

Codeforces Global Round 8

A. C+=

每次操作可令'a+=b'或'b+=a'问最少多少次可以使得\$a\$或\$b\$严格大于\$n\$

题解：每次把较大的一个加在较小的一个上就好。

```
#include<iostream>
#include<cstdio>
#include<cstring>
#include<cstring>
#define LL long long
using namespace std;
int main()
{
    int q,a,b,n;
    scanf("%d",&q);
    while(q--)
    {
        scanf("%d%d%d",&a,&b,&n);
        if(a>b)swap(a,b);
        int cnt=0;
        while(a<=n&&b<=n)
        {
            if(cnt%2)b+=a;
            else a+=b;
            cnt++;
        }
        printf("%d\n",cnt);
    }
    return 0;
}
```

B. Codeforces Subsequences

给出一个最短的字符串使得至少有\$k\$个`codeforces`子序列。

题解：一开始没看到至少以为是exactly好愚蠢，，随便想象一下最后的形态应该是每个字母重复若干次的样子（比如 $\text{ccccooodeeffoorrcceess}$ ）只有不同位置的字母重复次数相差不超过1才是最优，所以对每个位置字母的个数轮流加一直到符合即可

```
#include<iostream>
```

```
#include<cstdio>
#include<cstring>
#include<cstring>
#define LL long long
using namespace std;
char s[15]="codeforces";
LL num[10];
int main()
{
    LL k;
    scanf("%lld",&k);
    for(int i=0;i<10;i++)num[i]=1;
    LL kk=1,i=0;
    while(kk<k)
    {
        kk=kk/num[i]*(num[i]+1);
        num[i]++;
        i++;
        if(i==10)i=0;
    }
    for(int i=0;i<10;i++)
    while(num[i]--)putchar(s[i]);
    return 0;
}
```

C. Even Picture

格子染色，要使得恰有 n 个灰色格相邻所有（上下左右）格子也是灰的，而所有灰色格子必相邻偶数（明显 $\rightarrow 2$ 个）灰色格子。

题解：看图的话立刻就懂了，但是我想了半天。写的是第二种，代码有点乐色不想放了。



D. AND, OR and square sum

给出大小为 n 的数组 a 每次选择 i, j 若 $a_i = x, a_j = y$ 则操作过后 $a_i = a_i \& y, a_j = a_i \mid y$ 问操作任意次数后能得到的最大的数组的平方和。

题解：这个操作前后 $a_i + a_j$ 不变，但是可以使得两数中大的更大，小的更小，于是增大了平方和，故要尽可能的做。最终的形态应该是对于每一位来说 1 个数不变且都集中在了最大的一边。

```

#include<iostream>
#include<cstdio>
#include<cstring>
#include<algorithm>
#include<vector>
#define ll long long
using namespace std;
#define pii pair<int,int>
#define mp make_pair
#define pb push_back
#define fi first
#define se second
int a[200005],b[25];
int main()
{
    int n;ll res=0;
    scanf("%d",&n);
    for(int i=1;i<=n;i++)
    {
        scanf("%d",&a[i]);
        int j=0;
        while(a[i])b[j]+=(a[i]&1),a[i]>>=1,j++;
    }
    for(int i=1;i<=n;i++)
    {
        ll t=0;
        for(int j=0,k=1;j<=20;j++,k<=1)
            if(b[j])t|=k,b[j]--;
        res+=t*t;
    }
    printf("%lld\n",res);
    return 0;
}

```

E. Ski Accidents

给出 n 个点的有向无环图，每个点的出度最多为 2，问最多删去 $\frac{4}{7}n$ 个点，使得图里没有包含两条边的路径的方案。

题解：节点可以分为三类。

- V_0 : 没有来自 V_0 和 V_1 的入度的点集。
- V_1 : 有来自 V_0 的入度，没有来自 V_1 的入度。
- V_2 : 有来自 V_1 的入度。

显然删去 V_2 后满足要求。由

于 $|V_0| \geq |V_1|/2 \geq |V_2|/4$ 有 $n = |V_0| + |V_1| + |V_2| \geq |V_2|(1 + 1/2 + 1/4) = 7|V_2|/4$

```
#include<bits/stdc++.h>
#define ll long long
using namespace std;
#define pii pair<int,int>
#define mp make_pair
#define pb push_back
#define fi first
#define se second
const int N=2e5+6;
vector<int>G[N],ans;
int d[N];
int main()
{
    int t;
    scanf("%d",&t);
    while(t--)
    {
        int n,m;
        scanf("%d%d",&n,&m);
        ans.clear();
        for(int i=1;i<=n;i++)G[i].clear(),d[i]=0;
        for(int i=1;i<=m;i++)
        {
            int x,y;
            scanf("%d%d",&x,&y);
            G[y].pb(x);
        }
        for(int i=1;i<=n;i++)
        {
            for(int j:G[i])d[i]=max(d[i],d[j]+1);
            if(d[i]==2)ans.pb(i),d[i]=-1;
        }
        printf("%d\n",ans.size());
        for(int i:ans)printf("%d ",i);
        puts("");
    }
    return 0;
}
```

F. Lamps on a Circle

交互□ n 个灯围成一圈，初始全灭，每回合你可以打开若干盏灯（也可以选择结束游戏），而对方可以使使得连续的、相同数量的一段中的灯全部灭掉。要使得结束游戏时打开的灯的数量是双方采用最优策略情况下的最多。

题解：梦幻贪心。想象一下，最后一个回合中你会打开 k 盏灯，之后对方关掉连续长度为 k 的一段，而这个过程开灯数是增加了的。所以，打开 k 盏灯后的形态应该是“不存在长度为 k 的连续一段灯全部打开”，所以至多连续 $k-1$ 盏灯打开。

灯最多的情形将会是以 $k-1$ 亮 1 灭为循环（但第 n 盏必灭）。比如 $n=17$ 的情况下

□ $k=3$ □ $\text{mathsf{11011011011011010}}$ □在对方关完后变为 $\text{mathsf{00011011011011010}}$ □一种可能）即最优解。故可以先枚举 k 算出 $R(n)=\max_k \left(n - k \cdot \left\lceil \frac{n}{k} \right\rceil + 1 \right)$ □

交互过程中每次点亮补全至“被以 1 灭隔开的若干 $k-1$ 亮段”□ $\text{mathsf{11011011011011010}}$ □的相同局面即可，直至到达最优 $R(n)$ □

```
#include<bits/stdc++.h>
using namespace std;
#define pii pair<int,int>
#define mp make_pair
#define pb push_back
#define fi first
#define se second
#define ll long long
int n,r,k,a[1005],t,x;
int main()
{
    scanf("%d",&n);
    for(int i=1;i<=n;i++)
        if(n-n/i-(n%i!=0)-i+1>r)r=n-n/i-(n%i!=0)-i+1,k=i;
    while(t<r)
    {
        printf("%d ",r+k-1-t);
        for(int i=1;i<=n;i++)
            if(i%k&&!a[i])a[i]=1,printf("%d ",i);
        puts(""),fflush(stdout);
        scanf("%d",&x);
        int g=0;
        for(int i=0,j=x;i<r+k-1-t;i++,j=(j+1>n)?1:j+1)
            g+=a[j],a[j]=0;
        t=r+k-1-g;
    }
    puts("0");
    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_global_round_8 

Last update: **2020/06/30 23:56**