我们每次进行游戏，可以令n-1个人还需要进行的轮数-1,也就是所有人的总需要轮数减去n-1轮。

那么我们统计所有人的总需要轮数sum，在最优策略下（每次都选取当前需要轮数最小的那个人当上帝）需要进行的轮数为[sum/(n-1)]=x轮（向上取整）。

但是如果有某一个人的需求次数特别多有y次，则在进行上述x轮后，这个需求次数最多的人会剩下来，并且他必然在前面的x轮中次次都可以作为玩家参与，最少还需要其他人陪他继续y-x轮。这种情况的所需轮数就是y轮，也就是需求次数最多的那个人的轮数。

C语言标程:

#include<stdio.h>

#define ll long long

ll max(ll a,ll b)

{

**return** a>b?a:b;

}

**int** main()

{

**int** n;

    scanf("%d",&n);

    ll sum=0,Max=-1;

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

    {

        ll x;

        scanf("%lld",&x);

        sum+=x;

        Max=max(Max,x);

    }

    printf("%lld\n",max(Max,(sum+n-2)/(n-1)));

    //(sum+n-2)/(n-1)等价于sum/(n-1)+sum%(n-1)?1:0

}

C++标程:

#include<bits/stdc++.h>

#define ll long long

#define IOS ios::sync\_with\_stdio(0); cin.tie(0); cout.tie(0);

//取消cin,cout同步，可以使得cin,cout提速。

//取消同步后在cf上与scanf,printf同速度，其他大部分oj上仍然慢与scanf,printf

**using** **namespace** std;

**int** main()

{

    IOS

**int** n;cin>>n;

    ll sum=0,Max=-1;

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

    {

        ll x;cin>>x;

        sum+=x;

        Max=max(Max,x);

    }

    cout<<max(Max,(sum+n-2)/(n-1))<<endl;

    //(sum+n-2)/(n-1)等价于sum/(n-1)+sum%(n-1)?1:0

}