二分答案题解

数列分段

思路:与砍木头类似，略 根据$check$条件的不同，二分的代码有所区别，这里会给出不修改二分模板的代码。$check$条件是$t<=m$,返回1；如果大家是$t>=m$，返回1，那么你需要修改一下二分模板，自行尝试。

代码：

```
#include<bits/stdc++.h>
using namespace std;
int n,m,a[100010],maxx=0,tot,ans;
bool check(int x) //当最小值是x时是否可以
{
    int sum=0,t=1;
    for(int i=1;i<=n;i++)
    {
        if(sum+a[i]>x)//sum是一直累加，当超过x时月份++，说明这一天自己另一个月份
        {
            sum=a[i];
            t++;
        }
        else
        sum+=a[i];
   }
   if(t<=m) return 1;//月份等于m时正好分成m个月份，小于m时，可以将某些月份中的天数拆开组成新月份，满足分成m个月份
   else return 0;
}
int main()
{
    scanf("%d%d",&n,&m);
    for(int i=1;i<=n;i++)
    {
        scanf("%d",&a[i]);
        tot+=a[i];//二分的右端点是总的开销
        if(a[i]>maxx) maxx=a[i];//二分的左端点是n天中最大的开销
    }
    int l=maxx,r=tot,mid;
    while(l<=r)//二分查找
    {
        mid=(l+r)/2;
        if(check(mid))//如果中间值可以
        {
            ans=mid;//记录答案
            r=mid-1;//缩小范围
        }
        else l=mid+1;//否则扩大范围
    }
    printf("%d",ans);//输出最小值
```

```
}
```

Aggressive Cows

思路: 把C头牛放到N个带有编号的隔间里，使得任意两头牛所在的隔间编号的最小差值最大。例如样例排完序后变成1 2 4 8 9，那么1位置放一头牛，4位置放一头牛，它们的差值为3；最后一头牛放在8或9位置都可以，和4位置的差值分别为4、5，和1位置的差值分别为7和8，不比3小，所以最大的最小值为3。 分析：这是一个最小值最大化的问题。先对隔间编号从小到大排序，则最大距离不会超过两端的两头牛之间的差值，最小值为0。所以我们可以通过二分枚举最小值来求。假设当前的最小值为x，如果判断出最小差值为x时可以放下C头牛，就先让x变大再判断；如果放不下，说明当前的x太大了，就先让x变小然后再进行判断。直到求出一个最大的x就是最终的答案。

本题的关键就在于讨论差值的大小。

代码：

```
#include<bits/stdc++.h>
using namespace std;
const int  N  = 50010;
int a[N];
int n,k;
int check(int r)
{
    int t=a[1]+2*r;
    int num=1;
    for(int i=1;i<=n;i++)
    {
        if(a[i]>t)
        {
            num++;
            if(num>k) return 0;
            t=a[i]+2*r;
        }
    }
    return 1;
}
int main()
{
    scanf("%d%d",&n,&k);
    for(int i=1;i<=n;i++)
    {
        scanf("%d",&a[i]);
    }
    sort(a+1,a+n+1);
    int l=0,r=500000000;
    while(l<=r)
    {

        int mid=(l+r)/2;
    //  cout<<l<<" "<<r<<" "<<mid<<":"<<check(mid)<<endl;
```

```
        if(check(mid)) r=mid-1;
        else l=mid+1;
    }
    printf("%d\n",l);
    return 0;
}
```

Monthly Expense

思路: 经典的二分题目，和数列分段是一样的

代码:

```
#include<bits/stdc++.h>
using namespace std;
int n,m,a[100010],maxx=0,tot,ans;
bool check(int x) //当最小值是x时是否可以
{
    int sum=0,t=1;
    for(int i=1;i<=n;i++)
    {
        if(sum+a[i]>x)//sum是一直累加，当超过x时月份++，说明这一天自己另一个月份
        {
            sum=a[i];
            t++;
        }
        else
        sum+=a[i];
   }
   if(t<=m) return 1;//月份等于m时正好分成m个月份，小于m时，可以将某些月份中的天数拆开组成新月份，满足分成m个月份
   else return 0;
}
int main()
{
    scanf("%d%d",&n,&m);
    for(int i=1;i<=n;i++)
    {
        scanf("%d",&a[i]);
        tot+=a[i];//二分的右端点是总的开销
        if(a[i]>maxx) maxx=a[i];//二分的左端点是n天中最大的开销
    }
    int l=maxx,r=tot,mid;
    while(l<=r)//二分查找
    {
        mid=(l+r)/2;
        if(check(mid))//如果中间值可以
        {
            ans=mid;//记录答案
            r=mid-1;//缩小范围
        }
```

```
        else l=mid+1;//否则扩大范围
    }
    printf("%d",ans);//输出最小值
}
```

进制数

根据题目提示，y在十进制下的大小随y的进制数增大而增大，那么就可以二分来做这个题。二分y是多少进制数，然后判断y在十进制数下与n的大小。

代码：

```
#include<bits/stdc++.h>
using namespace std;
int k,flag,a1,a2;
//n进制转十进制
int jinzhi(char a[],int n)//分别为要转成10进制的数和原进制
{
    long t1;
    int i,t,t3;
    t3=strlen(a);
    t1=0;
    for(i=0;i<t3;i++)
    {
        if(a[i]>='0'&&a[i]<='9')//如果在0~9之间
            t=a[i]-'0';//转换为数字
        else if (n>=11&&(a[i]>='A'+n-10))//如果是其他字符
            t=a[i]-'A'+10;//转换为数字
        t1=t1*n+t;//一位一位加
    }
    return t1;
}
int main(){
    int l=10,r=15000;
    char x[1000],y[1000];
    cin>>k;
    for(int t=1;t<=k;t++){//有多组数据
        cin>>x>>y;
        for(int i=10;i<=15000;i++){//循环枚举出第一个
            a1=jinzhi(x,i);
            l=10,r=15000;
            //用二分算出第二个
            while(l<=r){
                int mid=(l+r)/2;
                a2=jinzhi(y,mid);
                if(a2==a1){//满足条件
                    cout<<i<<" "<<mid<<endl;
                    break;
                }
                else if(a2<a1) l=mid+1;
```

```
                else r=mid-1;
            }
            if(a1==a2) break;
        }
    }
}
```

复制书稿

思路:由题目可以分析出，答案一定是在1 ~ 所有书本的页数和sum之间,答案是单调且有序的，因些我们可以二分答案来解这道题。如果二分的mid能让抄书人从后向前分给m个人抄写能分得下,则让继续搜索左区间，继续寻找更小的值，如果m个人抄写还不够，则说明这个mid小了，需要搜索右区间。这道题的输出也值得看一下，利用递归实现倒序输出。

这道题还有一种解法是动态规划可以解决，后续我们会学习。

代码：

```
#include<bits/stdc++.h>
int n,m,ans;
int a[510];
bool check(int x)
{
    int su=0,an=0;//an表示能分几个人
    //printf("%d\n",x);
    for(int i=n;i>=1;i--)
    {
        if(i==1) an++; //最后一人（最前面的）需要特殊处理
        if(su+a[i]<=x)
        {
            su+=a[i];
        }
        else
        {
            an++;su=a[i];
        }
    }
    if(an<=m) return 1;
    return 0;
}
void print(int l,int r) //打印当前左右边界内的部分
{
    int ss=0;
    for(int i=r;i>=l;i--)
    {
        if(ss+a[i]>ans)
        {
            pr(l,i);
            printf("%d %d\n",i+1,r);//逆序输出，回溯时才打印
            return ;
        }
```

```
            ss+=a[i];
        }
        printf("%d %d\n",1,r);//第一个人特殊处理（边界条件）
    }
    int main()
    {
        scanf("%d %d",&n,&m);
        if(n==0) return 0;  //特判0
        int l=0,r,mid;
        for(int i=1;i<=n;i++)
        {
            scanf("%d",&a[i]);
            r+=a[i];
        }
        // 二分模板
        while(l<=r)
        {
            mid=(l+r)/2;
            if(check(mid))
            {
                ans=mid; //ans存的就是每个人能分到的页数最大值，用这个来判输出
                r=mid-1;
            }
            else l=mid+1;
        }
        //printf("%d\n",ans);
        print(1,n);// 打印函数（递归实现倒序输出）
        return 0;
    }
```