分析

```
//input 5 uint numbers ,x1,x2,x3,x4,x5
    //the sum of them should be MIN
    //这里输入5个无符号整形数，func2要求二进制表示时，有奇数个1，返回1
    //func3接收到1，返回1，表示奇数
    /*
    cout << func3< func2<x1> > << endl;
    cout << func3< func2<x2> > << endl;
    cout << func3< func2<x3> > << endl;
    cout << func3< func2<x4> > << endl;
    cout << func3< func2<x5> > << endl;
    */
    // output: 1 1 1 1 1


    //开根号
    //要求开根号之后等于接下来的五个数，由于是向下取整，会有很多满足的
    //!因此我在之前提到，要求这五个数的 累加和 最小
    //所以只会有一组解
    /*
    cout << _func1<x1>::result << endl;
    cout << _func1<x2>::result << endl;
    cout << _func1<x3>::result << endl;
    cout << _func1<x4>::result << endl;
    cout << _func1<x5>::result << endl;
    */
    //output: 963 4396 6666 1999 3141


    // 927369    963
    // 19324816  4396
    // 44435556  6666
    // 3996001   1999
    // 9865881   3141









    //素数返回1
    //预期的值是1229，因为1-10000有1229个素数
```

```
    //if you input 10000
    //how many "1" will func4<1>,func4<2>,fun4<3>......fun4<1000
    //x6 = count;


    // your flag is flag{x1-x2-x3-x4-x5-x6}
    // if x1=1,x2=2,x3=3,x4=4,x5=5,x6=6
    // flag is     flag{1-2-3-4-5-6}


    //于是最后的flag是
    //flag{927369-19324816-44435556-3996001-9865881-1229}
```

exp

用计算器就行了

举个例子，f(x1) == 963

计算器按一下，963的平方是927369

二进制表示是 1110 0010 0110 1000 1001，有奇数个1，所以已经符合了

> 如果不符合，就+1再判断

由此得出x1-x5 927369 19324816 44435556 3996001 9865881

关于编译期判断素数

随便写个代码跑一下，1到10000一共有1229个素数，x6就是1229

flag{927369-19324816-44435556-3996001-9865881-1229}