目录

[C. Angry FFF Party](#_Toc20638_WPSOffice_Level1) [1](#_Toc20638_WPSOffice_Level1)

[D.Match Stick Game](#_Toc31648_WPSOffice_Level1) [4](#_Toc31648_WPSOffice_Level1)

[H. Coloring Game](#_Toc32748_WPSOffice_Level1) [6](#_Toc32748_WPSOffice_Level1)

[I. Max answer](#_Toc22319_WPSOffice_Level1) [6](#_Toc22319_WPSOffice_Level1)

[J. Distance on the tree](#_Toc3557_WPSOffice_Level1) [7](#_Toc3557_WPSOffice_Level1)

[K. MORE XOR](#_Toc14708_WPSOffice_Level1) [11](#_Toc14708_WPSOffice_Level1)

[M. Subsequence](#_Toc30755_WPSOffice_Level1) [13](#_Toc30755_WPSOffice_Level1)

## Angry FFF Party

先矩阵快速幂暴力打表，然后贪心求，最后小于10的情况都枚举单独处理 (好像只需要枚举<=5)

**import** java**.**util**.**Scanner**;**

**import** java**.**math**.**BigInteger**;**

**import** static java**.**math**.**BigInteger**.\*;**

public class Main **{**

public static void main**(**String**[]** args**)** **{**

Scanner in **=** **new** Scanner**(**System**.**in**);**

BigInteger res**[]** **=** **new** BigInteger**[**30**];**

BigInteger a**[][]** **=** **new** BigInteger**[**2**][**2**];**

BigInteger b**[][];**

**for** **(**int i **=** 1**;** i **<** 30**;** i**++)** **{**

a**[**0**][**0**]** **=** ONE**;** a**[**0**][**1**]** **=** ONE**;**

a**[**1**][**0**]** **=** ONE**;** a**[**1**][**1**]** **=** ZERO**;**

b **=** matPow**(**a**,** BigInteger**.**valueOf**(**i**-**1**),** 2**);**

a**[**0**][**0**]** **=** ONE**;** a**[**0**][**1**]** **=** ONE**;**

a**[**1**][**0**]** **=** ONE**;** a**[**1**][**1**]** **=** ZERO**;**

b **=** matPow**(**a**,** b**[**0**][**0**].**subtract**(**ONE**),** 2**);**

res**[**i**]** **=** b**[**0**][**0**];**

**}**

int T **=** in**.**nextInt**();**

int ans**[]** **=** **new** int**[**30**];**

**while** **((**T**--)** **>** 0**)** **{**

BigInteger W **=** in**.**nextBigInteger**();**

int tot**=**0**;**

**for** **(**int i **=** 29**;** i **>=** 1**;** i**--)** **{**

**if** **(**W**.**compareTo**(**TEN**)<**0**)** **{**

**if** **(**W**.**compareTo**(**BigInteger**.**valueOf**(**9**))** **==** 0**)** **{**

ans**[**tot**++]** **=** 5**;**

ans**[**tot**++]** **=** 4**;**

ans**[**tot**++]** **=** 2**;**

ans**[**tot**++]** **=** 1**;**

**}**

**else** **if** **(**W**.**compareTo**(**BigInteger**.**valueOf**(**8**))** **==** 0**)** **{**

ans**[**tot**++]** **=** 5**;**

ans**[**tot**++]** **=** 3**;**

ans**[**tot**++]** **=** 2**;**

ans**[**tot**++]** **=** 1**;**

**}**

**else** **if** **(**W**.**compareTo**(**BigInteger**.**valueOf**(**7**))** **==** 0**)** **{**

ans**[**tot**++]** **=** 5**;**

ans**[**tot**++]** **=** 2**;**

ans**[**tot**++]** **=** 1**;**

**}**

**else** **if** **(**W**.**compareTo**(**BigInteger**.**valueOf**(**6**))** **==** 0**)** **{**

ans**[**tot**++]** **=** 5**;**

ans**[**tot**++]** **=** 1**;**

**}**

**else** **if** **(**W**.**compareTo**(**BigInteger**.**valueOf**(**5**))** **==** 0**)** **{**

ans**[**tot**++]** **=** 4**;**

ans**[**tot**++]** **=** 3**;**

ans**[**tot**++]** **=** 2**;**

ans**[**tot**++]** **=** 1**;**

**}**

**else** **if** **(**W**.**compareTo**(**BigInteger**.**valueOf**(**4**))** **==** 0**)** **{**

ans**[**tot**++]** **=** 4**;**

ans**[**tot**++]** **=** 2**;**

ans**[**tot**++]** **=** 1**;**

**}**

**else** **if** **(**W**.**compareTo**(**BigInteger**.**valueOf**(**3**))** **==** 0**)** **{**

ans**[**tot**++]** **=** 3**;**

ans**[**tot**++]** **=** 2**;**

ans**[**tot**++]** **=** 1**;**

**}**

**else** **if** **(**W**.**compareTo**(**BigInteger**.**valueOf**(**2**))** **==** 0**)** **{**

ans**[**tot**++]** **=** 2**;**

ans**[**tot**++]** **=** 1**;**

**}**

**else** **if** **(**W**.**compareTo**(**BigInteger**.**valueOf**(**1**))** **==** 0**)** **{**

ans**[**tot**++]** **=** 1**;**

**}**

W **=** ZERO**;**

**}**

**if** **(**W**.**compareTo**(**res**[**i**])>=**0**)** **{**

ans**[**tot**++]** **=** i**;**

W **=** W**.**subtract**(**res**[**i**]);**

**}**

**}**

**if** **(**W**.**compareTo**(**ZERO**)** **==** 0**)** **{**

**for** **(**int i**=**tot**-**1**;**i**>**0**;--**i**)** **{**

System**.**out**.**print**(**ans**[**i**]+**" "**);**

**}**

System**.**out**.**println**(**ans**[**0**]);**

**}**

**else** **{**

System**.**out**.**println**(-**1**);**

**}**

**}**

**}**

static BigInteger**[][]** matMul**(**BigInteger a**[][],** BigInteger b**[][],** int n**)** **{**

BigInteger res**[][]** **=** **new** BigInteger**[**n**][**n**];**

**for** **(**int i **=** 0**;** i **<** n**;** i**++)**

**for** **(**int j **=** 0**;** j **<** n**;** j**++)** **{**

res**[**i**][**j**]** **=** ZERO**;**

**for** **(**int k **=** 0**;** k **<** n**;** k**++)**

res**[**i**][**j**]** **=** res**[**i**][**j**].**add**(**a**[**i**][**k**].**multiply**(**b**[**k**][**j**]));**

**}**

**return** res**;**

**}**

static BigInteger**[][]** matPow**(**BigInteger a**[][],** BigInteger p**,** int n**)** **{**

BigInteger res**[][]** **=** **new** BigInteger**[**n**][**n**];**

**for** **(**int i **=** 0**;** i **<** n**;** i**++)**

**for** **(**int j **=** 0**;** j **<** n**;** j**++)**

**if** **(**i **==** j**)** res**[**i**][**j**]** **=** ONE**;**

**else** res**[**i**][**j**]** **=** ZERO**;**

**while** **(**p**.**compareTo**(**ZERO**)** **!=** 0**)** **{**

**if** **(**p**.**and**(**ONE**).**compareTo**(**ONE**)** **==** 0**)** res **=** matMul**(**res**,** a**,** n**);**

a **=** matMul**(**a**,** a**,** n**);**

p **=** p**.**divide**(**BigInteger**.**valueOf**(**2**));**

**}**

**return** res**;**

**}**

**}**

## Match Stick Game

先DP预处理用i位数用j个火柴组成的最大值和最小值

然后每组数据继续用这个思路DP即可

#include<cstdio>

#include<iostream>

#include<string>

#include<cstring>

using namespace std**;**

int g**[]={**6**,**2**,**5**,**5**,**4**,**5**,**6**,**3**,**7**,**6**};**

int T**,**tot**,**n**,**ge**,**a**[**115**];**

long long ans**[**55**][**2005**],**fmax**[**15**][**125**],**fmin**[**15**][**1235**];**

int main**()**

**{**

scanf**(**"%d"**,&**T**);**

//位数 个数

**for(**int j**=**0**;**j**<=**9**;**j**++)**

**for(**int i**=**0**;**i**<=**120**;**i**++)**

fmax**[**j**][**i**]=**fmin**[**j**][**i**]=-**1**;**

fmax**[**0**][**0**]=**0**;**

fmin**[**0**][**0**]=**0**;**

**for(**int i**=**0**;**i**<**9**;**i**++)**

**for(**int j**=**0**;**j**<=**100**;**j**++)**

**if(**fmax**[**i**][**j**]>=**0**)**

**{**

// printf("%d %d %lld %lld\n",i,j,fmax[i][j],fmin[i][j]);

**for(**int k**=**0**;**k**<=**9**;**k**++)**

**{**

**if(**fmax**[**i**+**1**][**j**+**g**[**k**]]==-**1**)**

fmax**[**i**+**1**][**j**+**g**[**k**]]=**fmax**[**i**][**j**]\***10**+**k**;**

**else**

fmax**[**i**+**1**][**j**+**g**[**k**]]=**max**(**fmax**[**i**+**1**][**j**+**g**[**k**]],**fmax**[**i**][**j**]\***10**+**k**);**

**if(**fmin**[**i**+**1**][**j**+**g**[**k**]]==-**1**)**

fmin**[**i**+**1**][**j**+**g**[**k**]]=**fmin**[**i**][**j**]\***10**+**k**;**

**else**

fmin**[**i**+**1**][**j**+**g**[**k**]]=**min**(**fmin**[**i**+**1**][**j**+**g**[**k**]],**fmin**[**i**][**j**]\***10**+**k**);**

**}**

**}**

**while(**T**--)**

**{**

char c**;**

tot**=**1**;**

ge**=**0**;**

scanf**(**"%d"**,&**n**);**

memset**(**ans**,**0**,**sizeof ans**);**

memset**(**a**,**0**,**sizeof a**);**

**for(**int i**=**1**;**i**<=**n**;**i**++)**

**{**

cin**>>**c**;**

**if(**c**==**'+'**||**c**==**'-'**)**

**{**

tot**++;**

ge**+=**1**+(**c**==**'+'**);**

**}**

**else**

**{**

a**[**tot**]++;**

ge**+=**g**[**c**-**'0'**];**

**}**

**}**

**for(**int i**=**0**;**i**<=**tot**;**i**++)**

**for(**int j**=**0**;**j**<=**ge**;**j**++)**

ans**[**i**][**j**]=-**999999999999999999999LL**;**

ans**[**0**][**0**]=**0**;**

**for(**int i**=**0**;**i**<**tot**;**i**++)**

**for(**int j**=**0**;**j**<=**ge**;**j**++)**

**{**

**if(**i**==**0**)**

**{**

**for(**int k**=**0**;**k**<=**100**;**k**++)**

**if(**fmax**[**a**[**i**+**1**]][**k**]>=**0**)**

**{**

ans**[**i**+**1**][**j**+**k**]=**max**(**ans**[**i**+**1**][**j**+**k**],**ans**[**i**][**j**]+**fmax**[**a**[**i**+**1**]][**k**]);**

**}**

**}**

**else**

**{**

**for(**int k**=**0**;**k**<=**100**;**k**++)**

**if(**fmax**[**a**[**i**+**1**]][**k**]>=**0**)**

**{**

ans**[**i**+**1**][**j**+**k**+**2**]=**max**(**ans**[**i**+**1**][**j**+**k**+**2**],**ans**[**i**][**j**]+**fmax**[**a**[**i**+**1**]][**k**]);**

ans**[**i**+**1**][**j**+**k**+**1**]=**max**(**ans**[**i**+**1**][**j**+**k**+**1**],**ans**[**i**][**j**]-**fmin**[**a**[**i**+**1**]][**k**]);**

**}**

**}**

**}**

printf**(**"%lld\n"**,**ans**[**tot**][**ge**]);**

**}**

**}**

1. Coloring Game

因为只有两行，所以从左上角走到右下角，必须每列至少走过一个

因为开始和结尾两列有一个必须为黑，另一个可以走或不走，2种方案

中间的列有3钟方案 （黑黑，黑白，白黑），共有2\*pow(3,n-2)\*2种情况

#include<cstdio>

typedef long long ll**;**

const int mod**=**1e9**+**7**;**

ll pow\_mod**(**ll a**,**ll b**)** **{**

ll res**=**1**;**

**while** **(**b**)** **{**

**if** **(**b**&**1**)** res**=**res**\***a**%**mod**;**

a**=**a**\***a**%**mod**;**

b**>>=**1**;**

**}**

**return** res**;**

**}**

int n**;**

int main**()** **{**

scanf**(**"%d"**,&**n**);**

**if** **(**n**==**1**)** **return** 0**\***printf**(**"1\n"**);**

**if** **(**n**==**2**)** **return** 0**\***printf**(**"4\n"**);**

ll res**=**pow\_mod**(**3**,**n**-**2**)\***4**%**mod**;**

printf**(**"%lld\n"**,**res**);**

**return** 0**;**

**}**

## Max answer

RMQ-ST表实现，支持查询最小前缀和(minsum)，最大前缀和(maxsum)，原数组最小值位置

对于区间[L,R] ,先找到区间最小值位置P，考虑覆盖最小值的情况，剩下的都不会覆盖最小值

递归的处理[L,P-1]和[P+1,R]就行了

覆盖最小值的情况：

最小值大于0

a[p]\*( maxsum(P,R)-minsum(L-1,P-1) )

最小值小于0

a[p]\*( minsum(P,R)-maxsum(L-1,P-1) )

这道题还可以用单调栈来做

#include<cstdio>

#include<algorithm>

#include<iostream>

using namespace std**;**

typedef long long ll**;**

const int mod**=**1e9**+**7**;**

const int maxn**=**5e5**+**5**;**

int n**,**a**[**maxn**];**

ll sum**[**maxn**],**minsum**[**maxn**][**21**],**maxsum**[**maxn**][**21**];**

ll res**=**0**;**

int len**[**maxn**];**

int minv**[**maxn**][**21**],**minp**[**maxn**][**21**];**

void init\_rmq**(**int n**)** **{**

**for** **(**int i**=**2**;**i**<=**n**;++**i**)** len**[**i**]=**len**[**i**-**1**]+!(**i**&**i**-**1**);**

**for** **(**int i**=**1**;**i**<=**n**;++**i**)** minv**[**i**][**0**]=**a**[**i**],**minp**[**i**][**0**]=**i**;**

**for** **(**int i**=**1**;**i**<=**n**;++**i**)** maxsum**[**i**][**0**]=**minsum**[**i**][**0**]=**sum**[**i**];**

**for** **(**int j**=**1**;**j**<=**len**[**n**];++**j**)**

**for** **(**int i**=**1**;**i**+(**1**<<**j**)-**1**<=**n**;++**i**)** **{**

minv**[**i**][**j**]=**min**(**minv**[**i**][**j**-**1**],**minv**[**i**+(**1**<<(**j**-**1**))][**j**-**1**]);**

minp**[**i**][**j**]=**minv**[**i**][**j**]==**minv**[**i**][**j**-**1**]?**

minp**[**i**][**j**-**1**]:**minp**[**i**+(**1**<<(**j**-**1**))][**j**-**1**];**

maxsum**[**i**][**j**]=**max**(**maxsum**[**i**][**j**-**1**],**maxsum**[**i**+(**1**<<(**j**-**1**))][**j**-**1**]);**

minsum**[**i**][**j**]=**min**(**minsum**[**i**][**j**-**1**],**minsum**[**i**+(**1**<<(**j**-**1**))][**j**-**1**]);**

**}**

**}**

ll rmq\_maxsum**(**int x**,**int y**)** **{**

int k**=**len**[**y**-**x**+**1**];** y**-=(**1**<<**k**)-**1**;**

**return** max**(**maxsum**[**x**][**k**],**maxsum**[**y**][**k**]);**

**}**

ll rmq\_minsum**(**int x**,**int y**)** **{**

int k**=**len**[**y**-**x**+**1**];** y**-=(**1**<<**k**)-**1**;**

**return** min**(**minsum**[**x**][**k**],**minsum**[**y**][**k**]);**

**}**

int rmq\_minp**(**int x**,**int y**)** **{**

int k**=**len**[**y**-**x**+**1**];** y**-=(**1**<<**k**)-**1**;**

**return** minv**[**x**][**k**]<**minv**[**y**][**k**]?**minp**[**x**][**k**]:**minp**[**y**][**k**];**

**}**

void query**(**int l**,**int r**)** **{**

**if** **(**l**>**r**)** **return;**

int p**=**rmq\_minp**(**l**,**r**);**

int v**=**a**[**p**];**

**if** **(**v**<**0**)** **{**

ll sum1**=**rmq\_maxsum**(**l**-**1**,**p**-**1**);**

ll sum2**=**rmq\_minsum**(**p**,**r**);**

res**=**max**(**res**,(**sum2**-**sum1**)\***v**);**

**}**

**else** **{**

ll sum1**=**rmq\_minsum**(**l**-**1**,**p**-**1**);**

ll sum2**=**rmq\_maxsum**(**p**,**r**);**

res**=**max**(**res**,(**sum2**-**sum1**)\***v**);**

**}**

query**(**l**,**p**-**1**);**

query**(**p**+**1**,**r**);**

**}**

int main**()** **{**

// freopen("in.txt","r",stdin);

scanf**(**"%d"**,&**n**);** n**++;**

**for** **(**int i**=**2**;**i**<=**n**;**i**++)** scanf**(**"%d"**,&**a**[**i**]),**sum**[**i**]=**sum**[**i**-**1**]+**a**[**i**];**

init\_rmq**(**n**);**

query**(**2**,**n**);**

printf**(**"%lld\n"**,**res**);**

**return** 0**;**

**}**

## Distance on the tree

需要用到DFS序，LCA最近公共祖先，线段树维护边权和

DFS序列含义：

A

/ \

B E

/ \

C D

对于这棵树，我们DFS搜出来他的访问序列 ABCDE

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | A | B | C | D | E |
| 起始 | 1 | 2 | 3 | 4 | 5 |
| 结束 | 5 | 4 | 3 | 4 | 5 |

这样每个点的起始到结束就是他的子树的节点

DFS序列还有其他类型的，这里不过多介绍。

我们需要维护的是每个点到根节点的边权和(边权处理后只有0、1)，代表有效边数量

我们先把每条边的边权赋值为0，然后把 边和查询 按权值一起排序

处理排序过的 边和查询

如果是边，把这条边的权改为1，这样这条边下的 每个点到根节点边权和都加1

如果是查询，查询**两点到根节点的边权和**减去他们2倍 **最近公共祖先到根节点的边权和**

因为按权排序，所以对于查询，树上有权的边都是 权值小于查询的

#include<cstdio>

#include<algorithm>

using namespace std**;**

//区间更新

#define lson rt<<1,l,m

#define rson rt<<1|1,m+1,r

const int maxn **=** 1e5**+**5**;**

typedef long long ll**;**

ll sum**[**maxn**<<**2**],**lz**[**maxn**<<**2**];**

inline void push\_up**(**int rt**)** **{**

sum**[**rt**]=**sum**[**rt**<<**1**]+**sum**[**rt**<<**1**|**1**];**

**}**

inline void push\_down**(**int rt**,**int l**,**int r**)** **{**

**if** **(**lz**[**rt**])** **{**

int m**=(**l**+**r**)>>**1**;**

sum**[**rt**<<**1**]+=(**m**-**l**+**1**)\***lz**[**rt**];**

sum**[**rt**<<**1**|**1**]+=(**r**-**m**)\***lz**[**rt**];**

lz**[**rt**<<**1**]+=**lz**[**rt**];**

lz**[**rt**<<**1**|**1**]+=**lz**[**rt**];**

lz**[**rt**]=**0**;**

**}**

**}**

void update**(**int rt**,**int l**,**int r**,**int L**,**int R**,**int v**)** **{**

**if** **(**L**<=**l**&&**r**<=**R**)** **{** sum**[**rt**]+=(**r**-**l**+**1**)\***v**;** lz**[**rt**]+=**v**;** **return;** **}**

push\_down**(**rt**,**l**,**r**);**

int m**=(**l**+**r**)>>**1**;**

**if** **(**L**<=**m**)** update**(**lson**,**L**,**R**,**v**);**

**if** **(**m**<**R**)** update**(**rson**,**L**,**R**,**v**);**

push\_up**(**rt**);**

**}**

inline int query**(**int rt**,**int l**,**int r**,**int p**)** **{**

**if** **(**l**==**r**)** **return** sum**[**rt**];**

push\_down**(**rt**,**l**,**r**);**

int m**=**l**+**r**>>**1**;**

**if** **(**p**<=**m**)** **return** query**(**lson**,**p**);**

**return** query**(**rson**,**p**);**

**}**

int dfs**[**maxn**],**dq**,**tot**,**tbt**,**L**[**maxn**],**R**[**maxn**],**l**[**maxn**],**r**[**maxn**],**f**[**maxn**][**25**],**cen**[**maxn**],**ans**[**maxn**];**

struct Bian**{**

int u**,**v**;**

**}**bian**[**maxn**<<**1**];**

struct CZ**{**

int a**,**u**,**v**,**q**,**name**;**

**}**cz**[**maxn**<<**1**];**

void Dfs**(**int x**)**

**{**

dfs**[**x**]=++**dq**;**

L**[**x**]=**dq**;**

**for(**int i**=**l**[**x**];**i**<=**r**[**x**];**i**++)**

**if(**bian**[**i**].**v**!=**f**[**x**][**0**])**

**{**

f**[**bian**[**i**].**v**][**0**]=**x**;**

cen**[**bian**[**i**].**v**]=**cen**[**x**]+**1**;**

Dfs**(**bian**[**i**].**v**);**

**}**

R**[**x**]=**dq**;**

// printf("DFS -> %d %d %d %d %d\n",x,dfs[x],L[x],R[x],cen[x]);

**}**

int lca**(**int x**,**int y**)**

**{**

**if(**cen**[**x**]>**cen**[**y**])**

**{**

**for(**int j**=**20**;**j**>=**0**;**j**--)**

**if(**f**[**x**][**j**]>**0**)**

**{**

**if(**cen**[**f**[**x**][**j**]]>=**cen**[**y**])**

x**=**f**[**x**][**j**];**

**}**

**}**

**if(**cen**[**y**]>**cen**[**x**])**

**{**

**for(**int j**=**20**;**j**>=**0**;**j**--)**

**if(**f**[**y**][**j**]>**0**)**

**{**

**if(**cen**[**f**[**y**][**j**]]>=**cen**[**x**])**

y**=**f**[**y**][**j**];**

**}**

**}**

**if(**x**!=**y**)**

**{**

**for(**int j**=**20**;**j**>=**0**;**j**--)**

**if(**f**[**x**][**j**]>**0**&&**f**[**y**][**j**]>**0**)**

**{**

**if(**f**[**x**][**j**]!=**f**[**y**][**j**])**

x**=**f**[**x**][**j**],**y**=**f**[**y**][**j**];**

**}**

**}**

// printf("%d %d\n",x,y);

**while(**x**!=**y**)**x**=**f**[**x**][**0**],**y**=**f**[**y**][**0**];**

**return** x**;**

**}**

void add**(**int x**,**int y**)**

**{**

tbt**++;**

bian**[**tbt**].**u**=**x**;**

bian**[**tbt**].**v**=**y**;**

**}**

int comb**(**Bian x**,**Bian y**)**

**{**

**return** x**.**u**<**y**.**u**;**

**}**

int comc**(**CZ x**,**CZ y**)**

**{**

**if(**x**.**q**<**y**.**q**)**

**return** 1**;**

**if(**x**.**q**>**y**.**q**)**

**return** 0**;**

**return** x**.**a**>**y**.**a**;**

**}**

int main**()**

**{**

int n**,**m**,**u**,**v**,**lon**;**

scanf**(**"%d%d"**,&**n**,&**m**);**

**for(**int i**=**1**;**i**<**n**;**i**++)**

**{**

scanf**(**"%d%d%d"**,&**u**,&**v**,&**lon**);**

cz**[++**tot**].**a**=**1**;**cz**[**tot**].**u**=**u**;**

cz**[**tot**].**v**=**v**;**cz**[**tot**].**q**=**lon**;**

add**(**u**,**v**);**

add**(**v**,**u**);**

**}**

sort**(**bian**+**1**,**bian**+**tbt**+**1**,**comb**);**

// for(int i=1;i<=tbt;i++)

// printf("%d %d\n",bian[i].u,bian[i].v);

**for(**int i**=**1**;**i**<=**tbt**;**i**++)**

**{**

**if(**i**==**1**||**bian**[**i**].**u**!=**bian**[**i**-**1**].**u**)**

l**[**bian**[**i**].**u**]=**i**;**

**if(**i**==**tbt**||**bian**[**i**].**u**!=**bian**[**i**+**1**].**u**)**

r**[**bian**[**i**].**u**]=**i**;**

**}**

cen**[**1**]=**0**;**dq**=**0**;**f**[**1**][**0**]=-**1**;**cen**[**0**]=-**1**;**

Dfs**(**1**);**

**for(**int j**=**1**;**j**<=**20**;**j**++)**

**for(**int i**=**1**;**i**<=**n**;**i**++)**

**if(**f**[**i**][**j**-**1**]!=-**1**)**

f**[**i**][**j**]=**f**[**f**[**i**][**j**-**1**]][**j**-**1**];**

**else**

f**[**i**][**j**]=-**1**;**

**for(**int i**=**1**;**i**<=**m**;**i**++)**

**{**

scanf**(**"%d%d%d"**,&**u**,&**v**,&**lon**);**

cz**[++**tot**].**a**=**0**;**cz**[**tot**].**u**=**u**;**

cz**[**tot**].**v**=**v**;**cz**[**tot**].**q**=**lon**;**

cz**[**tot**].**name**=**i**;**

**}**

sort**(**cz**+**1**,**cz**+**tot**+**1**,**comc**);**

**for(**int i**=**1**;**i**<=**tot**;**i**++)**

**{**

// printf("%d %d %d %d %d\n",cz[i].a,cz[i].u,cz[i].v,cz[i].q,cz[i].name);

**if(**cz**[**i**].**a**==**1**)**

**{**

int U**=**cz**[**i**].**u**,**V**=**cz**[**i**].**v**;**

**if(**cen**[**U**]<**cen**[**V**])**

update**(**1**,**1**,**n**,**L**[**V**],**R**[**V**],**1**);**

**else**

update**(**1**,**1**,**n**,**L**[**U**],**R**[**U**],**1**);**

**}**

**else**

**{**

int U**=**cz**[**i**].**u**,**V**=**cz**[**i**].**v**;**

int F**=**lca**(**U**,**V**);**

ans**[**cz**[**i**].**name**]=**query**(**1**,**1**,**n**,**dfs**[**U**])+**query**(**1**,**1**,**n**,**dfs**[**V**])-**2**\***query**(**1**,**1**,**n**,**dfs**[**F**]);**

**}**

**}**

**for(**int i**=**1**;**i**<=**m**;**i**++)**

printf**(**"%d\n"**,**ans**[**i**]);**

**}**

## MORE XOR

1. 暴力打表找规律可解
2. 对于W[l,r],可以看作在[L,R]范围内取两个括号P,Q，然后G[P,Q]在[P,Q]范围内再取俩括号

这样对于点x，他的贡献为L<=P<=x的情况\*x<=Q<=R的情况=C(2,x-L+2)\*C(2,R-X+2)

C(2,x-L+2)\*C(2,R-X+2)是奇数代表需要答案异或a[x],偶数不异或a[x]

对于C(2,n)

n和n-1一定有偶数和奇数，如果偶数是4的倍数，那么C(2,n)是偶数，否则是奇数

所以，x-L+2,x-L+1,R-x+1,R-x+2这四个数中有4的倍数就是偶数，答案不需要异或A[x]

#include<cstdio>

using namespace std**;**

int T**,**n**,**a**[**100005**],**m**,**l**,**r**;**

int main**()**

**{**

scanf**(**"%d"**,&**T**);**

**while(**T**--)**

**{**

scanf**(**"%d"**,&**n**);**

**for(**int i**=**1**;**i**<=**n**;**i**++)**

**{**

scanf**(**"%d"**,&**a**[**i**]);**

**}**

**for(**int i**=**4**;**i**<=**n**;**i**++)**

a**[**i**]^=**a**[**i**-**4**];**

scanf**(**"%d"**,&**m**);**

**while(**m**--)**

**{**

scanf**(**"%d%d"**,&**l**,&**r**);**

**if(**l**%**4**==**r**%**4**)**

**{**

**if(**l**>=**4**)**

printf**(**"%d\n"**,**a**[**r**]^**a**[**l**-**4**]);**

**else**

printf**(**"%d\n"**,**a**[**r**]);**

**}**

**else**

**if(**l**%**4**==(**r**-**1**)%**4**)**

**{**

int ans**=**0**;**

**if(**l**>=**4**)**

ans**^=**a**[**r**-**1**]^**a**[**l**-**4**];**

**else**

ans**^=**a**[**r**-**1**];**

**if(**l**+**1**>=**4**)**

ans**^=**a**[**r**]^**a**[**l**-**3**];**

**else**

ans**^=**a**[**r**];**

printf**(**"%d\n"**,**ans**);**

**}**

**else**

**if((**l**+**1**)%**4**==(**r**-**1**)%**4**)**

**{**

**if(**l**+**1**>=**4**)**

printf**(**"%d\n"**,**a**[**r**-**1**]^**a**[**l**-**3**]);**

**else**

printf**(**"%d\n"**,**a**[**r**-**1**]);**

**}**

**else**

printf**(**"0\n"**);**

**}**

**}**

**}**

## Subsequence

#include<cstdio>

#include<cstring>

#include<algorithm>

typedef long long ll**;**

using namespace std**;**

const int maxn**=**1e5**+**5**;**

int a**[**maxn**][**26**],**C**[**26**],**tot**,**cun**[**26**],**n**;**

char c**;**

int main**()** **{**

// freopen("in.txt","r",stdin);

tot**=**0**;**

c**=**getchar**();**

**while(**c**<=**'z'**&&**c**>=**'a'**)**

**{**

C**[++**tot**]=**c**-**'a'**;**

c**=**getchar**();**

**}**

**for(**int i**=**0**;**i**<**26**;**i**++)**

cun**[**i**]=**tot**+**1**;**

**for(**int i**=**tot**;**i**>=**1**;**i**--)**

**{**

**for(**int j**=**0**;**j**<**26**;**j**++)**

a**[**i**][**j**]=**cun**[**j**];**

cun**[**C**[**i**]]=**i**;**

**}**

**for(**int j**=**0**;**j**<**26**;**j**++)**

a**[**0**][**j**]=**cun**[**j**],**a**[**tot**+**1**][**j**]=**tot**+**1**;**

scanf**(**"%d"**,&**n**);**

**while(**n**--)**

**{**

c**=**getchar**();**

**while(**c**<**'a'**||**c**>**'z'**)**c**=**getchar**();**

int mu**=**0**;**

**while(**c**>=**'a'**&&**c**<=**'z'**)**

**{**

mu**=**a**[**mu**][**c**-**'a'**];**

c**=**getchar**();**

**}**

**if(**mu**<=**tot**)**

printf**(**"YES\n"**);**

**else**

printf**(**"NO\n"**);**

**}**

**return** 0**;**

**}**