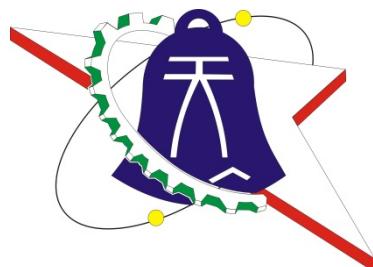


**國立彰化師範大學技術及職業教育學院  
工業教育與技術學系**



**程式設計  
期末專題**

**題目名稱**

檔案加密器

班級：生一

學號：S0724008

姓名：余紹華

**中華民國108年6月19日**

## 目錄

一.系統功能，輸入及輸出資料

二.應用層面說明

三.資料結構說明

四.程式剖析

五.程式執行範例

六.完整的程式

## 一. 系統功能，輸入及輸出資料

透過凱薩加密，對文字進行加密與解密。分為兩個部分，分別為加密與解密。

### 1. 加密

會輸入一個plaintext.txt的檔案，檔案的內容為你想要加密的文字，接著程式會要求你輸入密碼(長度在100個字以內)，進行加密。待加密完成後，會顯示「完成了」，並輸出一個檔名為ciphertext.txt的檔案，裡面的內容將會是一段加密後的文字。

### 2. 解密

如果要進行加密，則須輸入一個檔名為ciphertext.txt的檔案，內容為加密過後的文字，接著輸入與加密時相同的密碼，即進行解密解密的動作。解密完成以後，程式會顯示「完成了」，並輸出一個plaintext-recovered的檔案，檔案內容將會顯示解密完成的文字。

## 二. 應用層面說明

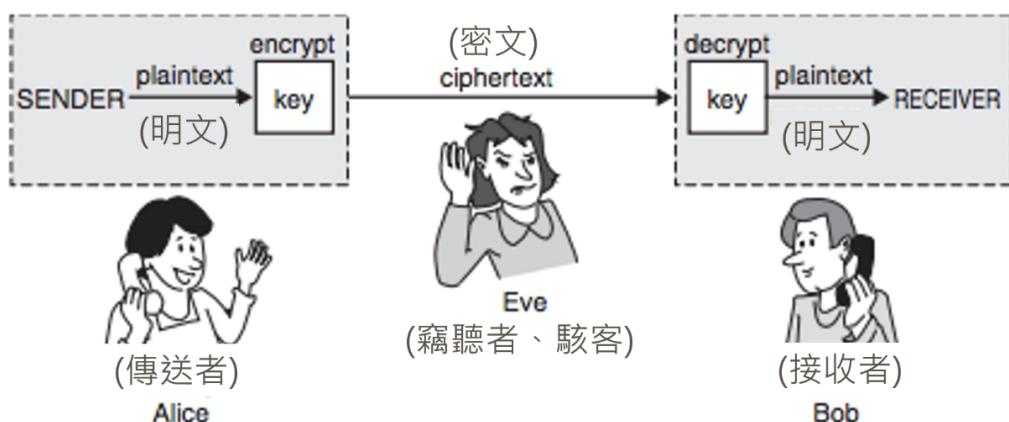


Fig.1 加密與解密

當Alice(傳送者)要傳送一則訊息給Bob(接收者)時，將欲傳送的訊息(明文)透過金鑰進行加密，就可以避免被Eve(竊聽者)竊取。收到訊息的Bob必須具有與傳送者相同的金鑰，才可以進行解密，獲取訊息。透過這個程式，以後在傳遞訊息時，就可以不用擔心被竊取資料了!!!!

### 三. 資料結構說明(流程圖)

#### 1. 加密

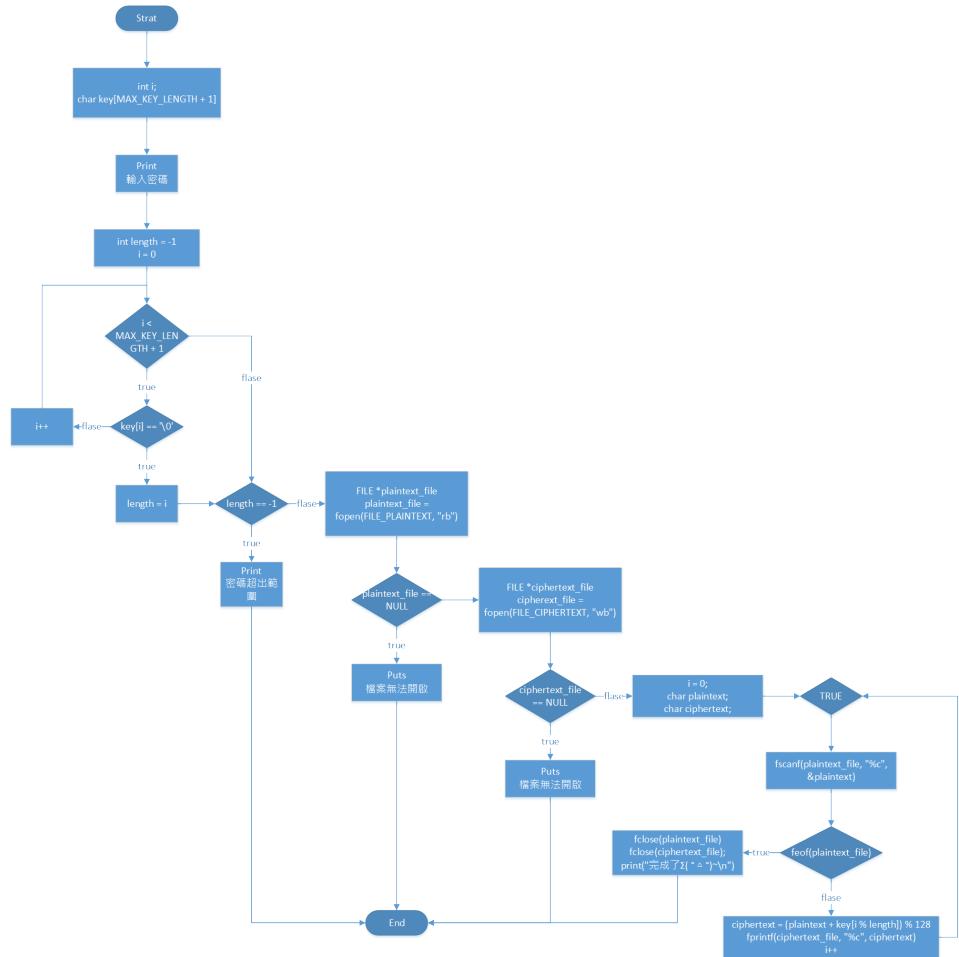


Fig.2 加密流程圖

## 2. 解密

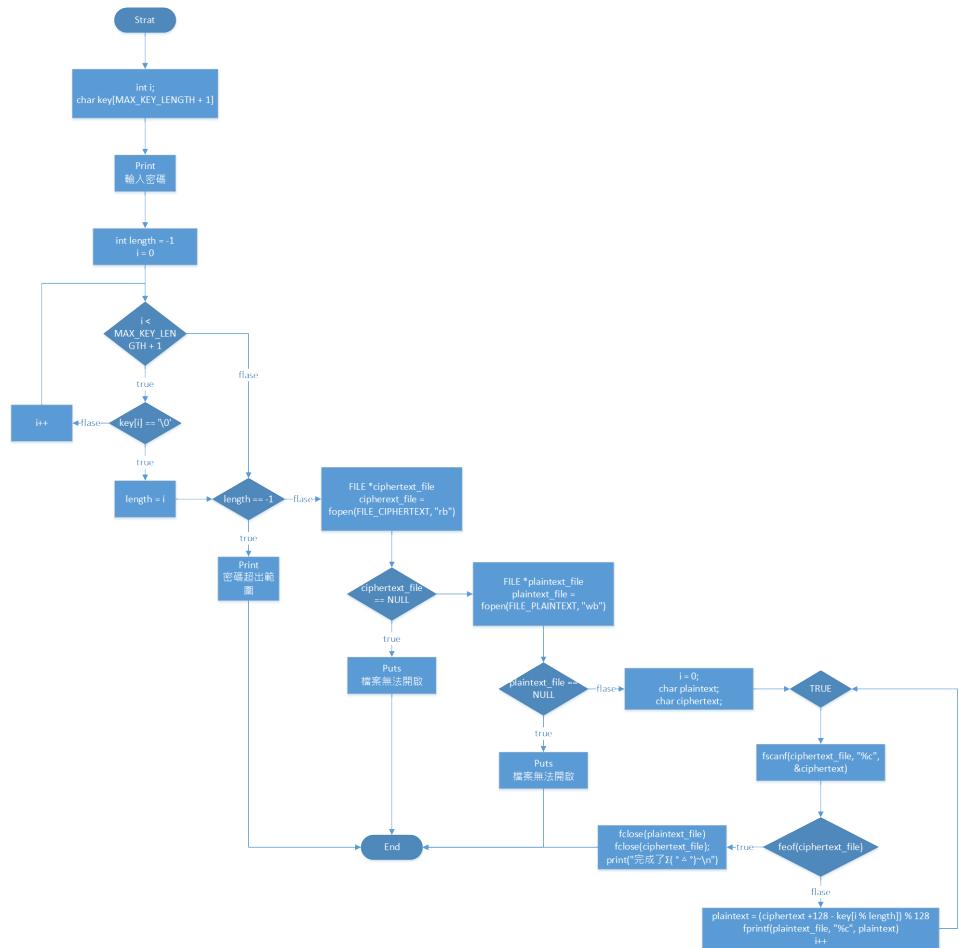


Fig.3流程圖-解密

## 四. 程式剖析

### 1. 加密

```
1 #define MAX_KEY_LENGTH 100
2 #define FILE_PLAINTEXT "plaintext.txt"
3 #define FILE_CIPHERTEXT "ciphertext.txt"
4
5 #include <stdio.h>
6
7 int main(void) {
8     int i;
9     char key[MAX_KEY_LENGTH + 1];
10    printf("輸入密碼(最多 %d 個字)：" , MAX_KEY_LENGTH);
11    scanf("%s", key);
12    int length = -1;
13    for (i = 0; i < MAX_KEY_LENGTH + 1; i++) {
14        if (key[i] == '\0') {
15            length = i;
16            break;
17        }
18    }
19    if (length == -1) {
20        printf("密碼超出範圍了QQ!!!");
21        return -1;
22    }
23 //    printf("password: %s\n", key);
24 }
```

Fig.4加密程式(1)

第1~3行，以第1行為例，只要在這個程式內看到Max\_Key\_Length都會自動代入100。

第7行，開始main函式。

第9行，宣告一陣列可以存放 $100 + 1$ 個字元，指定密碼長度為100個字(第1行)，電腦會自動在字串的最後加上/0，故需要多要一格記憶體。

第12~22行，將length的初始值設為-1，利用for迴圈來檢測密碼的長度，一旦看到\0(字串的結尾)，表示密碼在合理範圍內，就會更新length值，若到檢查完都沒有更新此變數，那就會維持-1，代表長度超過設定之最大值。未來在執行時，有可能會出錯，所以必須告訴使用者，並結束程式。

```
25     FILE *plaintext_file;
26     plaintext_file = fopen(FILE_PLAINTEXT, "rb");
27     if(plaintext_file == NULL) {
28         puts("無法開啟檔案!!!!");
29         return -1;
30     }
31
32     FILE *ciphertext_file;
33     ciphertext_file = fopen(FILE_CIPHERTEXT, "wb");
34     if(ciphertext_file == NULL) {
35         puts("無法編輯檔案!!!!");
36         return -1;
37     }
38 }
```

Fig.5加密程式(2)

第25~30行，會以二進位的方式讀取一個檔名plaintext.txt的檔案，接著會

判斷檔案是否可以讀取，若檔案無法讀取，則會告訴使用者並結束程式。

第32~37行，會以二進位的方式將內容寫入ciphertext.txt的檔案中，接著會判斷檔案是否能寫入，若檔案無法寫入，則會告訴使用者並結束程式。

```
39     i = 0;
40     char plaintext;
41     char ciphertext;
42     while (1) {
43         fscanf(plaintext_file, "%c", &plaintext);
44         if (feof(plaintext_file)) {
45             break;
46         }
47         ciphertext = (plaintext + key[i % length]) % 128;
48         //printf("%d %c(%x) %c(%x)\n", i, plaintext, plaintext & 0xff, ciphertext, ciphertext & 0xff);
49         fprintf(ciphertext_file, "%c", ciphertext);
50         i++;
51     }
52
53     fclose(plaintext_file);
54     fclose(ciphertext_file);
55     printf("完成了『( °_△ °)~』\n");
56
57     return 0;
58 }
```

Fig.6加密程式(3)

第42~51行，設了一個while(1)迴圈，一次讀一個字元，直到讀到檔案的最後，跳出迴圈，否則繼續。

第47行，(可參考Fig.6)電腦會自動將讀到的字元轉為一組數字儲存，加上密碼轉換的數字，除以128取餘數(能夠加密的範圍在ASCII code表上的00~127，共128個字)。理論上密碼的長度會小於被加密文字的長度，密碼會不斷重複，故除以密碼本身的長度取餘數。

第53~58行，待讀取與輸入完畢後，關閉檔案，結束程式。

明文	密碼	明文ASCII值	密碼ASCII值	加總	密文ASCII值	密文
B	t	66	116	182	54	6
a	e	97	101	198	70	F
d	s	100	115	215	87	W
	t	32	116	148	20	
R	t	82	116	198	70	F
o	e	111	101	212	84	T
m	s	109	115	224	96	`
a	t	97	116	213	85	U
n	t	110	116	226	98	b
c	e	99	101	200	72	H
e	s	101	115	216	88	X
	t	32	116	148	20	

Fig.7加密程式第47行轉換機制

# ASCII TABLE

Decimal	Hex	Char	Decimal	Hex	Char	Decimal	Hex	Char	Decimal	Hex	Char
0	0	[NULL]	32	20	[SPACE]	64	40	@	96	60	`
1	1	[START OF HEADING]	33	21	!	65	41	A	97	61	a
2	2	[START OF TEXT]	34	22	"	66	42	B	98	62	b
3	3	[END OF TEXT]	35	23	#	67	43	C	99	63	c
4	4	[END OF TRANSMISSION]	36	24	\$	68	44	D	100	64	d
5	5	[ENQUIRY]	37	25	%	69	45	E	101	65	e
6	6	[ACKNOWLEDGE]	38	26	&	70	46	F	102	66	f
7	7	[BELL]	39	27	'	71	47	G	103	67	g
8	8	[BACKSPACE]	40	28	(	72	48	H	104	68	h
9	9	[HORIZONTAL TAB]	41	29	)	73	49	I	105	69	i
10	A	[LINE FEED]	42	2A	*	74	4A	J	106	6A	j
11	B	[VERTICAL TAB]	43	2B	+	75	4B	K	107	6B	k
12	C	[FORM FEED]	44	2C	,	76	4C	L	108	6C	l
13	D	[CARRIAGE RETURN]	45	2D	-	77	4D	M	109	6D	m
14	E	[SHIFT OUT]	46	2E	.	78	4E	N	110	6E	n
15	F	[SHIFT IN]	47	2F	/	79	4F	O	111	6F	o
16	10	[DATA LINK ESCAPE]	48	30	0	80	50	P	112	70	p
17	11	[DEVICE CONTROL 1]	49	31	1	81	51	Q	113	71	q
18	12	[DEVICE CONTROL 2]	50	32	2	82	52	R	114	72	r
19	13	[DEVICE CONTROL 3]	51	33	3	83	53	S	115	73	s
20	14	[DEVICE CONTROL 4]	52	34	4	84	54	T	116	74	t
21	15	[NEGATIVE ACKNOWLEDGE]	53	35	5	85	55	U	117	75	u
22	16	[SYNCHRONOUS IDLE]	54	36	6	86	56	V	118	76	v
23	17	[ENG OF TRANS. BLOCK]	55	37	7	87	57	W	119	77	w
24	18	[CANCEL]	56	38	8	88	58	X	120	78	x
25	19	[END OF MEDIUM]	57	39	9	89	59	Y	121	79	y
26	1A	[SUBSTITUTE]	58	3A	:	90	5A	Z	122	7A	z
27	1B	[ESCAPE]	59	3B	:	91	5B	[	123	7B	{
28	1C	[FILE SEPARATOR]	60	3C	<	92	5C	\	124	7C	
29	1D	[GROUP SEPARATOR]	61	3D	=	93	5D	]	125	7D	}
30	1E	[RECORD SEPARATOR]	62	3E	>	94	5E	^	126	7E	~
31	1F	[UNIT SEPARATOR]	63	3F	?	95	5F	_	127	7F	[DEL]

Fig.8 ASCII code

## 2. 解密

```

1 #define MAX_KEY_LENGTH 100
2 #define FILE_PLAINTEXT "plaintext-recovered.txt"
3 #define FILE_CIPHERTEXT "ciphertext.txt"
4
5 #include <stdio.h>
6
7 int main(void) {
8     int i;
9     char key[MAX_KEY_LENGTH + 1];
10    printf("輸入密碼(最多 %d 個字)：" , MAX_KEY_LENGTH);
11    scanf("%s", key);
12    int length = -1;
13    for (i = 0; i < MAX_KEY_LENGTH + 1; i++) {
14        if (key[i] == '\0') {
15            length = i;
16            break;
17        }
18    }
19    if (length == -1) {
20        printf("碼超出範圍了QQ!!!");
21        return -1;
22    }
23 //    printf("password: %s\n", key);
24

```

Fig.9解密程式(1)

此部分與加密程式相同!!!(可參考加密程式(1))

檢測密碼的長度，若密碼超出範圍，告知使用者後結束程式。

```

25     FILE *ciphertext_file;
26     ciphertext_file = fopen(FILE_CIPHERTEXT, "rb");
27     if(ciphertext_file == NULL) {
28         puts("法開啟檔案!!!");
29         return -1;
30     }
31
32     FILE *plaintext_file;
33     plaintext_file = fopen(FILE_PLAINTEXT, "wb");
34     if(plaintext_file == NULL) {
35         puts("無法編輯檔案!!!");
36         return -1;
37     }
38

```

Fig.10解密程式(2)

第25~30行，會以二進位的方式讀取一個檔名ciphertext.txt的檔案，接著會判斷檔案是可以讀取，若檔案無法讀取，會告訴使用者並結束程式。

第32~37行，會以二進位的方式將內容寫入plaintext-recovered.txt的檔案中，接著會判斷檔案是否可以寫入，若檔案無法寫入，會告訴使用者並結束程式。

```

39     i = 0;
40     char plaintext;
41     char ciphertext;
42     while (1) {
43         fscanf(ciphertext_file, "%c", &ciphertext);
44         if (feof(ciphertext_file)) {
45             break;
46         }
47         plaintext = (ciphertext + 128 - key[i % length]) % 128;
48         //printf("%d %c(%x) %c\n", i, ciphertext, ciphertext & 0xff, plaintext);
49         fprintf(plaintext_file, "%c", plaintext);
50         i++;
51     }
52
53     fclose(ciphertext_file);
54     fclose(plaintext_file);
55     printf("完成了喔(\u2764\u2764)\n");
56
57     return 0;
58 }

```

Fig.11解密程式(3)

第47行，先加上128確保數值在減掉密碼的長度後仍為正數，最後除以128是為了避面有超過128的疑慮。

密文	密碼	密文ASCII值	密碼ASCII值	相減	公式	明文ASCII值	明文
6	t	54	116	-62	66	66	B
F	e	70	101	-31	97	97	a
W	s	87	115	-28	100	100	d
	t	20	116	-96	32	32	
F	t	70	116	-46	82	82	R
T	e	84	101	-17	111	111	o
'	s	96	115	-19	109	109	m
U	t	85	116	-31	97	97	a
b	t	98	116	-18	110	110	n
H	e	72	101	-29	99	99	c
X	s	88	115	-27	101	101	e
	t	20	116	-96	32	32	

Fig.12解密程式第47行轉換機制

## 五. 程式執行範例

```
輸入密碼（最多 100 個字）：This is a test!
完成了Σ( °△°)~  
-----  
Process exited after 17.85 seconds with return value 0  
請按任意鍵繼續 . . . ■
```

Fig.13 加密的執行結果

使用者需要輸入一段密碼，若密碼在指定長度內，加密完成後會告知使用者「完成了」。

Fig.14 密碼超過長度

若使用者輸入的密碼超過指定長度，則會顯示「密碼超出範圍」，結束程式。

```
輸入密碼（最多 100 個字）：This is a test!
無法開啟檔案!!!
-----
Process exited after 14.47 seconds with return value 4294967295
請按任意鍵繼續 . . .
```

Fig.15 檔案不存在

若檔案不存在，也會告知使用者，然後結束程式。

```
輸入密碼(最多 100 個字) : This is a test!
完成了喔(≥▽≤)~  
-----  
Process exited after 81.48 seconds with return value 0  
請按任意鍵繼續 . . . ■
```

Fig.16解密的執行結果

使用者需要輸入與加密時相同的密碼，解密完成後會告知使用者「完成了」。

 plaintext.txt - 記事本  
檔案(F) 編輯(E) 格式(O) 檢視(V) 說明(H)  
**Bad Romance by Lady Gaga**

Oh-oh-oh-oh-oh! Oh-oh-oh-oh-oh-oh!  
Caught in a bad romance  
Oh-oh-oh-oh-oh! Oh-oh-oh-oh-oh-oh!  
Caught in a bad romance

Rah rah ah-ah-ah!  
Ro mah ro-mah-mah  
Gaga ooh-la-la!  
Want your bad romance

/\* End \*/

Fig.17 輸入的檔案(Plaintext.txt)擷取頭、尾兩部分。

完整檔案的內容為Lady Gaga 的 Bad Romance。

Fig.18加密後輸出的檔案/解密時輸入的檔案(ciphertext.txt)。

plaintext-recovered.txt - 記事本

檔案(F) 編輯(E) 格式(O) 檢視(V) 說明(H)

Bad Romance by Lady Gaga

Oh-oh-oh-oh-oh! Oh-oh-oh-oh-oh-oh!  
Caught in a bad romance  
Oh-oh-oh-oh-oh! Oh-oh-oh-oh-oh-oh!  
Caught in a bad romance

Rah rah ah-ah-ah!  
Ro mah ro-mah-mah  
Gaga ooh-la-la!  
Want your bad romance

/\* End \*/

Fig.19解密後輸出的檔案(Plaintext-recovered.txt)擷取頭、尾兩部分。

可與Fig.17做比較，內容是相同的。

## 六. 完整的程式碼

### 1. 加密

```
1 #define MAX_KEY_LENGTH 100
2 #define FILE_PLAINTEXT "plaintext.txt"
3 #define FILE_CIPHERTEXT "ciphertext.txt"
4
5 #include <stdio.h>
6
7 int main(void) {
8     int i;
9     char key[MAX_KEY_LENGTH + 1];
10    printf("輸入密碼(最多 %d 個字) : ", MAX_KEY_LENGTH);
11    scanf("%s", key);
12    int length = -1;
13    for (i = 0; i < MAX_KEY_LENGTH + 1; i++) {
14        if (key[i] == '\0') {
15            length = i;
16            break;
17        }
18    }
19    if (length == -1) {
20        printf("密碼超出範圍了QQ!!!");
21        return -1;
22    }
23 //    printf("password: %s\n", key);
24
25 FILE *plaintext_file;
26 plaintext_file = fopen(FILE_PLAINTEXT, "rb");
27 if(plaintext_file == NULL) {
28     puts("無法開啟檔案!!!!");
29     return -1;
30 }
31
32 FILE *ciphertext_file;
33 ciphertext_file = fopen(FILE_CIPHERTEXT, "wb");
34 if(ciphertext_file == NULL) {
35     puts("無法編輯檔案!!!!");
36     return -1;
37 }
38
39 i = 0;
40 char plaintext;
41 char ciphertext;
42 while (1) {
43     fscanf(plaintext_file, "%c", &plaintext);
44     if (feof(plaintext_file)) {
45         break;
46     }
47     ciphertext = (plaintext + key[i % length]) % 128;
48     //printf("%d %c(%x) %c(%x)\n", i, plaintext, plaintext & 0xff, ciphertext,
49     ciphertext & 0xff);
50     fprintf(ciphertext_file, "%c", ciphertext);
51     i++;
52 }
53 fclose(plaintext_file);
54 fclose(ciphertext_file);
55 printf("完成了Σ( ̄△ ̄)~\n");
56
57 return 0;
58 }
```

## 2. 解密

```
1 #define MAX_KEY_LENGTH 100
2 #define FILE_PLAINTEXT "plaintext-recovered.txt"
3 #define FILE_CIPHERTEXT "ciphertext.txt"
4
5 #include <stdio.h>
6
7 int main(void) {
8     int i;
9     char key[MAX_KEY_LENGTH + 1];
10    printf("輸入密碼(最多 %d 個字)：" , MAX_KEY_LENGTH);
11    scanf("%s", key);
12    int length = -1;
13    for (i = 0; i < MAX_KEY_LENGTH + 1; i++) {
14        if (key[i] == '\0') {
15            length = i;
16            break;
17        }
18    }
19    if (length == -1) {
20        printf("碼超出範圍了QQ!!!");
21        return -1;
22    }
23    // printf("password: %s\n", key);
24
25    FILE *ciphertext_file;
26    ciphertext_file = fopen(FILE_CIPHERTEXT, "rb");
27    if(ciphertext_file == NULL) {
28        puts("法開啟檔案!!!!");
29        return -1;
30    }
31
32    FILE *plaintext_file;
33    plaintext_file = fopen(FILE_PLAINTEXT, "wb");
34    if(plaintext_file == NULL) {
35        puts("無法編輯檔案!!!!");
36        return -1;
37    }
38
39    i = 0;
40    char plaintext;
41    char ciphertext;
42    while (1) {
43        fscanf(ciphertext_file, "%c", &ciphertext);
44        if (feof(ciphertext_file)) {
45            break;
46        }
47        plaintext = (ciphertext + 128 - key[i % length]) % 128;
48        //printf("%d %c(%x) %c\n", i, ciphertext, ciphertext & 0xff, plaintext);
49        fprintf(plaintext_file, "%c", plaintext);
50        i++;
51    }
52
53    fclose(ciphertext_file);
54    fclose(plaintext_file);
55    printf("完成了喔(ง•̀ㅂ•́)ง~\n");
56
57    return 0;
58 }
```