# 1 LEVEL 1 amocafe write-up

아모카페에 오신 것을 환영합니다!

메뉴 번호를 입력하여 주문할 수 있는 웹 서비스가 작동하고 있습니다. 아모의 최애 메뉴를 대신 주문해 주면 아모가 플래그를 준다고 합니다. 첨부파일로 주어지는 웹 서비스의 코드를 분석하여 메뉴 번호를 알아 내세요!

플래그는 `flag.txt` 파일과 `FLAG` 변수에 있습니다.

문제에서 주어진 `flag.txt` 파일에 적혀있는 플래그는 sample 플래그입니다.

플래그 형식은 DH{…} 입니다.

---

코드를 한참 읽어봐도 나는 이해할 수 없었다.

그렇게 4시간이 지나던 때에 코드 조각을 하나씩 깨달았다.

```
for i in range (0, 16):
        res = (org >> (4 * i)) & 0xf
        if 0 < res < 12:
            # 'b'이면 '_'처리
            if ~res & 0xf == 0x4:
                st[16-i-1] = '_'
            else:
                st[16-i-1] = str(res)
        else:
            st[16-i-1] = format(res, 'x')
    menu_str = menu_str.join(st)
```

이 문제에서 가장 핵심이 되는 건 이 로직이다.

이게 무슨 의미인가 분석하려고 직접 하나씩 손코딩 했다. 먼저

```
res = (org >> (4 * i)) & 0xf
```

FLAG가 담겨있는 org 변수를 반복 횟수*4 만큼 오른쪽 시프트 연산한다.

그리고 0xf와 AND연산 한다.

이 의미는 FLAG[10:29]를 2진수로 변환해보면 알 수 있다.

1565678866679402738 & 15

0001 0101 1011 1010 0110 1000 1010 0010 0100 1000 1100 0011 0001 0000 1111 0010 & 1111

해보면 알겠지만 무조건 마지막 네 자리가 그대로 결과 값으로 나온다.

여기서 깊은 깨달음을 얻었다...

어떤 수든 1의 개수만큼 AND하면 우측에서부터 그대로 결과 값이라는 것을.

즉 0010 이 res에 담긴다.

```
        if 0 < res < 12:
            # 'b'이면 '_'처리
            if ~res & 0xf == 0x4:
                st[16-i-1] = '_'
```

내가 주석을 달아놓았는데,

"res가 0-11이고, res를 not 연산 한 뒤에 16진수 F와 AND 연산한 결과 값이 4이면" 을 요약하면

If res == 0xb: 이다.

만약 res가 11일 때 res의 NOT은 0100 이고 F와 AND를 해도 0100이다. 이것이 4이냐? 그렇다.

➔ 즉 res의 NOT이 4인지를 묻는 것이다. 이건 신기하게도 합이 15라는 규칙이 있었다.

10의 NOT은 0101(2)=5

7의 NOT은 1000(2)=8

13의 NOT은 0010(2)=2

또한 10은 그대로 출력되고 12(0xc)부터 16진수로 출력된다는 것을 알 수 있다.

따라서 python코딩으로 이 과정을 뒤집으면 될 것 같다.

그 전에 여기서 정리를 해보자.

이 코드는,

DH{55f23941565678866679402738395 75eb0af4b3f115345f0cc8b9}를 2진수로 바꿔서 오른쪽 끝 4자리씩 st[15,14,13...]에 저장하는 알고리즘이다.

➔ 1_c_3_c_0__ff_3e 나오기 위한 재료

<문제 코드를 그대로 가져와서 test 해보았다>

```
FLAG =
'DH{55f23941565678866679402738395 75eb0af4b3f115345f0cc8b9}'


menu_str = ''
# org = 2002760202557848382
#org = 0x1bcb3bcb0bbffb3e
org = FLAG[10:29]
org = int(org)
print(f'org: {org}')

st = ['' for i in range(16)]

for i in range (0, 16):
    res = (org >> (4 * i)) & 0xf
    if 0 < res < 12:
        # ~res 값이 11 이면 '_' 대입
        # res & 0xf == 0xb:
        if ~res & 0xf == 0x4:
            st[16-i-1] = '_'
        else:
            st[16-i-1] = str(res)
    else:
        st[16-i-1] = format(res, 'x')


menu_str = menu_str.join(st)
print(menu_str)
```

org: 1565678866679402738

15_106810248c310f2

우리는 DH{...}를 FLAG로 두었지만 "1_c_3_c_0__ff_3e"가 나오기 위한 FLAG 값을 구하는 문제이다

즉 15_106810248c310f2 이것은

DH{55f2394156567886667940273839575eb0af4b3f115345f0cc8b9} 의

알고리즘 연산인 것이다. 15_106810248c310f2 가 1_c_3_c_0__ff_3e로 출력되게 하려면 어떤 FLAG 값이 와야 하는가? 이것이 문제의 핵심이다.

나는 코드를 만들었다.

```
# 목적 문자열을 저장한다
menu = '1_c_3_c_0__ff_3e'
# 문자열을 하나씩 나눠서 저장한다.
menu = list(menu)

# index, data 를 나눠서 반복문을 돌린다.
for i, d in enumerate(menu):
    # '_'이면 11(b)로 저장
    if d == '_':
        menu[i] = 'b'
    # menu[i]가 12 이상이면 16 진수로 저장
    elif d == 'c':
        menu[i] = 'c'
    elif d == 'd':
        menu[i] = 'd'
    elif d == 'e':
        menu[i] = 'e'
    elif d == 'f':
        menu[i] = 'f'

# 쪼개진 list 들의 int 형을 str 로 한번에 바꾸고 s 로 저장한다.
s = ''.join(str(s) for s in menu)
# s 를 16 진수로 바꾼뒤 10 진수로 출력한다.
print(int(s, 16))
```

2002760202557848382

이것이 10진수로 바뀐 FLAG이다. FLAG를 넣어서

```
'1_c_3_c_0__ff_3e'
```

이것이 나오는지 확인해보자.

아까 FLAG를 테스트 해본 코드에 주석을 제거하고 기존에 FLAG를 주석처리한다.

```
menu_str = ''
# org = 2002760202557848382
org = 0x1bcb3bcb0bbffb3e
print(f'org: {org}')

출력:
1_c_3_c_0__ff_3e
```

성공적으로 출력되었으며 FLAG를 입력하면

"My favorite menu is 1_c_3_c_0__ff_3e"

menu number order

DH{ }

드디어 8시간의 원정이 끝났다.

~~난이도 레벨1 맞냐~~