Hdu 4834

昨天做了一下，我对这类题目很苦手，+2哥请见谅我的效率……
一开始只找到一个线索，是从这里的题意得出的：

> - 若$(x+y)/2$为整数，那么$(x+y)/2$也属于S。

也就是说：

> - 如果将某个合法集合S里面的元素进行排序的话，相邻的两个元素x,y必定是是不符合第一点的。

为什么？

> 反证法：如果某个合法集合S进行排序之后，相邻两个元素x,y符合第一点，则表示$(x+y)/2$为整数，但因为$x<(x+y)/2<y$，且$(x+y)/2\notin S$（因为x,y已经相邻勒），与S为合法集合矛盾了。所以第二点得证啦。

但这样感觉还不够稳妥，所以就列了几个小例子：

> $n=1$时有$\{\},\{1\}$
> $n=2$时有$\{\},\{1\},\{2\},\{1,2\}$
> $n=3$时有$\{\},\{1\},\{2\},\{1,2\},\{3\},\{2,3\},\{1,2,3\}$

列举到3的时候就会发现n的所有合法集合$\{S\}_n$,必定包含$n-1$的合法集合$\{S\}_{n-1}$，而$\{S\}_n$和$\{S\}_{n-1}$的唯一区别，就是多了以n结尾的那些结合，也就是说它们之间有睇推关系！！！:

> 定义$C(n)$为$|\{S\}_n|$，即合法集合数，则有$C(n)=C(n-1)+\Delta_n$

下面举出增量Δ:

> $n=4$时多了$\{4\},\{3,4\},\{2,3,4\},\{1,2,3,4\},\{1,4\}$
> $n=5$时多了$\{5\},\{4,5\},\{3,4,5\},\{2,3,4,5\},\{1,2,3,4,5\},\{2,5\}$
> $n=6$时多了
> $\{6\},\{5,6\},\{4,5,6\},\{3,4,5,6\},\{2,3,4,5,6\},\{1,2,3,4,5,6\},\{3,6\},\{1,6\}$
> $n=7$时多了
> $\{7\},\{6,7\},\{5,6,7\},\{4,5,6,7\},\{3,4,5,6,7\},\{2,3,4,5,6,7\},\{1,2,3,4,5,6,7\},\{4,7\},\{1,4,7\},\{2,7\}$

列举到这里的时候，基本上可以明白第二点勒吧？第二点其实还可以推导出：

> - 某个合法集合S经过排序后，相邻两个元素必定是一奇一偶。

这个应该很容易想到吧，因为同奇同偶才能整除2嘛。从上面第三点又可以得出:

> - 某个合法集合S经过排序后，相邻两个元素的差必定是奇数。

因为如果是偶数，就会是同奇同偶勒。所以又得出第四点：

> - 合法集合们其实就是公差为奇数的等差数列。

相邻两个元素的差必定是奇数不就是这个意思咯……那么：

> - Δ_n就是以n结尾的公差为奇数的等差数列的个数。

最后得出一个递推式：

$$\begin{cases} C(n) = 2 & \text{n=1} \\ C(n) = C(n-1) + \Delta_n & \text{n>1} \end{cases}$$

那么问题就转化为如何求以n结尾的公差为奇数的等差数列的个数了（变简单勒吧^_^）。

When you can’t solve the hard one, try the easier version.

我们把上面的问题简化一下，例如：

> 如何求以n结尾的公差为1的等差数列的个数。

这个答案瞬间就出来了吧……
例如$n = 7$时:

$$\{7\}, \{6, 7\}, \{5, 6, 7\}, \{4, 5, 6, 7\}, \{3, 4, 5, 6, 7\}, \{2, 3, 4, 5, 6, 7\}, \{1, 2, 3, 4, 5, 6, 7\}$$

就是7个啦。那么第二个问题来了：

> 如何求以n结尾的公差为3的等差数列的个数。

再看下$n = 7$:

$$\{4, 7\}, \{1, 4, 7\}$$

2个。那么第三个问题来了：

> 如何求以n结尾的公差为5的等差数列的个数。

继续

$$\{2, 7\}$$

1个。
这里面好像有一个规律吧：

> 以n结尾的公差为odd的等差数列的个数等于$(n-1)/odd$取整加一。

原因其实很简单，$\lfloor (n-1)/odd \rfloor$（为什么要减一？因为这题的数是从1开始的）的意义可以理解为$[1,n]$里面有多少个odd,这些odd的个数其实就是等于最长的以n结尾的公差为odd的等差数列的长度，而长度减一和所求数列个数是一一对应的（不减一的话$\{n\}$就会被统计多次勒），最后的加一是这个$\{n\}$，一个数也算是等差数列。

那么回到原来未简化的版本，我们就知道勒以n结尾的公差为奇数的等差数列的个数其实等于：

$$\Delta_n = 1 + \sum_{1<=odd<n} \lfloor (n-1)/odd \rfloor$$

那么题目的答案就变成勒：

$$\begin{cases} C(n) = 2 & \text{n=1} \\ C(n) = C(n-1) + (1 + \sum_{1<=odd<n} \lfloor (n-1)/odd \rfloor) & \text{n>1} \end{cases}$$

由于题目的n是$1e7$的，T是$1e5$的，典型的预处理题目。
接下来出问题了，直接推上面那个公式可是要$O(n^2)$的，所以要优化一下Δ_n的求值方法。

$$\Delta_n = 1 + \sum_{1<=odd<n} \lfloor (n-1)/odd \rfloor$$

好吧，我一眼真的看不出什么，真的是压力山大，只有举例子勒。

Scrath small cases is faster than guess.

$$\begin{aligned}
&\Delta_1 = 1 + 0/1 = 1 \\
&\Delta_2 = 1 + 1/1 = 2 \\
&\Delta_3 = 1 + 2/1 = 3* \\
&\Delta_4 = 1 + 3/1 + 3/3 = 5 \\
&\Delta_5 = 1 + 4/1 + 4/3 = 6 \\
&\Delta_6 = 1 + 5/1 + 5/3 + 5/5 = 8* \\
&\Delta_7 = 1 + 6/1 + 6/3 + 6/5 = 10* \\
&\Delta_8 = 1 + 7/1 + 7/3 + 7/5 + 7/7 = 12* \\
&\Delta_9 = 1 + 8/1 + 8/3 + 8/5 + 8/7 = 13 \\
&\Delta_{10} = 1 + 9/1 + 9/3 + 9/5 + 9/7 + 9/9 = 16*
\end{aligned}$$

规律终于出来勒（汗）。
注意带星的行，它们与前一行的差都偏大勒，我们单看分母相同的列，发现1的列是每行都增加一次，3的列是每3行增加一次，5的列是每5行增加一次，也就是说第n行增加的数量其实就是等于$n-1$能整除的奇数个数……。
那么我们把Δ_n的$\sum_{1<=odd<n} \lfloor (n-1)/odd \rfloor$部分独立出来：

$$\Delta'_n = \sum_{1<=odd<n} \lfloor (n-1)/odd \rfloor$$

并且设：

$nodd(n) = n - 1$能整除的奇数个数。

那么根据上面的例子可以得出下面睇推式:

$$\begin{cases} \Delta'_n = 0 & \text{n=1} \\ \Delta'_n = \Delta'_{n-1} + nodd(n-1) & \text{n>1} \end{cases}$$

那$n-1$能整除的奇数个数怎么搞出来？一个简单的方法就是对每个数做$O(\sqrt{n})$的因子分解，求出奇因子个数:

```
for (int i = 0; i < N; i++)
    for (int j = 3; j*j <= i; j += 2)
        if (i%j == 0)
            nodd[i]++;
```

总复杂度$O(n\sqrt{n})$，显然是不行的。观察上面的循环我们可以发现，对于每个奇数j，它的每个倍数都会加一吧，从奇数的角度出发，就可以排除很多重复的操作。

```
for (int i = 1; i < N; i += 2)
    for (int j = i; j < N; j += i)
        nodd[j]++;
```

这个不就是筛法嘛……时间复杂度是$O(n\ln n)$（$\ln n$是和谐级数的上界）。
好了最后一个问题都解决了……
最后得到的递推式们就是：

$$\begin{cases} C(n) = 2 & \text{n=1} \\ C(n) = C(n-1) + \Delta_n & \text{n>1} \end{cases}$$

$$\Delta_n = 1 + \Delta'_n$$

$$\begin{cases} \Delta'_n = 0 & \text{n=1} \\ \Delta'_n = \Delta'_{n-1} + nodd(n-1) & \text{n>1} \end{cases}$$

预处理之后每个case直接输出C(n)就可以勒。
最终代码：

```
#include <iostream>
#include <cmath>
using namespace std;
const int N = 1e7+10;

long long C[N];
long long delta[N];
long long nodd[N];

void Init()
{
  for (int i = 1; i < N; i += 2)
    for (int j = i; j < N; j += i)
      nodd[j]++;

  delta[1] = 0;
  for (int i = 2; i < N; i++)
    delta[i] = delta[i-1]+nodd[i-1];
  for (int i = 1; i < N; i++)
```

```
    delta[i]++;

  C[0] = 1;
  for (int i = 1; i < N; i++)
    C[i] = C[i-1]+delta[i];
}

int main()
{
  cin.sync_with_stdio(false);

  Init();
  int T;
  cin >> T;
  for (int kas = 1; kas <= T; kas++) {
    int n;
    cin >> n;
    cout << "Case #" << kas << ":\n" << C[n] << endl;
  }
  return 0;
}
```

这道题需要多做一些数论题目才能有感觉的，你可以先锻炼一下这方面的基本题目，再来看可能就好理解勒。