Codeforces Round 645(div2)

A

题意：有一块n*m的矩阵，定义方格边界上放一盏灯可以点亮上下或者左右的格子(若放在上下边上，点亮上下两个格子，若左右两边，则左右两个格子)，问要点亮所有各自需要多少盏灯?

题解：不妨枚举几种情况，发现如果总共有n*m个格子，我们一定能找到一种方法，使每一个灯正好点亮两个未点亮的区域，那么答案就显而易见了，为$\lceil\frac{nm}{2}\rceil$□代码略

B

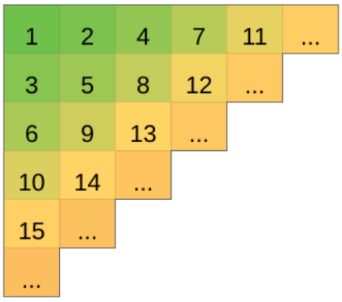
题意：一个场景□A要请一系列人来♂van□每个人都有一个编号，一次可以请许多人，但是要保证每一次请一堆人过来时需要保证，每一个刚来的人的编号要比除了他之外在场的人的数量小。最初A算一个人，问最多能叫多少个人。

题解：很容易可以发现每从序列中选一个人则，总容量会+1，那么叫第k个人的时候，当场会有至少k个人在场，所以考虑最值只要考虑边界条件，我们先把序列进行排序，考虑序列中每一个满足$a_i≤i$的点，只要有该式满足可以保证前面i个数一定满足条件，所以只需要考虑每一个满足$a_i≤i$□的点并且取i的最大值即可。

```
#include <bits/stdc++.h>
using namespace std;
const int maxn=1e5+13;
int a[maxn];
int main()
{
    int t,n,m;
    scanf("%d",&t);
    while(t--)
    {
        int ans=0;
        scanf("%d",&n);
        for(int i=1;i<=n;i++) scanf("%d",&a[i]);
        sort(a+1,a+1+n);
        for(int i=1;i<=n;i++)
        {
            if(a[i]<=i)
            {
                ans=max(ans,i);
            }
        }
        printf("%d\n",ans+1);
    }
 }
```

C

题意：有一个三角型排列为左图所示，定义一次移动只能从(i,j)到$(i+1,j)$或者从(i,j)到$(i,j+1)$问若将该三角形阵列的(i_1,j_1),点走到(i_2,j_2)中途经过的所有点的值全部加在一起，有多少种不同的答案？



题解：~~这题真是坑死我子……~~，可以找规律然后暴力计算个数，因为枚举之后不难发现两点之间路径和最长一定是先向下走，再向上走，最小一定是先向右走再向下走，然后根据等差数列的关系暴力求式子计算。后来发现答案是很简单的$(i_2-i_1)*(j_2-j_1)+1$,~~直接自闭~~，后来想想发现还挺自然(~~知道答案当然自然子~~)，每个数的下边一定比左边大1，加的效果是叠加的，假如这一次比上一次早一步走了下面，那么一定会给整体加上(j_2-j_1)那么一共可以早(i_2-i_1)次。可以得到，最小的和最大的差，然后+1即可。代码略。

D

题意：重新定义月份的数量和每月的天数，规定每月的第i天有一个权值为i给定月数与每个月的天数，问连续的k天的最大权值和为多少。

题解：容易证明若要最大，连续k天末尾必须要是一个月的月末，否则当我们向后移动，一定能找到更加优秀的解。那么我门枚举月份即可。注意先预处理，要不然会tle.....

```
#include <bits/stdc++.h>
using namespace std;
typedef long long ll;
const int maxn=5e5+13;
ll a[maxn];
int n;
ll x;
int main()
{
    scanf("%d%lld",&n,&x);
    for(int i=1;i<=n;i++)
    {
        scanf("%lld",&a[i]);
        a[i+n]=a[i];
     }
    n*=2;
    ll ans=0;
    ll t=0,f=0;
    int l=n/2;
    while(a[l]+t<=x) t+=a[l],f+=a[l]*(a[l]+1)/2,l--;
    for(int i=n/2+1;i<=n;i++)
```

```
    {
        t+=a[i];
        f+=a[i]*(a[i]+1)/2;
        while(t>x) t-=a[l+1],f-=a[l+1]*(a[l+1]+1)/2,l++;
        ll tp=f;
        if(t<x)
        {
            ll res=x-t;
            f+=res*a[l]-res*(res-1)/2;
        }
        ans=max(ans,f);
        f=tp;
    }
    printf("%lld",ans);
}
```

E

题意：给定长度为n的序列，而且这个序列满足后$\lfloor\frac{n}{2}\rfloor$个数全部为一个值$x$,问是否存在一个整数$k$使得所有长度为$k$的子序列和一定为正数。

题解：容易证明若一个小于等于$\lfloor\frac{n}{2}\rfloor$的数$k$满足条件，则$2k$也满足条件。定义$s_i=\sum_{j=i}^{i+k-1}a_j$还有一个明显的递推式为$s_{i+1}=s_i-a_{i}+a_{i+k}$,这样我们可以预处理出一个序列$[s_i,a_{k+1}-a_1,a_{k+2}-a_2,.....a_{n}-a_{n-k}]$,发现我们需要求的$s_{i}$就是该序列的前缀和，由于$k≥\lfloor\frac{n}{2}\rfloor$,则上式可以改写为$[s_i,x-a_1,x-a_2,.....x-a_{n-k}]$,我们处理出这个式子的前缀和，然后找出最小值,这也可以预处理出来，之后为了求s_{i}我们与处理出a的前缀和然后我们枚举k的情况，看是否存在k使所有s都大于0，复杂度$O(n)$

```
#include <bits/stdc++.h>
using namespace std;
typedef long long ll;
const int maxn=6e5+13;
ll a[maxn];
ll m[maxn];
ll pre[maxn];
int main()
{
    ll n,x;
    ll ans=0;
    scanf("%lld",&n);
    for(int i=1;i<=(n+1)/2;i++)
    {
        scanf("%lld",&a[i]);
        pre[i]=pre[i-1]+a[i];
    }
    scanf("%lld",&x);
    m[1]=0;
    ll t=0;
```

```
    for(int i=2;i<=(n+1)/2+1;i++)
    {
        t+=x-a[i-1];
        m[i]=min(m[i-1],t);
    }
    ll test=pre[(n+1)/2];
    for(int k=(n+1)/2;k<=n;k++)
    {
        if(test+m[n-k+1]>0)
        {
            ans=1ll*k;
            break;
         }
        test+=x;
    }
    if(ans==0) printf("-1");
    else printf("%lld",ans);
}
```