二分查找与二分答案

**二分答案：**二分答案与二分查找类似，即对有着单调性的答案进行二分，大多数情况下用于求解满足某种条件下的最大(小)值。

1. **二分查找**

在一个已知的有序数据集上进行二分地查找。

1. **lower\_bound(begin,end,val):**在值有序的数组连续地址[begin,end)中找到第一个位置并返回其地址，使得val插入在这个位置前面，整个数组仍然保持有序。
2. **upper\_bound(begin,end,val):**在值有序的数组连续地址[begin,end)中找到最后一个位置并返回其地址，使得val插入在这个位置前面，整个数组仍然保持有序。
3. **二分答案**

答案有一个区间，在这个区间中二分，直到找到最优答案。

**砍树：**n棵树高度分别为a1,a2,…，an,对于一个砍树高度h，可以锯下并收集到每棵树上比h高的部分的木材(不高于h的部分保持不变)，现在需要求最大的整数高度h，使得能够收集到长度为m的木材。其中n<=10^6,树高不超过10^9.

可以从1开始一米一米地往上面枚举，每次枚举高度都需要计算收集到的木材数量，如果x米时满足要求，但是x+1米时无法满足要求，输出x.这种方法虽然答案是对的，但是复杂度是O(n\*h)，效率很低，因此需要考虑更好的办法。

先来变换一下题目：令“条件”表示“当砍树高度为x时可以获取不少于m的木材”，那么就是要找最大的x使得“条件”成立。再来看一下这个“条件”是否具有单调性：当x超过某个数时，“条件”一定不成立，而不超过这个数时，“条件”一定成立。完全符合二分条件！

#include<bits/stdc++.h>  
using namespace std;  
#define maxn 1000010  
typedef long long LL;  
LL a[maxn],n,m;  
bool P(int h){//当砍树高度为h时，能否得到大于m的木材   
 LL tot=0;  
 for(int i=1;i<=n;i++)  
 if(a[i]>h)  
 tot+=a[i]-h;//按照题意模拟   
 return tot>=m;  
}  
int main(){  
 scanf("%lld%lld",&n,&m);  
 for(int i=1;i<=n;i++)  
 scanf("%lld",&a[i]);  
 int L=0,R=1e9,ans,mid;  
 while(L<=R)  
 if(P(mid=L+R>>1))//一种压行技巧   
 ans=mid,L=mid+1;  
 //如果P(mid)为真，mid可以成为答案，真正的答案可能在mid右侧，左端点右移   
 else  
 R=mid-1;//P(mid)为假，答案在mid左侧，右端点左移  
 printf("%d",ans);  
}

**使用二分答案技巧的条件：**

1. **命题可以被归纳为找到使得某命题P(x)成立(或不成立)的最大(或最小)的x.**
2. **把P(x)看作一个值为真或假的函数，那么它一定在某个分界线的一侧全为真，另一侧全为假。**
3. **可以找到一个复杂度优秀的算法来检验P(x)的真假。**

**通俗来讲，二分答案可以用来处理“最大的最小”或“最小的最大”问题。**