

Codeforces Round #651 (Div. 2)

A. FashionabLee

正 n 边形能有两边成九十度角的条件是被四整除。

```
#include<bits/stdc++.h>
using namespace std;
int main()
{
    int t;
    scanf("%d",&t);
    while(t--)
    {
        int n;
        scanf("%d",&n);
        if(n%4)puts("NO");
        else puts("YES");
    }
    return 0;
}
```

B. AccurateLee

给一个 01 串， 10 可以变成 1 或 0 ，可以变成的最短串中字典序最小的一个？

最前面的 0 原样输出，最后面的 1 原样输出，中间如果有的话变作 1 个 0

```
#include<bits/stdc++.h>
using namespace std;
char s[100005],ss[100005];
int n;
int main()
{
    int t;
    scanf("%d",&t);
    while(t--)
    {
        scanf("%d%s",&n,s);
        int i,j;
        for(i=0;s[i]=='0';i++);
        for(j=0;s[n-j-1]=='1';j++);
        for(int k=0;k<i;k++)putchar('0');
    }
}
```

```
        if(i+j!=n)putchar('0');
        for(int k=0;k<j;k++)putchar('1');
        putchar('\n');
    }
    return 0;
}
```

C. RationalLee

\$n\$ 个数字分配给 \$k\$ 个朋友，\$w_i\$ 指定了第 \$i\$ 个朋友得到的个数，每个朋友得到的幸福值是最大最小数之和，问总的最大幸福值。

每个朋友先来一个最大值，分配前 \$k\$ 大数字，优先 \$w_i=1\$ 的。之后按 \$w_i\$ 从多到少的顺序给朋友分配 \$w_i-1\$ 个小的数字

```
#include<bits/stdc++.h>
#define LL long long
using namespace std;
int n,k,w[200005];
LL a[200005];
bool cmp(LL x,LL y){return x>y;}
int main()
{
    int t;
    scanf("%d",&t);
    while(t--)
    {
        scanf("%d%d",&n,&k);
        for(int i=1;i<=n;i++)scanf("%lld",&a[i]);
        sort(a+1,a+1+n,cmp);
        for(int i=1;i<=k;i++)scanf("%d",&w[i]);
        sort(w+1,w+1+k);
        int q=n;LL res=0;
        for(int i=1;i<=k;i++)
        {
            res+=a[i];
            if(w[i]==1)res+=a[i];
        }
        for(int i=k;i;i--)
        {
            if(w[i]>1)res+=a[q],q-=w[i]-1;
            else break;
        }
        printf("%lld\n",res);
    }
}
```

```
    return 0;
}
```

D. TediousLee

观察一下发现这不那什么分形图嘛，中间挂 $n-1$ level左右 $n-2$ level可以记录一下顶端的点有没有用到，也可以直接用模 10^9+7 规律。

```
#include<bits/stdc++.h>
#define LL long long
#define N 2000005
#define Mod 1000000007
LL f[N];
bool top[N];
using namespace std;
int main()
{
    f[3]=4;top[3]=1;
    for(int i=4;i<N;i++)
    {
        f[i]=(f[i-1]+f[i-2]*2%Mod)%Mod;
        if(!top[i-1]&&!top[i-2])
            top[i]=1,f[i]=(f[i]+4)%Mod;
    }
    int t;
    scanf("%d",&t);
    while(t--)
    {
        int n;
        scanf("%d",&n);
        printf("%lld\n",f[n]);
    }
    return 0;
}
```

E. DeadLee

n 个朋友 m 盘菜，每个朋友会喜欢两盘菜，第 i 盘有 w_i 份，朋友到达后会试图去吃两盘菜各一份，如果两盘菜都没吃到你就死了。问能不能不死以及如何安排朋友到达顺序。

贪心，不断把供大于求的菜入队。处理到某个菜意味着喜欢这盘菜的朋友就算安排在最后也必然能满足，

把朋友加入答案序列（如果还没的话），然后朋友喜欢的另一个菜就可以需求-。最后答案序列逆序输出。

```
#include<bits/stdc++.h>
#define pb push_back
const int N=1e5+5,M=2e5+5;
using namespace std;
int n,m,w[N],d[N],x[M],y[M],vis[N],done[M];
vector<int>g[N],res;
queue<int>q;
int main()
{
    scanf("%d%d",&n,&m);
    for(int i=1;i<=n;i++)scanf("%d",&w[i]);
    for(int i=1;i<=m;i++)
    {
        scanf("%d%d",&x[i],&y[i]);
        d[x[i]]++,d[y[i]]++;
        g[x[i]].pb(i),g[y[i]].pb(i);
    }
    for(int i=1;i<=n;i++)if(w[i]>=d[i])q.push(i),vis[i]=1;
    while(!q.empty())
    {
        int u=q.front();q.pop();
        for(int i:g[u])
        {
            int v=x[i]+y[i]-u;
            if(!vis[v])
            {
                d[v]--;
                if(d[v]<=w[v])q.push(v),vis[v]=1;
            }
            if(!done[i])done[i]=1,res.pb(i);
        }
    }
    if(res.size()!=m)puts("DEAD");
    else
    {
        puts("ALIVE");
        reverse(res.begin(),res.end());
        for(int i:res)printf("%d ",i);
    }
    return 0;
}
```

F. BareLee

进行 t 回合游戏。第 i 回合初始数字 s_i 二人轮流操作给数字 $\times 2$ 或 $+1$ ，操作后严格大于 e_i 者输，本回合游戏结束，输的人做下一局先手。问，作为第一局先手，能否独立于对方操作赢/输。

$f(i,j)$ 都想赢（想操作后到达 $\text{exactly} \sim j$ 的情况下，先手能否必赢。

- j 为奇数 $\rightarrow i$ 奇败偶胜（能到达必败态的状态才必胜，奇数 $+1$ 达偶数， $\times 2$ 达偶数，可归纳证明）。
- j 为偶数。
- $i > j/2$ 奇胜偶败（所有 $\times 2$ 行为都马上死，只能 $+1$ ）
- $j/2 \geq i > j/4$ $\times 2$ 得上分类中“偶败”态，全必胜。
- $j/4 \geq i$ 一旦操作后到达上一分类则对方必胜，故相当于 $f(i,j/4)$

$g(i,j)$ 都想输（想操作后大于 j 的情况下，先手能否必输。

- $i > j/2$ $\times 2$ 即可，必败。
- $j/2 \geq i$ 一旦操作后到达上一分类则对方必败，故相当于 $f(i,j/2)$

对于每回合记录一下能否作为先手/后手。

```
#include<bits/stdc++.h>
#define ll long long
using namespace std;
int f(ll i,ll j)
{
    if(j&1)return (i&1)==0;
    if(i>j/2)return i&1;
    if(i>j/4)return 1;
    return f(i,j/4);
}
int g(ll i,ll j)
{
    if(i>j/2)return 1;
    return f(i,j/2);
}
const int N=1e5+10;
ll s[N],e[N];
int main()
{
    int t,fi=1,se=0,win,lose;
    scanf("%d",&t);
    for(int i=1;i<=t;i++)
    {
        win=lose=0;
        scanf("%lld%lld",&s[i],&e[i]);
        if(fi)win|=f(s[i],e[i]),lose|=g(s[i],e[i]);
        if(se)win|=(f(s[i],e[i])^1),lose|=(g(s[i],e[i])^1);
        se=win,fi=lose;
    }
}
```

```
}  
printf("%d %d\n",win,lose);  
return 0;  
}
```

From:
<https://wiki.cvbbacm.com/> - CVBB ACM Team

Permanent link:
https://wiki.cvbbacm.com/doku.php?id=2020-2021:teams:wangzai_milk:codeforces_round_652_div_2_zars19 

Last update: **2020/07/16 17:58**