```
IDA VIEW-A | PSEUDOCODE-A | HEX VIEW-1 | STRUCTURES | ENUMS | IMPORTS

1 int __cdecl main(int argc, const char **argv, const char **envp)
2 {
3   char v4; // [esp+16h] [ebp-1Ah]
4
5   __main();
6   random();
7   puts("Oh no, this damn program is wrong again. Where on earth is Flag!!!\nMaybe you could ask Forgotten?");
8   scanf("%s", &v4);
9   if ( strlen(&v4) == 25 && judge0(&v4, 26) )
10    puts("Congratulations! You found the flag!");
11  else
12    puts("Oh no, you can go and wash sleep.");
13  return 0;
14 }
```

打开main判断了长度是否等于25

然后judge0返回值非0的话就返回成功

```
1 signed int __cdecl judge0(_BYTE *a1)
2 {
3   signed int result; // eax
4
5   if ( *a1 != 'C' )
6     goto LABEL_12;
7   if ( a1[1] != 'T' )
8     return 0;
9   if ( a1[2] == 'F' && a1[3] == '{' && a1[24] == '}' && judge1(a1) )
10    result = 1;
11  else
12 LABEL_12:
13    result = 0;
14  return result;
15 }
```

判断输入前四位是否`CTF{`，最后一位是否为`}`，然后进入judge1函数

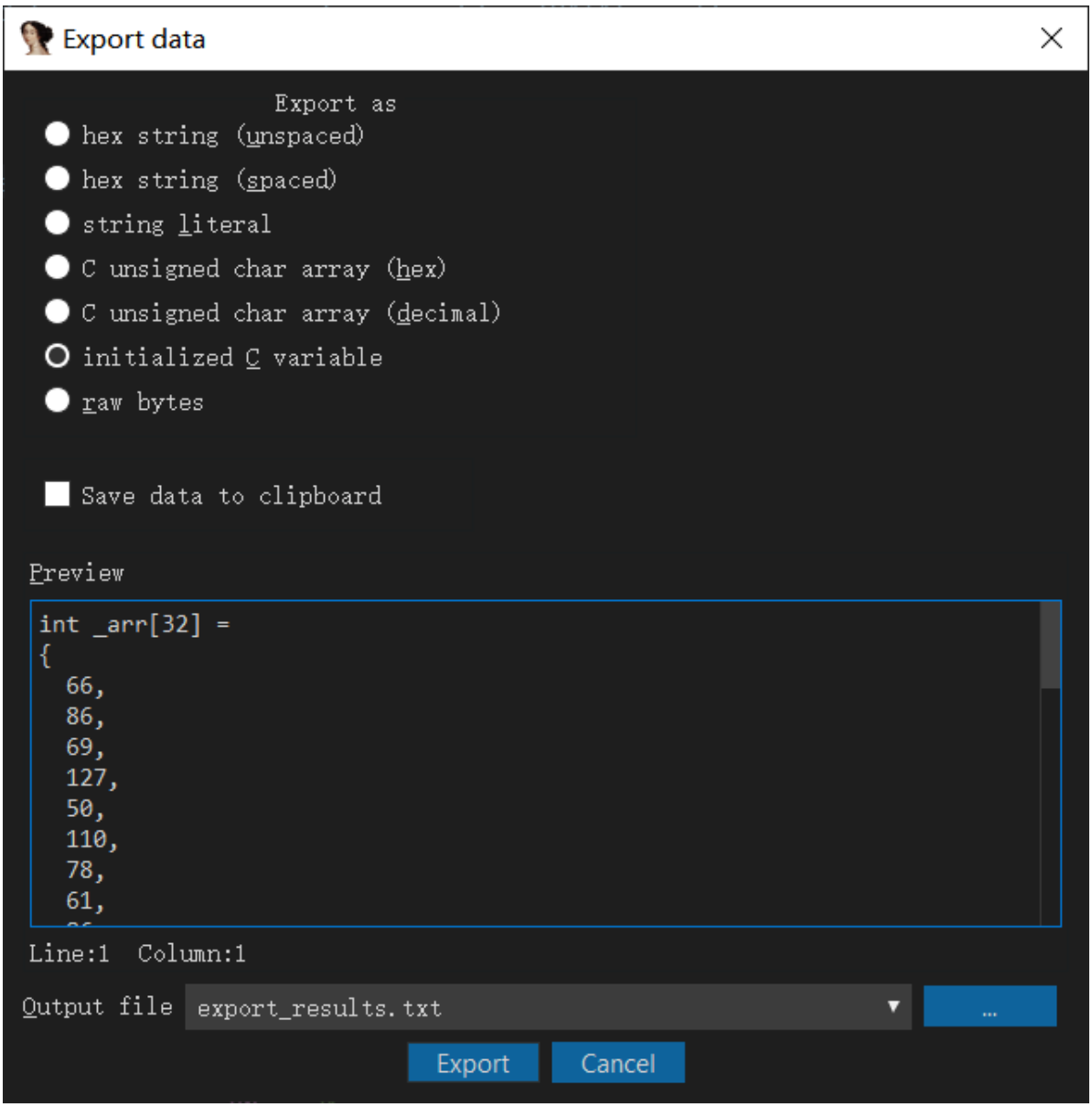
```
IDA VIEW-A | PSEUDOCODE-A | HEX

1 signed int __cdecl judge1(int a1)
2 {
3   signed int i; // [esp+Ch] [ebp-4h]
4   signed int j; // [esp+Ch] [ebp-4h]
5
6   for ( i = 0; i <= 25; ++i )
7   {
8     if ( *(i + a1) )
9       *(i + a1) ^= i + 1;
10  }
11  for ( j = 0; j <= 25; ++j )
12  {
13    if ( *(j + a1) != arr[j] )
14      return 0;
15  }
16  return 1;
17 }
```

指针和数组的转化`*(i+a1)==a1[i]`

将`a1[i]`异或后`i+1`后与数组`arr`比较

```
.data:00402000 _data            segment para public 'DATA' use32
.data:00402000                  assume cs:_data
.data:00402000                  ;org 402000h
.data:00402000                  public _arr
.data:00402000 ; int arr[32]
.data:00402000 _arr             dd 42h                  ; DATA XREF: _random+2E↑r
.data:00402000                                          ; _judge1+62↑r
.data:00402004                  db  56h ; V
.data:00402005                  db    0
.data:00402006                  db    0
.data:00402007                  db    0
.data:00402008                  db  45h ; E
.data:00402009                  db    0
.data:0040200A                  db    0
.data:0040200B                  db    0
.data:0040200C                  db  7Fh ;
.data:0040200D                  db    0
.data:0040200E                  db    0
.data:0040200F                  db    0
.data:00402010                  db  32h ; 2
.data:00402011                  db    0
.data:00402012                  db    0
.data:00402013                  db    0
.data:00402014                  db  6Eh ; n
.data:00402015                  db    0
.data:00402016                  db    0
.data:00402017                  db    0
.data:00402018                  db  4Eh ; N
.data:00402019                  db    0
.data:0040201A                  db    0
.data:0040201B                  db    0
.data:0040201C                  db  3Dh ; =
```

查看arr数据，shift+e提取数据



总体流程就是，先判断输入长度，flag头，然后将其输入与下标`i+1`异或，异或后与arr数组比较。

所以逆过来就是将arr数组与`i+1`异或得到flag

然后写脚本

与下标i+1进行异或得到flag

```
#include <stdio.h>
#include <string.h>
unsigned int arr[32] = {
    0x00000042, 0x00000056, 0x00000045, 0x0000007F, 0x00000032, 0x0000006E, 0x0000004E, 0x0000003D,
    0x00000056, 0x0000003B, 0x00000078, 0x00000053, 0x0000004C, 0x0000004F, 0x00000050, 0x00000062,
    0x00000074, 0x00000073, 0x0000007F, 0x0000004B, 0x00000073, 0x00000021, 0x00000056, 0x0000007F,
    0x00000064, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000
};
int main(void)
{
```

```
    int i;
    char flag[26] = { 0 };
    for (i = 0; i < 25; i++)
    {
        flag[i] = arr[i] ^ (i + 1);
    }
    puts(flag);
    return 0;
}

```