

Открываем программу, вводим случайный пароль, получаем предсказуемый ответ

```
Please input the flag: thisisnotapassword
Sorry, please try again.
```

Идем в гидру, ищем вхождение строки. Находим функцию


```
undefined8 FUN_00401550(void)
{
    int check_flag;
    size_t user_input_length;
    size_t flag_length;
    unsigned long long uVar1;
    char user_input [39];
    undefined4 local_41;
    char flag_enc [28];
    int i;

    FUN_00401710();
    builtin_strncpy(flag_enc,"FK?DwObbkJ^di_V?_c0R4L\\Me",0x1a);
    printf("Please input the flag: ");
    scanf("%s",user_input);
    local_41 = 0;
    for (i = 0; uVar1 = (unsigned long long)i, user_input_length = strlen(user_input),
        uVar1 < user_input_length; i = i + 1) {
        flag_enc[i] = (char)i + flag_enc[i];
    }
    check_flag = strncmp(flag_enc,user_input,0x19);
    if (check_flag == 0) {
        user_input_length = strlen(user_input);
        flag_length = strlen(flag_enc);
        if (user_input_length == flag_length) {
            puts("Congratulations, you did it.");
            return 0;
        }
    }
    puts("Sorry, please try again.");
    return 0;
}
```

Хранится, видимо, зашифрованный флаг. Далее по логике программы он модифицируется через добавление к символу флага его позиции, а затем сравнивается с тем, что ввёл пользователь.

Тогда мы должны сделать то же самое – взять этот зашифрованный флаг, к каждому символу добавить его позицию и посмотреть, что получилось

```
< > main.py + ↕ 📄  
1 encrypted = "FK?DwObbkJ^di_V?_c0R4L\\Me"  
2 decrypted = ""  
3  
4 for i in range(len(encrypted)):  
5     decrypted += chr(ord(encrypted[i]) + i) # ПРИБАВЛЯЕМ, а не вычитаем  
6  
7 print(decrypted)
```

Powered by  trinket  
FLAG{ThisShouldNotBeHard}

Проверяем

```
Please input the flag: FLAG{ThisShouldNotBeHard}  
Congratulations, you did it.
```