1. 多组样例没有刷新dp数组为0或者没初始化别的会被上组数据影响的数组
2. unsigned long long的数据范围是0~2^64-1，long long的数据范围是-2^63~2^63-1
3. unsigned int 的数据范围是0～2^32-1，int的数据范围是-2^31~2^31-1
4. if或for括号{}少打
5. 行末得有换行符，不能有多余空格
6. string增长不要用str=str+char，这个式子的右值会生成一个新string对象再赋值给左值，超级无敌慢，换成str+=char超级无敌快。str1=str2可以换成str1=move(str2)，会快很多。
7. string头插入char和尾插入char可以写str.insert(str.begin(),char)，str.insert(str.end(),char) / str+=char（后者更好），头插入str和尾插入str可以写str1.insert(0,str2)，str1.insert(str1.size(),str2) / str1+=str2。尾部插用.push_back()
8. 综上要避免string左右值反复赋值。
9. 整数分块写成递归了。
10. 初始化dp数组时，没有把无效状态刷成INF，初始化从0开始到上界，边界情况要仔细预处理。
11. __lg是求以2为底，向下取整的函数，很快。log是求以e为底的函数，很慢，如果需要重复使用，记忆化可以快不是一点点。
12. 多组样例时，不要用Heltion的快速FFT的板子。
13. while和if中`==`写成了`=`。
14. 循环内转移的`i`写成了`n`。
15. 外层循环已经有`i`的情况下，内存循环错误使用了`i`。
16. 读入数据的过程中如果有特判可以直接判断无解，需要将本组数据全部读完之后再return，不然会影响下组数据的读入，可以写个tri记录结果。
17. 不要看错板子里的ϕ和μ。
18. 读题要读清楚，到底要不要行末空格，如果没有说明，就按没有行末空格处理。
19. Printf忘记加\n。
20. 用了char[]读字符串，就不要开string。
21. 比较公共子串之后的一个不公共字符时，需要考虑是否公共子串已经到了串的末尾，如果不判断<n会引起数组越界。
22. 1e6后面有6个0，如果是手算数据范围，一定要数清楚有多少个0。
23. 循环内变量的`i`和`j`写反
24. 小数点位数少的直接当整数处理，浮点数eps容易烂。
25. 有涉及到边界的问题，可以把单边的`=`去掉。
26. `++i`写成了`--i`或`--i`写成了`++i`
27. O(1)能计算出答案的结果，记忆化与不记忆化速度差别不大，使用unordered_map记忆化会慢很多，如果数据范围不大，数组记忆化会快一点。
28. 如果出现匪夷所思的输出，看看是不是没读入数据。
29. 提交的时候没有注释freopen
30. 如果计算实在觉得没问题还wa，多半是特判错了，需要把特判多写几组看看。
31. 每次交之前检查IOFast()，检查文件头有没有#define int long long
32. 1<<i超出int大小，需要写成1ll<<i
33. maxn开小了，导致RE，甚至是TLE
34. while循环内，pos++时，要记得判断是否超出边界。
35. 别用ceil，上取整(x+y-1)/y
36. 预处理前缀阶乘时，考虑是否应该是前缀lcm