

2017-2018 ACM-ICPC, NEERC, Moscow Subregional Contest

比赛情况

题号	A	B	C	D	E	F	G	H	I	J
状态	O	-	O	O	-	O	O	O	O	-

O 在比赛中通过 Ø 赛后通过! 尝试了但是失败了 - 没有尝试

比赛时间

2020-06-14 13:00-18:00

提交记录

A: Wrong Answer 2020-06-14 13:47 *3

A: Accepted 2020-06-14 13:49

D: Accepted 2020-06-14 14:00

H: Wrong Answer 2020-06-14 15:19 *3

H: Accepted 2020-06-14 15:28

C: Accepted 2020-06-14 15:42

G: Accepted 2020-06-14 16:16

F: Wrong Answer 2020-06-14 17:46 *7

F: Accepted 2020-06-14 17:58

题解

A. Advertising Strategy

目标订购者为 n 总花费 k a_i 为当前订购者数量，每天可以花费 x_i 使得 $b_i = a_i + x_i$ 之后 $a_{i+1} = b_i + \min(b_i, \lfloor \frac{n-b_i}{2} \rfloor)$ 问达到目标订购的最少天数

题解：显然至少要有花费要在最后一天给出，其他钱应该是早花早受益，于是在第一天和最后一天花钱。枚举在第一天花的钱，模拟天数增加直到剩下的人可以一次性花钱补全。

code:

```
#include<bits/stdc++.h>
```

```
#define ll long long
#define pii_pair<int,int>
#define mp_make_pair
#define pb push_back
#define fi first
#define se second
#define rep(i,a,b) for(int i=(a);i<=(b);i++)
#define per(i,a,b) for(int i=(a);i>=(b);i--)
#define show1(a) cout<<#a<<" = "<<a<<endl
#define show2(a,b) cout<<#a<<" = "<<a<<"; "<<#b<<" = "<<b<<endl
using namespace std;
const ll INF = 1LL<<60;
const int inf = 1<<30;
const int maxn = 2e5+5;
inline void fastio() {ios::sync_with_stdio(false);cin.tie(0);cout.tie(0);}

ll n,k;

int main()
{
    fastio();
    cin>>n>>k;
    ll res=INF;
    for(ll i=1;i<k;i++)
    {
        ll now = k-i,ans = 1;
        while(now<n-i){
            now = now + min(now,(n-now)/2);
            ans++;
        }
        res = min(res,ans);
    }
    cout<<res<<endl;
    return 0;
}
```

C.Carpet

要将 $n(1 \leq n \leq 100000)$ 个点的树铺在 1000000×20 的坐标平面里，使得边与边无端点之外的重合点。

题解：树剖，将轻儿子往上放，重儿子往右放。

code:

```
#include<bits/stdc++.h>
using namespace std;
```

```
const int N=1e5+5;
int n,head[N],cnt,sz[N];
int x[N],y[N],tot[25];
struct Node
{
    int nxt,to;
}Edges[N*2];
void addedge(int u,int v)
{
    Edges[++cnt].nxt=head[u];
    head[u]=cnt;
    Edges[cnt].to=v;
}
void dfs1(int u,int f)
{
    sz[u]=1;
    for(int i=head[u];~i;i=Edges[i].nxt)
    {
        int v=Edges[i].to;
        if(v==f)continue;
        dfs1(v,u),sz[u]+=sz[v];
    }
}
void dfs2(int u,int f,int d)
{
    y[u]=d,x[u]=++tot[d];
    int maxv=0,k=0;
    for(int i=head[u];~i;i=Edges[i].nxt)
    {
        int v=Edges[i].to;
        if(v==f)continue;
        if(sz[v]>maxv)maxv=sz[v],k=v;
    }
    for(int i=head[u];~i;i=Edges[i].nxt)
    {
        int v=Edges[i].to;
        if(v==f||v==k)continue;
        dfs2(v,u,d+1);
    }
    if(k)dfs2(k,u,d);
}
int main()
{
    memset(head,-1,sizeof(head));
    scanf("%d",&n);
    for(int i=1;i<n;i++)
    {
        int u,v;
        scanf("%d%d",&u,&v);
        addedge(u,v),addedge(v,u);
    }
}
```

```
dfs1(1,0),dfs2(1,0,1);  
for(int i=1;i<=n;i++)printf("%d %d\n",x[i],y[i]);  
return 0;  
}
```

D. Decoding of Varints

题意

语文题，就是给了一个类似128进制的定义，然后给一个未知的序列，对于这个序列中的每一个数字，如果这个数字 ≥ 0 的时候，这个数就变为这个数字的二倍，奇遇的时候是这个数字相反数的二倍再减一，把新数字搞成他之前定义的形式中的每一位，现在给你每一位，让你装换回去。

数据范围

$n \leq 10000$

题解

就 模拟啊

代码

```
#include <stdio.h>  
#include <string.h>  
#include <stdlib.h>  
#include <algorithm>  
using namespace std;  
const int N = 1e4+5;  
int a[N];  
typedef unsigned long long ll;  
int main()  
{  
    int n;  
    scanf("%d",&n);  
    for (int i = 1;i<= n;i++)scanf("%d",&a[i]);  
    for (int i = 1;i<= n;i++) {  
        int y = i;  
        ll ans = 0,x = 1;  
        while (y <= n) {  
            if (a[y]>=128) {  
                ans += (a[y]-128)*x;  
                x *= 128;  
                y++;  
            } else {  
                ans -= (a[y]+128)*x;  
                x *= 128;  
                y++;  
            }  
        }  
        printf("%d ",ans);  
    }  
}
```

```

        ans += a[y]*x;
        i = y;
        break;
    }
}
if (ans & 1)
    printf("%lld\n", (long long)0 - ((ans - 1) / 2 + 1));
else
    printf("%llu\n", ans / 2);
}
return 0;
}

```

H. Hilarious Cooking

题意

希望你构造一个序列，序列相邻两个数差值不超过1，和为 T 。然后其中有一些给定位置的给定数字，问能否构造成功。

数据范围

序列的长度 $n \leq 2 \times 10^9$ ， $1 \leq T \leq 10^{18}$

给定的数字个数 $1 \leq m \leq 100000$

题解

可以考虑，构造这样一个序列可以保证在一个范围内连续，因为除非在最大值和最小值的位置，其他情况下都可以找到一个位置+1或者找到一个位置-1并符合序列规定，那么我们要做的就是确定这个序列可能的最大值和可能的最小值。注意到其限制其实就是给定的数字，最大值在两个给定的数字构造一个像山顶或山坡的形状，最小值就构造一个山谷或山坡一样的形状即可，要注意一些细节，比如最开始和最后的。

代码

```

#include <stdio.h>
#include <string.h>
#include <stdlib.h>
#include <algorithm>
using namespace std;
typedef long long ll;
const int N = 1e5 + 5;
ll x[N], y[N];
ll T, n;

```

```
ll calc(ll x,ll y) {
    if (x > y || y < 0) return 0;
    ll nx = x < 0 ? -x : 0;
    return (x+y)*(y-x+1)/2 + (nx+1)*nx/2;
}
int main()
{
    scanf("%lld",&T);
    ll minn = 0;
    ll maxn = 0;
    int m;
    scanf("%lld%d",&n,&m);
    for (int i = 1;i<= m;i++) {
        scanf("%lld%lld",&x[i],&y[i]);
        minn += y[i];
        maxn += y[i];
    }
    for (int i = 1;i < m;i++) {
        ll dis = x[i+1]-x[i];
        ll val = abs(y[i+1]-y[i]);
        if (val > dis) {
            printf("No\n");
            return 0;
        }
        if (dis == 1) { continue; }
        ll maxx = max(y[i+1],y[i]) + (dis-val)/2;
        ll minx = min(y[i+1],y[i]) - (dis-val)/2;
        if (val % 2 == dis % 2) {
            maxn += calc(max(y[i+1],y[i]),maxx) +
            calc(min(y[i+1],y[i]),maxx) - max(maxx,0ll);
            minn += calc(minx,max(y[i+1],y[i])) +
            calc(minx,min(y[i+1],y[i])) - max(minx,0ll);
        } else {
            maxn += calc(max(y[i+1],y[i]),maxx) +
            calc(min(y[i+1],y[i]),maxx);
            minn += calc(minx,max(y[i+1],y[i])) +
            calc(minx,min(y[i+1],y[i]));
        }
        maxn -= y[i+1]+y[i];
        minn -= y[i+1]+y[i];
    }
    maxn += calc(y[1]+1,y[1]+x[1]-1);
    minn += calc(y[1]-x[1]+1,y[1]-1);
    maxn += calc(y[m]+1,y[m]+n-x[m]);
    minn += calc(y[m]-n+x[m],y[m]-1);
    if (T<= maxn && T >= minn) { printf("Yes\n"); }
    else { printf("No\n"); }
    return 0;
}
```

```
}
```

replay

比赛总结

From:

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

Permanent link:

https://wiki.cvbbacm.com/doku.php?id=2020-2021:teams:wangzai_milk:20200614%E6%AF%94%E8%B5%9B%E8%AE%B0%E5%BD%95&rev=1592993119 

Last update: 2020/06/24 18:05