(20%)

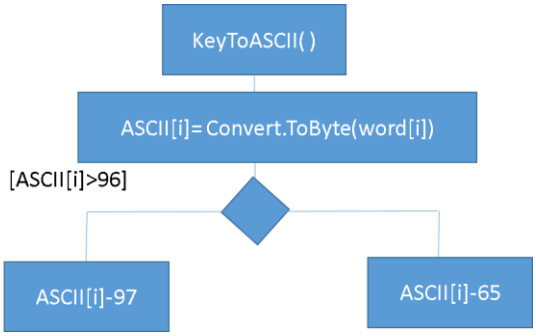
i. main

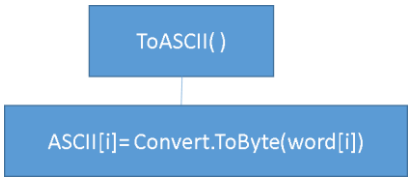


ii. ifError

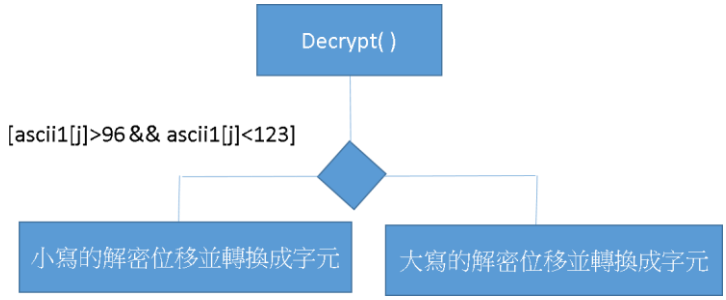


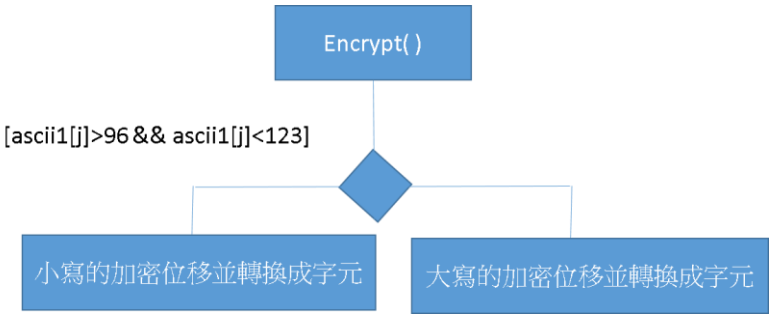
iii. ASCII





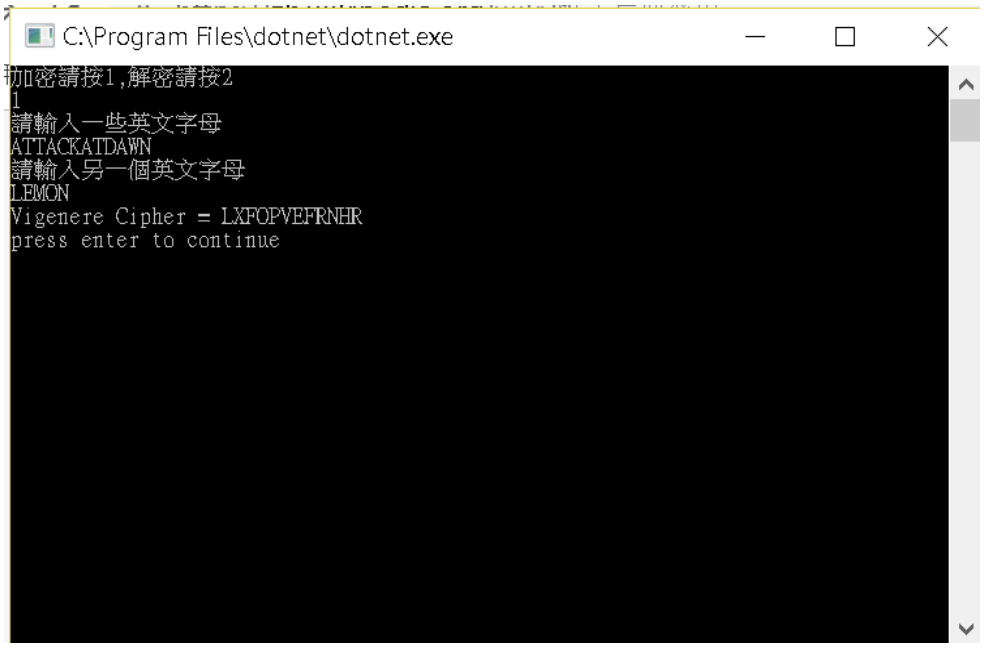
iiii. WordtoCode



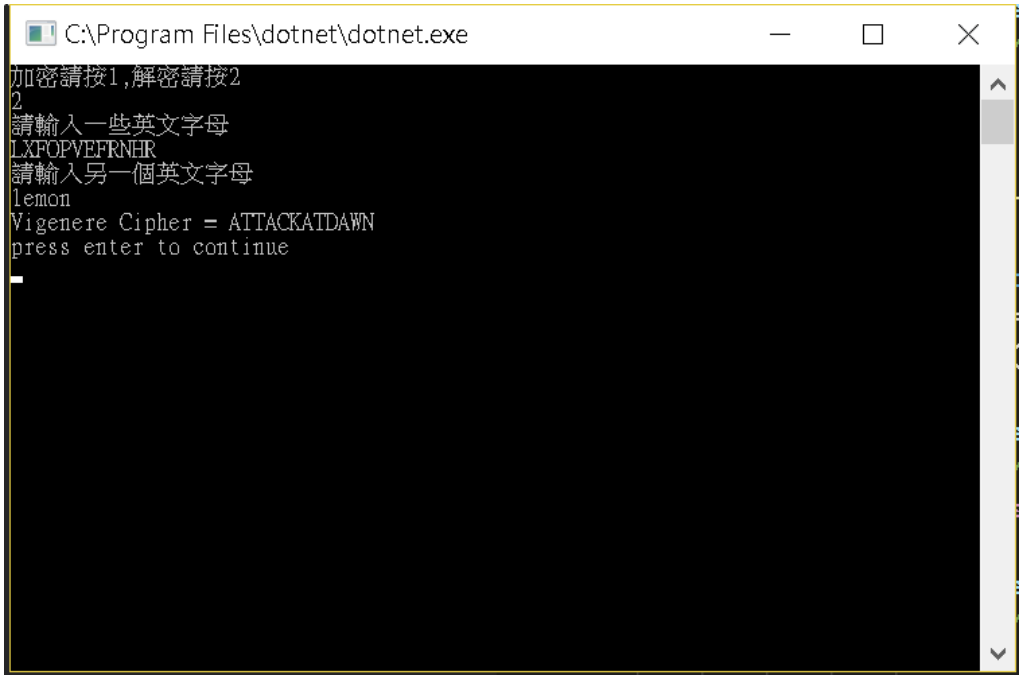


四、程式測驗執行結果

1. 加密



2. 解密



五、參考資料

1. C#上課講義
2. 維基百科—維吉尼亞密碼: https://zh.wikipedia.org/wiki/%E7%BB%B4%E5%90%89%E5%B0%BC%E4%BA%9A%E5%AF%86%E7%A0%81
3. 維基百科—ASCII: https://zh.wikipedia.org/wiki/ASCII
4. Stack Overflow

六、程式列表

1. program.cs

```
using System;


namespace HW2
{
    class Program
    {
        static void Main(string[] args)
        {
            Console.WriteLine("加密請按 1,解密請按 2");  //輸入加密或解密
            string c = Console.ReadLine() ;
```

```
ASCII text1 = new ASCII("1"); //設定初直
ASCII text2 = new ASCII("1");
bool e = false;
bool r = false;
string b = "1";
string a ="1";

while(e==false) //檢查輸入的字串中是否含有非英文字的符號，若有則重新輸入
{
Console.WriteLine("請輸入一些英文字母");
a = Console.ReadLine() ;
text1 = new ASCII(a);
text1.ToASCII();
e = IfError.iferror(text1.Ascii);
}

while(r==false)   //檢查 Key 中是否含有非英文字的符號，若有則重新輸入
{
Console.WriteLine("請輸入另一個英文字母");
b = Console.ReadLine();
text2 = new ASCII(repkey(a,b));  //呼叫涵式 repkey 將 key 重複至與明文一樣長
text2.ToASCII();
r = IfError.iferror(text2.Ascii);
}

text2.KeyToASCII();

WordtoCode post = new WordtoCode(text1.Ascii,text2.Ascii1);
if (c == "1")  //加密
{
    post.Encrypt();
}
else if (c=="2")
{
    post.Decrypt();  //解密
}
```

```
            Console.Write("Vigenère Cipher = ");
            for(int i = 0; i< a.Length; i++)
                {
                    Console.Write(post.Code[i]);
                }
            Console.WriteLine('\n'+"press enter to continue");
            Console.ReadLine();
        }
        static string repkey(string word,string key)
        {
            System.Text.StringBuilder key1 = new System.Text.StringBuilder(key);
            for (int i=0; i<word.Length-1; i++)
                {
                key1.Append(key);
                }
            return(key1.ToString());
        }
    }
}
```

2. IfError.cs

```
using System;

namespace HW2
{
    public class IfError
    {
        public static bool iferror(byte[] a)
        {
            for (int j =0; j< a.Length; j++)
            {
                if ((a[j]>64 && a[j]<91) || (a[j]>96 && a[j]<123))
                {
                   continue; //處理非英文字母
                }
                else
```

```
                {
                    return false;
                }
            }
            return true;
        }

    }
}
```

3. ASCII.cs

```
using System;

namespace HW2
{
    public class ASCII
    {
        private string word;
        private byte[] ascii;
        private byte[] ascii1;
        public ASCII (string w) //建構式
        {
            word = w;
        }
        public string Word
        {
            get { return word;} //屬性
        }

        public byte[] Ascii1
        {
            get { return ascii1;} //屬性
        }

        public byte[] Ascii
        {
            get { return ascii;}  //屬性
        }
```

```
        public void ToASCII() //將明文轉為 ASCII
        {
            ascii = new byte[word.Length];
            for (int i=0; i< word.Length; i++){
                //Console.WriteLine (chars[i]);
                char[] chars = word.ToCharArray();
                ascii[i] = Convert.ToByte(chars[i]);
                }
        }

        public void KeyToASCII()//將 KEY 轉為 ASCII
        {
            ascii1 = new byte[word.Length];
            for (int i=0; i< word.Length; i++){
                //Console.WriteLine (chars[i]);
                ascii1[i] = Convert.ToByte(word[i]);
                if (ascii1[i]>96){  //因為 KEY 跟大小寫沒有關係，所以做此處理
                    ascii1[i]-= 97;  //將不論大寫還是小寫都改為 a=1 開始
                    }
                else{
                    ascii1[i]-=65;
                }
            }
        }
    }
}
```

4. WordtoCode.cs

```
using System;

namespace HW2
{
    public class WordtoCode
    {
         private byte[] ascii1;
         private byte[] ascii2;
        public WordtoCode(byte[] a1,byte[] a2)
        {
```

```
            ascii1 = a1;
            ascii2 = a2;
        }
         public byte[] Ascii1
        {
            get { return ascii1;}
        }
        public byte[] Ascii2
        {
            get { return ascii2;}
        }
        private char[] code ;
        public char[] Code
        {
            get { return code;}
        }
        public void Encrypt ()
         {
            code = new char[ascii1.Length];
            for (int j =0; j< ascii1.Length; j++){
                if (ascii1[j]>96 && ascii1[j]<123){
                    code[j] = Convert.ToChar(((ascii1[j]-96+ascii2[j])%26)+96);
                    } //處理小寫英文字母
                else if (ascii1[j]>64 && ascii1[j]<91){
                    code[j] = Convert.ToChar(((ascii1[j]-64+ascii2[j])%26)+64);
                    } //處理大寫英文字母
                }
         }
        public void Decrypt ()
         {
            code = new char[ascii1.Length];
            for (int j =0; j< ascii1.Length; j++){
                    if (ascii1[j]>96 && ascii1[j]<123)
                    {
                    code[j] = Convert.ToChar(((ascii1[j]-96-ascii2[j]+26)%26)+96);
                    } //處理小寫英文字母
                     else if (ascii1[j]>64 && ascii1[j]<91)
                    {
```

```
                code[j] = Convert.ToChar(((ascii1[j]-64-ascii2[j]+26)%26)+64);
                } //處理大寫英文字母
            }
        }
    }
}
```