```
int sub_412960()
{
  char v1; // [sp+0h] [bp-10Ch]@1
  char v2; // [sp+Ch] [bp-100h]@1
  char input; // [sp+D0h] [bp-3Ch]@1
  unsigned int v4; // [sp+108h] [bp-4h]@1
  int savedregs; // [sp+10Ch] [bp+0h]@1

  memset(&v2, 0xCCu, 0x100u);
  v4 = (unsigned int)&savedregs ^ __security_cookie;
  printf("Input your flag:\n", v1);
  scanf_s("%s", &input, 50);
  if ( j_strlen_0(&input) == 30 )               // 判断输入长度是否为30
  {
    sub_4110A5(&input);
    if ( sub_411262(&input) )
      printf("Congratulation\nThe flag is your input\n", v1);
    else
      sub_4110E1();
    system("pause");
    sub_411181();
  }
  else
  {
    sub_4110E1();
  }
  sub_4112EE(&savedregs, &dword_412A3C);
  sub_411302();
  return sub_411181();
}
```

找到 main 函数

判断输入长度是否为 30 之后传入函数 sub_4110A5 跳的函数 sub_412420

函数 sub_412420

```
int __cdecl sub_412420(_DWORD a1)
{
  int result; // eax@1
  int v2; // [sp+Ch] [bp-D0h]@1
  int i; // [sp+D4h] [bp-8h]@1

  result = -858993460;
  memset(&v2, 0xCCu, 0xD0u);
  for ( i = 0; i < 600; ++i )
  {
    v2 = *(_DWORD *)&dword_41C000[4 * i];
    switch ( --v2 )
    {
      case 0:
        *(_BYTE *)(a1 + *(_DWORD *)&dword_41C000[4 * i + 2400]) += dword_41C000[4 * i + 4800];
        break;
      case 1:
        *(_BYTE *)(a1 + *(_DWORD *)&dword_41C000[4 * i + 2400]) -= dword_41C000[4 * i + 4800];
        break;
      case 2:
        *(_BYTE *)(a1 + *(_DWORD *)&dword_41C000[4 * i + 2400]) ^= dword_41C000[4 * i + 4800];
        break;
      case 3:
        *(_BYTE *)(a1 + *(_DWORD *)&dword_41C000[4 * i + 2400]) ^= 2 * dword_41C000[4 * i + 4800];
        break;
      default:
        break;
    }
    result = i + 1;
  }
  return result;
}
```

依据 0x41c000 处前 600 个 dword 型数据决定 case，第 601 个到第 1200 中 600 个数据是决定每次处理的是输入的第几个字节，第 1201 个到 1800 中 600 个数据是处理输入所用的数。（其实这就是一个[3][600]的二维数组）

处理完后在函数 sub_411262 中与 0x41dc20 中 30 个数据进行比较

```
signed int __cdecl sub_4123A0(int a1)
{
  char v2; // [sp+Ch] [bp-CCh]@1
  int i; // [sp+D0h] [bp-8h]@1

  memset(&v2, 0xCCu, 0xCCu);
  for ( i = 0; i < 30; ++i )
  {
    if ( *(_BYTE *)(i + a1) != byte_41DC20[i] )
      return 0;
  }
  return 1;
}
```

把数据 dump 下来用 py 处理一下或者直接 idc 搞也可以…….

Idc 脚本

```
#include<idc.idc>
static main()
{
    auto a,b,c,d,i;
    for(i=0;i<600;i++)
    {
        a=Byte(0x41c95c-i*4);
        b=Byte(0x41c95c-i*4+2400);
        c=Byte(0x41c95c-i*4+4800);
        a=a-1;
        if(a==0)
        {
            PatchByte(0x41dc20+b,Byte(0x41dc20+b)-c);
        }
        if(a==1)
        {
            PatchByte(0x41dc20+b,Byte(0x41dc20+b)+c);
        }
        if(a==2)
        {
            PatchByte(0x41dc20+b,Byte(0x41dc20+b)^c);
        }
        if(a==3)
        {
            PatchByte(0x41dc20+b,Byte(0x41dc20+b)^2*c);
        }
    }
    for(i=0;i<30;i++)
    {
        Message("%c",Byte(0x41dc20+i));
    }
}
```

```
412A3C: using guessed type int dword_412A3C;
zsctf{Turn_Loose_the_Mermaids}
```

Flag zsctf{Turn_Loose_the_Mermaids}