JBU CTF

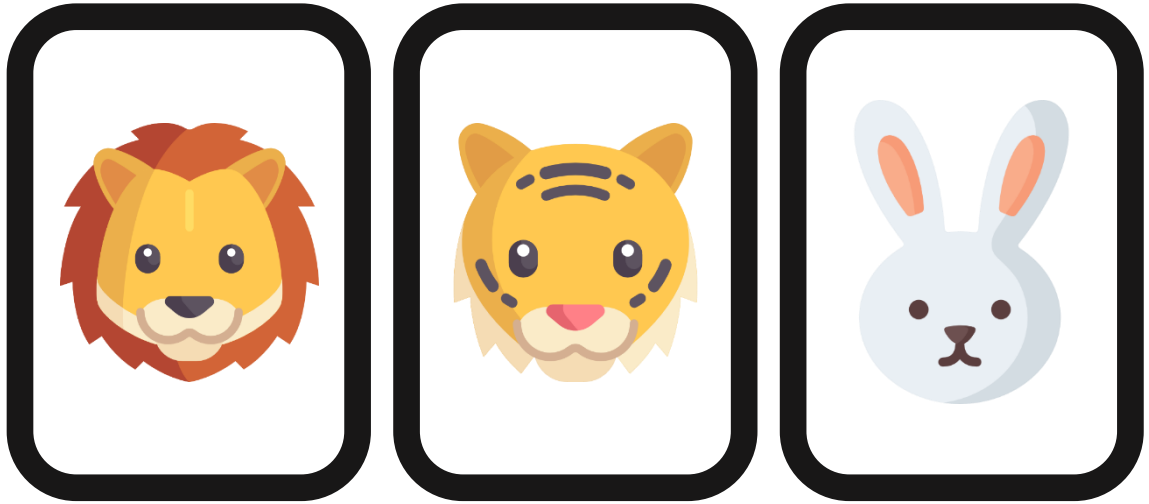
R E V E R S I N G

이다영

1. 문제

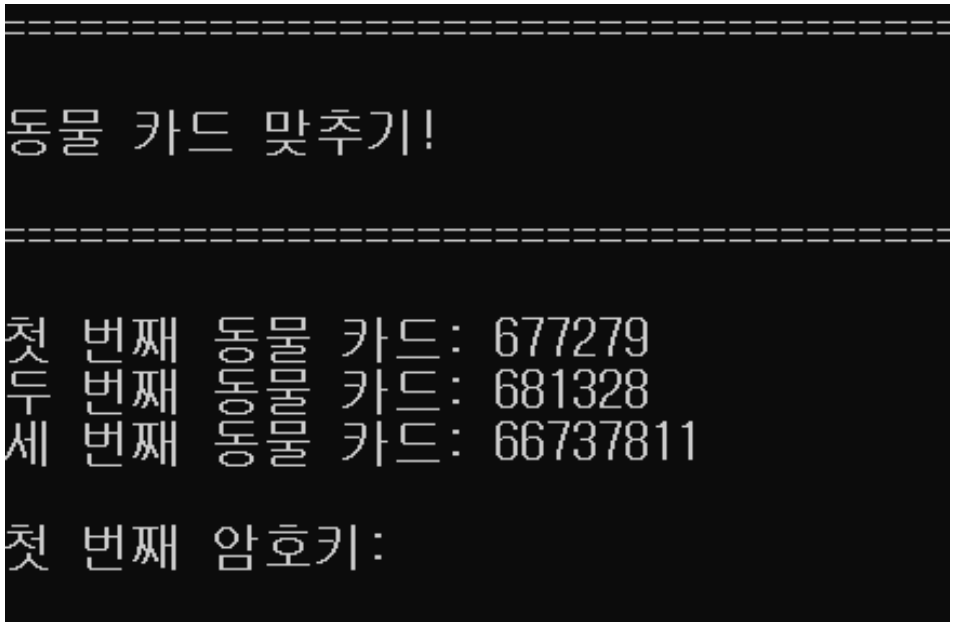
동물 카드 맞추기

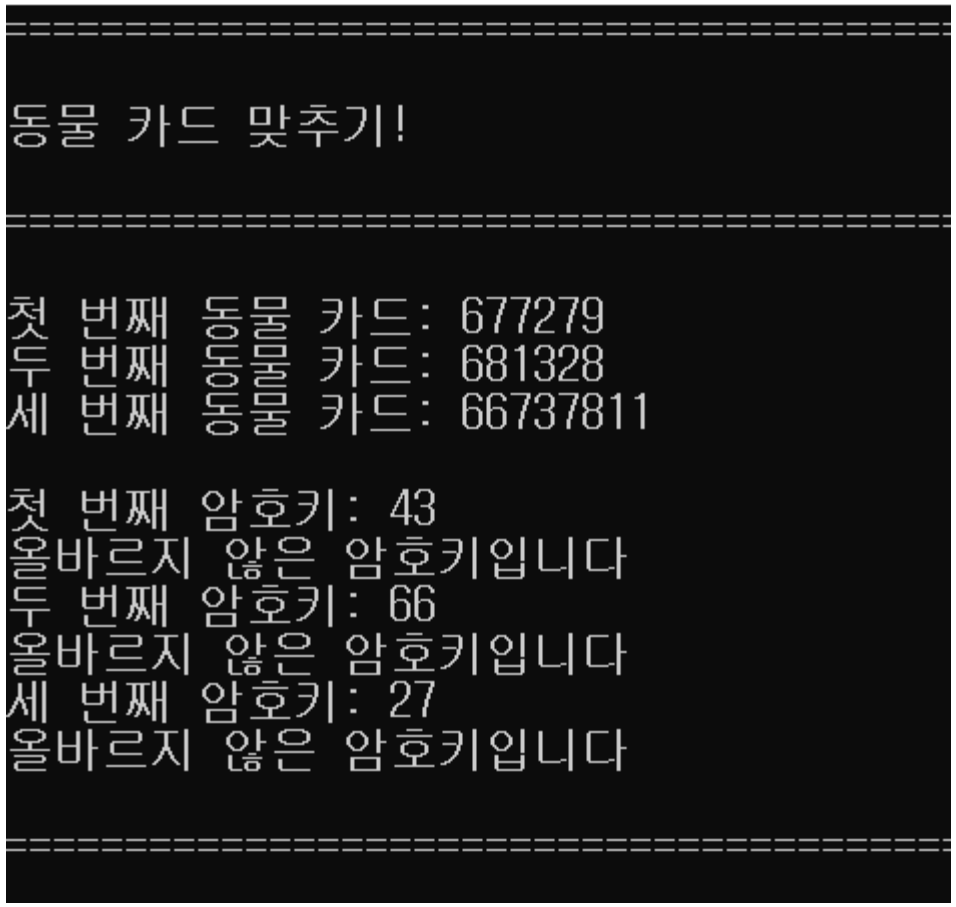
세 장의 동물 카드가 있습니다.
그런데 세 장 모두 암호화가 되었습니다.
복호화를 위해서는 암호키가 필요합니다.
암호키를 통해 복호화하고, 동물 카드를 맞춰봅시다!
Ex) scpCTF{LION_TIGER_RABBIT}



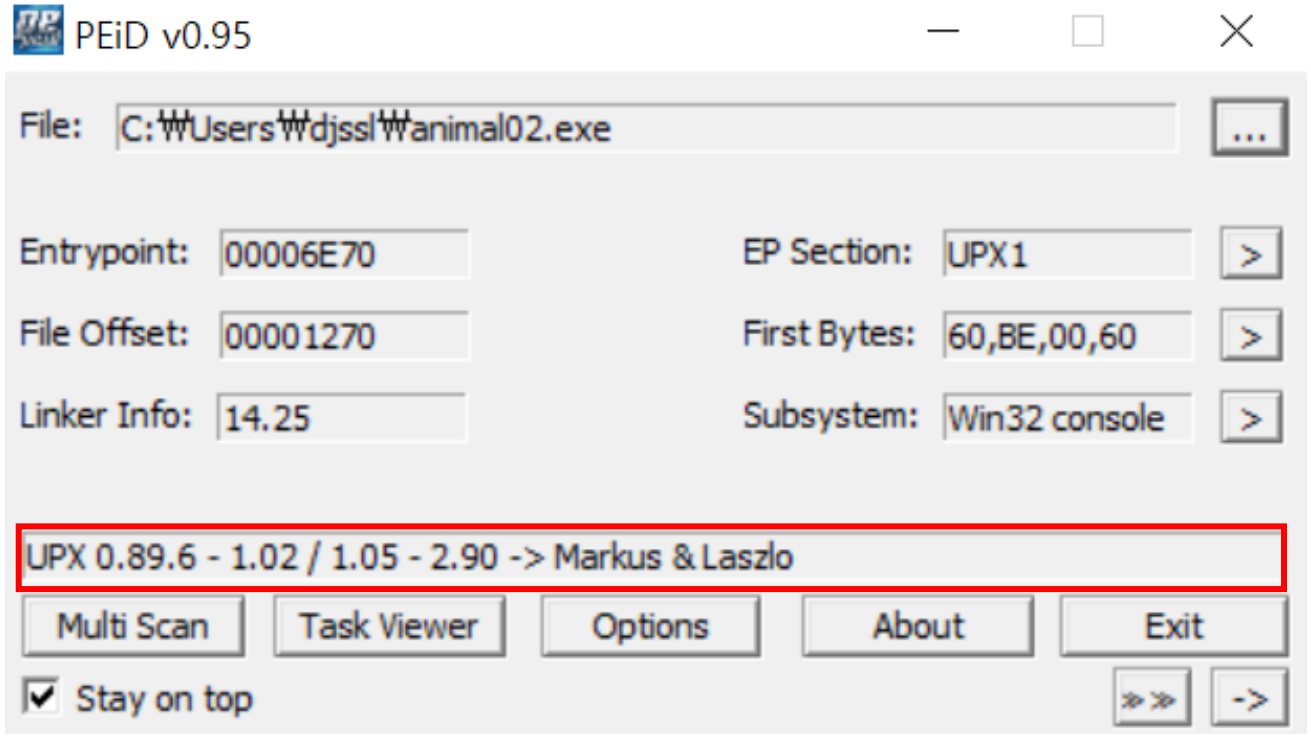
2. 문제 풀이

프로그램 실행


동물 카드 맞추기!
첫 번째 동물 카드: 677279
두 번째 동물 카드: 681328
세 번째 동물 카드: 66737811
첫 번째 암호키:


동물 카드 맞추기!
첫 번째 동물 카드: 677279
두 번째 동물 카드: 681328
세 번째 동물 카드: 66737811
첫 번째 암호키: 43
올바르지 않은 암호키입니다
두 번째 암호키: 66
올바르지 않은 암호키입니다
세 번째 암호키: 27
올바르지 않은 암호키입니다

패킹 여부 확인


PEiD v0.95
File: C:₩Users₩djssl₩animal02.exe
...
Entrypoint: 00006E70
EP Section: UPX1
>
File Offset: 00001270
First Bytes: 60,BE,00,60
>
Linker Info: 14.25
Subsystem: Win32 console
>
UPX 0.89.6 - 1.02 / 1.05 - 2.90 -> Markus & Laszlo
Multi Scan
Task Viewer
Options
About
Exit
Stay on top
>>
->

패킹 여부 확인

```
00406E70  $ 60           PUSHAD
00406E71  . BE 00604000  MOV ESI,animal02.00406000
00406E76  . 8DBE 00B0FFFF LEA EDI,DWORD PTR DS:[ESI+FFFFB000]
00406E7C  . 57           PUSH EDI
00406E7D  . 83CD FF      OR EBP,FFFFFFFF
00406E80  .v EB 10       JMP SHORT animal02.00406E92
00406E82    90           NOP
00406E83    90           NOP
00406E84    90           NOP
00406E85    90           NOP
00406E86    90           NOP
00406E87    90           NOP
00406E88  > 8A06         MOV AL,BYTE PTR DS:[ESI]
```

- OEP를 찾아야 한다!

OEP(Original Entry Point) : 실행프로그램의 실제 시작 지점

언패킹

```
C:₩Users₩djssl>upx -d -o ani.exe animal02.exe
                       Ultimate Packer for eXecutables
                          Copyright (C) 1996 - 2017
UPX 3.94w        Markus Oberhumer, Laszlo Molnar & John Reiser   May 12th 2017

        File size         Ratio      Format      Name
   --------------------   ------   -----------   -----------
      9728 <-      7168   73.68%    win32/pe     ani.exe

Unpacked 1 file.
```

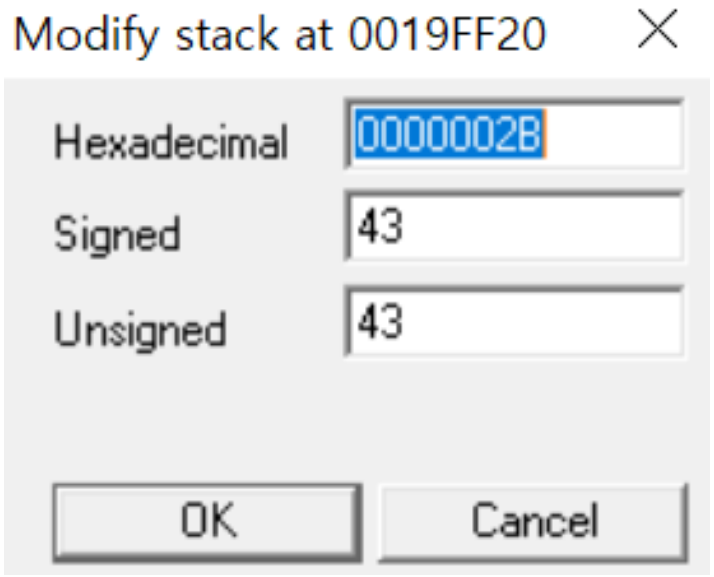
언패킹 방법 : upx -d -o ' 생성될 언패킹 파일의 경로' '언패킹에 사용할 파일 경로 '

언패킹

```
0040140E  $ E8 C4030000    CALL animal.__security_init_cookie
00401413  .^E9 74FEFFFF    JMP animal.__scrt_common_main_seh
00401418  ┌$ 55           PUSH EBP
00401419  │. 8BEC         MOV EBP,ESP
0040141B  │. 6A 00        PUSH 0                                  ┌pTopLevelFilter = NULL
0040141D  │. FF15 04204000 CALL DWORD PTR DS:[<&KERNEL32.SetUnhand └SetUnhandledExceptionFilter
00401423  │. FF75 08      PUSH DWORD PTR SS:[EBP+8]               ┌pExceptionInfo
00401426  │. FF15 20204000 CALL DWORD PTR DS:[<&KERNEL32.UnhandledE └UnhandledExceptionFilter
0040142C  │. 68 090400C0  PUSH C0000409                           ┌ExitCode = C0000409 (-1073740791.)
00401431  │. FF15 08204000 CALL DWORD PTR DS:[<&KERNEL32.GetCurrent ┌GetCurrentProcess
00401437  │. 50           PUSH EAX                                 │hProcess
00401438  │. FF15 0C204000 CALL DWORD PTR DS:[<&KERNEL32.TerminateP └TerminateProcess
0040143E  │. 5D           POP EBP
0040143F  └. C3           RETN
```

암호 키 찾기

```
004010FB  . 68 C4214000    PUSH OFFSET animal.??_C@_0BB@NDJJBHFE@?(  format = "첫 번째 암호키: "
00401100  . E8 1BFFFFFF    CALL animal.printf                        printf
00401105  . 8D45 F8        LEA EAX,DWORD PTR SS:[EBP-8]
00401108  . 50             PUSH EAX                                  Arg2
00401109  . 68 D8214000    PUSH OFFSET animal.??_C@_02DPKJAMEF@?$CF  format = "%d"
0040110E  . E8 3DFFFFFF    CALL animal.scanf                         scanf
00401113  . 817D F8 770700 CMP DWORD PTR SS:[EBP-8],777
```

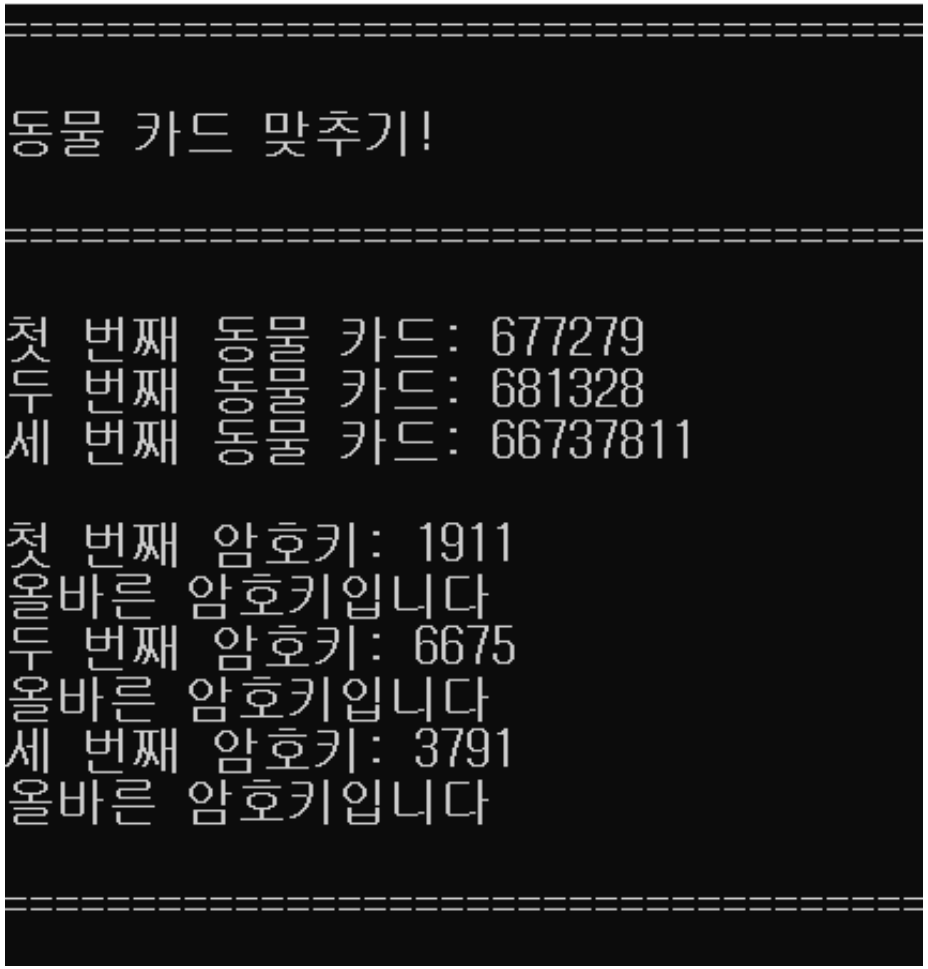


43과 1911 비교

첫 번째 암호키: 1911

두 번째 암호키: 6675

세 번째 암호키: 3791


동물 카드 맞추기!
첫 번째 동물 카드: 677279
두 번째 동물 카드: 681328
세 번째 동물 카드: 66737811
첫 번째 암호키: 1911
올바른 암호키입니다
두 번째 암호키: 6675
올바른 암호키입니다
세 번째 암호키: 3791
올바른 암호키입니다

어떻게 암호화 되었을까? - XOR연산

```
==================================

동물 카드 맞추기!

==================================

첫 번째 동물 카드: 677279
두 번째 동물 카드: 681328
세 번째 동물 카드: 66737811

첫 번째 암호키: 1911
올바른 암호키입니다
두 번째 암호키: 6675
올바른 암호키입니다
세 번째 암호키: 3791
올바른 암호키입니다

==================================
```

? 1911 = 677279

? 6675 = 681328

? 3791 = 66737811

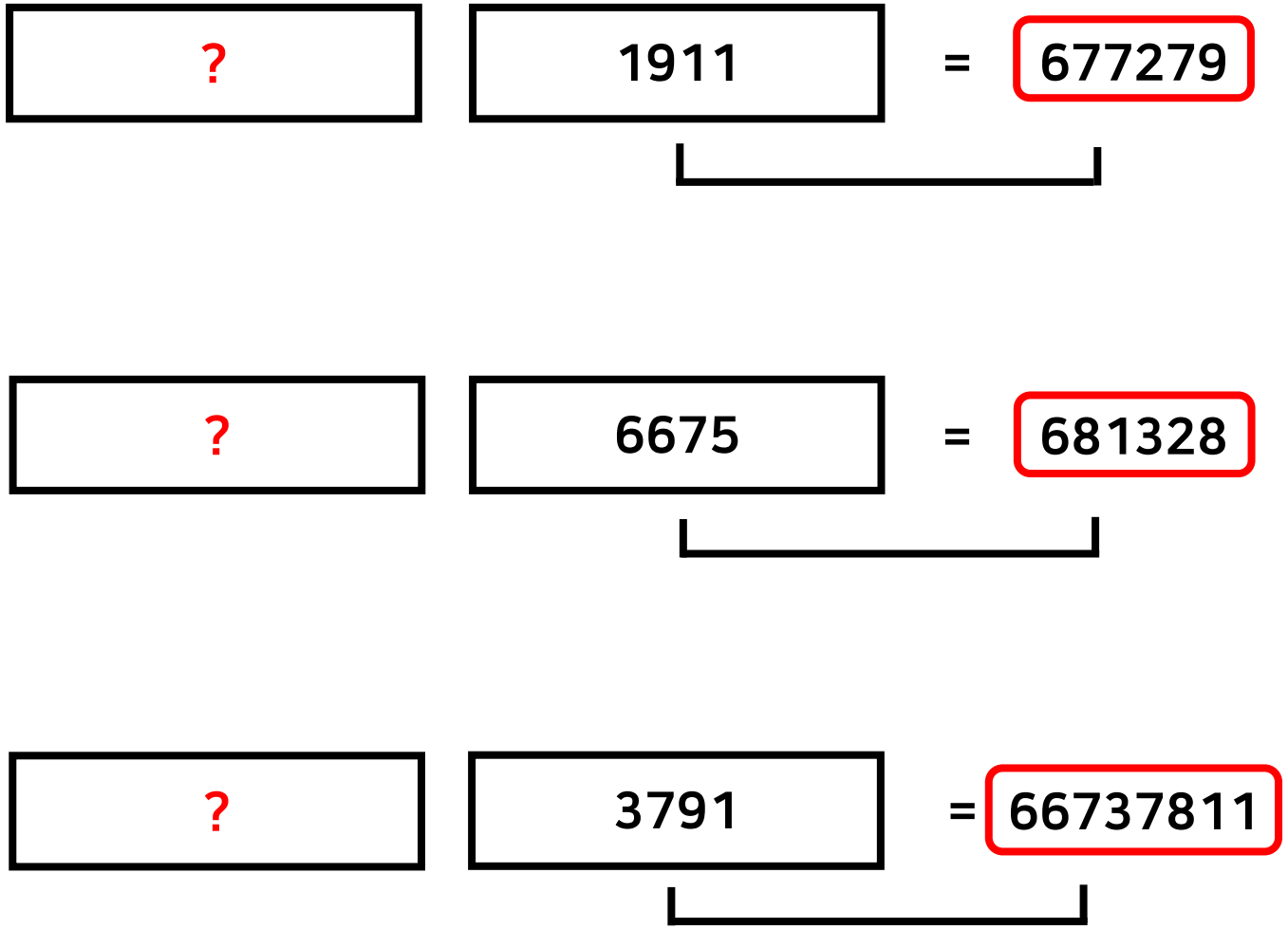
XOR연산이란?

: 두 명제 중에 정확히 하나만 참일 경우 참 값을 돌려주는 연산

XOR 연산	
입력값	결과값
0, 0	0
0, 1	1
1, 0	1
1, 1	0

XOR연산

```
==================================

동물 카드 맞추기!

==================================

첫 번째 동물 카드: 677279
두 번째 동물 카드: 681328
세 번째 동물 카드: 66737811

첫 번째 암호키: 1911
올바른 암호키입니다
두 번째 암호키: 6675
올바른 암호키입니다
세 번째 암호키: 3791
올바른 암호키입니다

==================================
```



XOR 연산 ①

```c
#include <stdio.h>
int main()
{
   int cryptograph_a = 677279, cryptograph_b = 681328, cryptograph_c = 66737811;    // 암호화 데이터
   int decrypt_a = 0, decrypt_b = 0, decrypt_c = 0;     // 복호화 데이터
   int key_a = 1911, key_b = 6675, key_c = 3791;     // 암·복호화에 사용하는 키

   decrypt_a = cryptograph_a ^ key_a; // 복호화
   decrypt_b = cryptograph_b ^ key_b;
   decrypt_c = cryptograph_c ^ key_c;

   printf("첫 번째 동물 카드 복호화 데이터: %d ₩n",decrypt_a);
   printf("두 번째 동물 카드 복호화 데이터: %d ₩n", decrypt_b);
   printf("세 번째 동물 카드 복호화 데이터: %d ₩n", decrypt_c);
   return 0;
}
```

XOR 연산 ①

```
#include <stdio.h>
int main()
{
   int cryptograph_a = 677279, cryptograph_b = 681328, cryptograph_c = 66737811;   // 암호화 데이터
   int decrypt_a = 0, decrypt_b = 0, decrypt_c = 0;     // 복호화 데이터
   int key_a = 1911, key_b = 6675, key_c = 3791;     // 암·복호화에 사용하는 키

   decrypt_a = cryptograph_a ^ key_a; // 복호화
   decrypt_b = cryptograph_b ^ key_b;
   decrypt_c = cryptograph_c ^ key_c;

   printf("첫 번째 동물 카드 복호화 데이터: %d ₩n",decrypt_a);
   printf("두 번째 동물 카드 복호화 데이터: %d ₩n", decrypt_b);
   printf("세 번째 동물 카드 복호화 데이터: %d ₩n", decrypt_c);
   return 0;
}
```

XOR 연산 ①

```
#include <stdio.h>
int main()
{
   int cryptograph_a = 677279, cryptograph_b = 681328, cryptograph_c = 66737811;    // 암호화 데이터
   int decrypt_a = 0, decrypt_b = 0, decrypt_c = 0;    // 복호화 데이터
   int key_a = 1911, key_b = 6675, key_c = 3791;    // 암·복호화에 사용하는 키

   decrypt_a = cryptograph_a ^ key_a; // 복호화
   decrypt_b = cryptograph_b ^ key_b;
   decrypt_c = cryptograph_c ^ key_c;

   printf("첫 번째 동물 카드 복호화 데이터: %d ₩n",decrypt_a);
   printf("두 번째 동물 카드 복호화 데이터: %d ₩n", decrypt_b);
   printf("세 번째 동물 카드 복호화 데이터: %d ₩n", decrypt_c);
   return 0;
}
```

XOR 연산 ①

```
#include <stdio.h>
int main()
{
   int cryptograph_a = 677279, cryptograph_b = 681328, cryptograph_c = 66737811;    // 암호화 데이터
   int decrypt_a = 0, decrypt_b = 0, decrypt_c = 0;    // 복호화 데이터
   int key_a = 1911, key_b = 6675, key_c = 3791;    // 암·복호화에 사용하는 키

   decrypt_a = cryptograph_a ^ key_a; // 복호화
   decrypt_b = cryptograph_b ^ key_b;
   decrypt_c = cryptograph_c ^ key_c;

   printf("첫 번째 동물 카드 복호화 데이터: %d ₩n",decrypt_a);
   printf("두 번째 동물 카드 복호화 데이터: %d ₩n", decrypt_b);
   printf("세 번째 동물 카드 복호화 데이터: %d ₩n", decrypt_c);
   return 0;
}
```

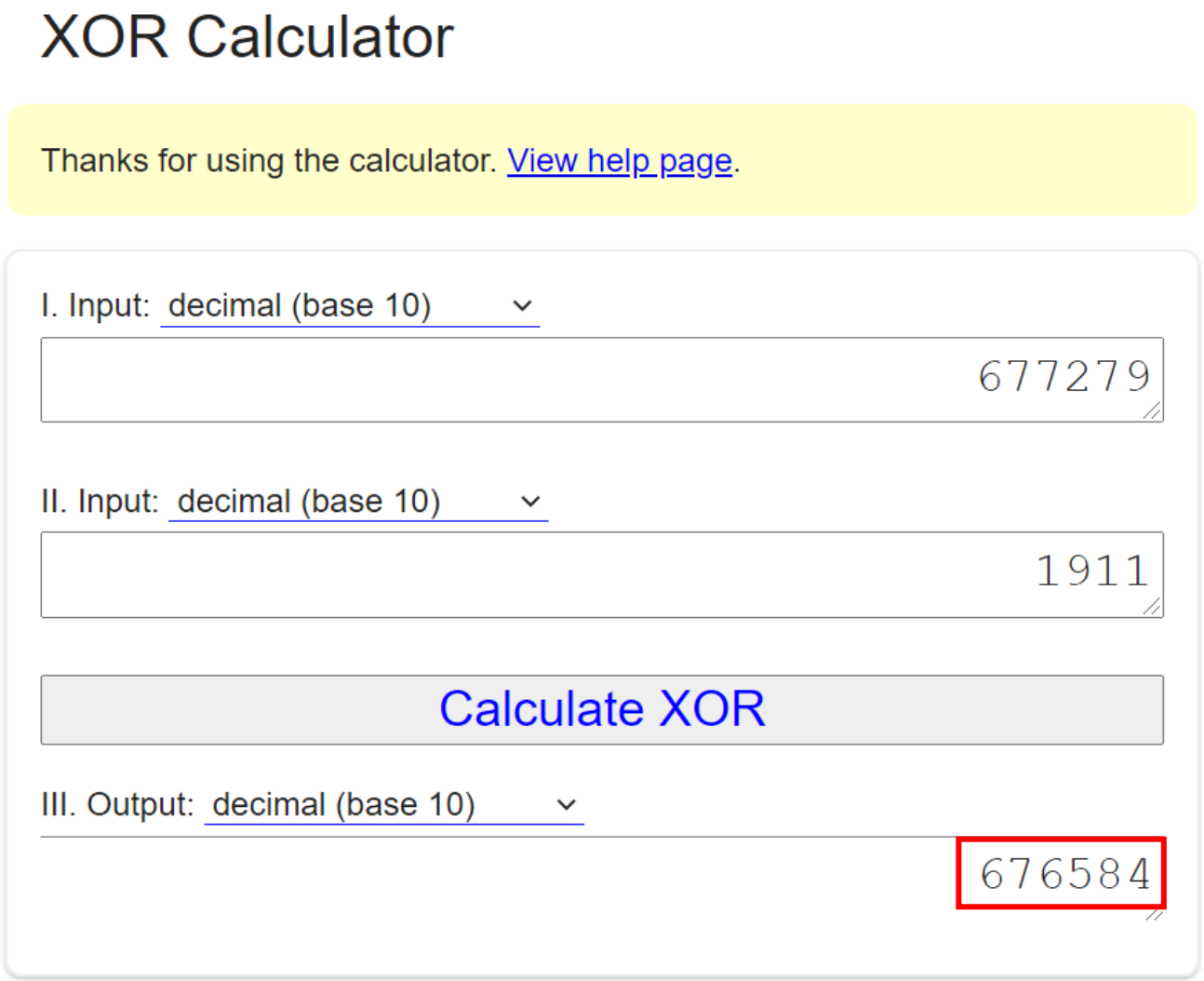
XOR 연산 ①

```
#include <stdio.h>
int main()
{
   int cryptograph_a = 677279, cryptograph_b = 681328, cryptograph_c = 66737811;    // 암호화 데이터
   int decrypt_a = 0, decrypt_b = 0, decrypt_c = 0;     // 복호화 데이터
   int key_a = 1911, key_b = 6675, key_c = 3791;     // 암·복호화에 사용하는 키

   decrypt_a = cryptograph_a ^ key_a; // 복호화
   decrypt_b = cryptograph_b ^ key_b;
   decrypt_c = cryptograph_c ^ key_c;

   printf("첫 번째 동물 카드 복호화 데이터: %d ₩n",decrypt_a);
   printf("두 번째 동물 카드 복호화 데이터: %d ₩n", decrypt_b);
   printf("세 번째 동물 카드 복호화 데이터: %d ₩n", decrypt_c);
   return 0;
}
```

XOR 연산 ①

```
첫 번째 동물 카드 복호화 데이터: 676584
두 번째 동물 카드 복호화 데이터: 687971
세 번째 동물 카드 복호화 데이터: 66738268
```

XOR 연산 ②


XOR Calculator
Thanks for using the calculator. View help page.
I. Input: decimal (base 10)
677279
II. Input: decimal (base 10)
1911
Calculate XOR
III. Output: decimal (base 10)
676584

아스키코드 변환

676584 → CAT

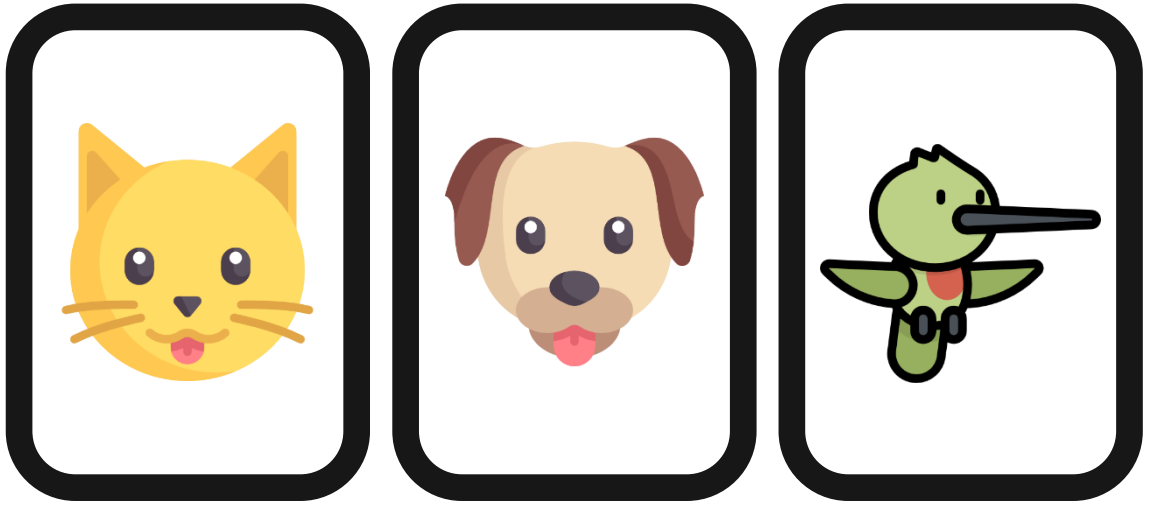
687971 → DOG

66738268 → BIRD

제어 문자 | 공백 문자 | 구두점 | 숫자 | 알파벳

10진	16진	문자	10진	16진	문자	10진	16진	문자	10진	16진	문자
0	0x00	NUL	32	0x20	SP	64	0x40	@	96	0x60	`
1	0x01	SOH	33	0x21	!	65	0x41	A	97	0x61	a
2	0x02	STX	34	0x22	"	66	0x42	B	98	0x62	b
3	0x03	ETX	35	0x23	#	67	0x43	C	99	0x63	c
4	0x04	EOT	36	0x24	$	68	0x44	D	100	0x64	d
5	0x05	ENQ	37	0x25	%	69	0x45	E	101	0x65	e
6	0x06	ACK	38	0x26	&	70	0x46	F	102	0x66	f
7	0x07	BEL	39	0x27	'	71	0x47	G	103	0x67	g
8	0x08	BS	40	0x28	(	72	0x48	H	104	0x68	h
9	0x09	HT	41	0x29	)	73	0x49	I	105	0x69	i
10	0x0A	LF	42	0x2A	*	74	0x4A	J	106	0x6A	j
11	0x0B	VT	43	0x2B	+	75	0x4B	K	107	0x6B	k
12	0x0C	FF	44	0x2C	,	76	0x4C	L	108	0x6C	l
13	0x0D	CR	45	0x2D	-	77	0x4D	M	109	0x6D	m
14	0x0E	SO	46	0x2E	.	78	0x4E	N	110	0x6E	n
15	0x0F	SI	47	0x2F	/	79	0x4F	O	111	0x6F	o
16	0x10	DLE	48	0x30	0	80	0x50	P	112	0x70	p
17	0x11	DC1	49	0x31	1	81	0x51	Q	113	0x71	q
18	0x12	DC2	50	0x32	2	82	0x52	R	114	0x72	r
19	0x13	DC3	51	0x33	3	83	0x53	S	115	0x73	s
20	0x14	DC4	52	0x34	4	84	0x54	T	116	0x74	t
21	0x15	NAK	53	0x35	5	85	0x55	U	117	0x75	u
22	0x16	SYN	54	0x36	6	86	0x56	V	118	0x76	v
23	0x17	ETB	55	0x37	7	87	0x57	W	119	0x77	w
24	0x18	CAN	56	0x38	8	88	0x58	X	120	0x78	x
25	0x19	EM	57	0x39	9	89	0x59	Y	121	0x79	y
26	0x1A	SUB	58	0x3A	:	90	0x5A	Z	122	0x7A	z
27	0x1B	ESC	59	0x3B	;	91	0x5B	[	123	0x7B	{
28	0x1C	FS	60	0x3C	<	92	0x5C	₩	124	0x7C	|
29	0x1D	GS	61	0x3D	=	93	0x5D	]	125	0x7D	}
30	0x1E	RS	62	0x3E	>	94	0x5E	^	126	0x7E	~
31	0x1F	US	63	0x3F	?	95	0x5F	_	127	0x7F	DEL

scpCTF{CAT_DOG_BIRD}



3. 문제 Review

난이도 : 중

요구 능력 : ① 패킹 여부를 확인하고 언패킹 할 수 있는가?

② CMP 비교구문을 통해 얻고자 하는 값을 읽어낼 수 있는가?

③ XOR 연산에 대해 이해하고 계산 코딩을 할 수 있는가? (XOR 계산기로도 가능)

④ 아스키코드 표를 이용해 10진수를 문자로 변환할 수 있는가?

감사합니다 ☺