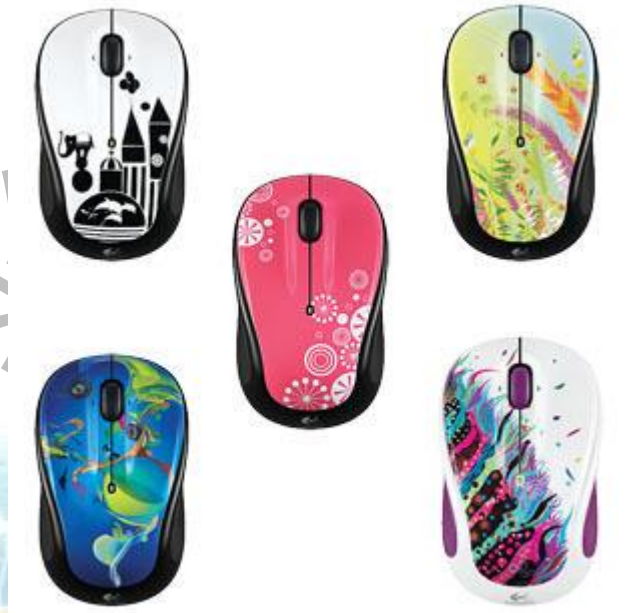
二分法 第2篇

"二分答案"

游戏：老师买了一只鼠标，价格是1到200之间的整数，大家猜猜这个数字是多少？

? 123

二分答案二分出答案来验证是否可行
通俗一点来说就是在[1,n]上
每次取中间点，猜答案..
把答案作为已知条件
判断答案是否满足题意..
从而求证出所需解的过程



引例：砍树

何师傅种了N棵树，今天他需要从这些树中砍下总长为M米的木材。

他有一个伐木机，工作过程如下：**何师傅给伐木机设置一个伐木高度H（米），伐木机会自动把每一棵树比H高的部分锯掉（高度没有超过H米的将保持不变）。锯掉的部分就是何师傅得到的木材。**

例如，有4棵树的高度分别为20，15，10和17，何师傅把伐木机高度设为15米，切割后树木剩下的高度将是15，15，10和15，何师傅将从第1棵树得到5米，从第4棵树得到2米，共得到7米木材。

何师傅非常关注生态保护，他不会砍掉过多的木材。所以他会把伐木机的伐木高度设置得尽可能高。**现在请你帮助何师傅找到伐木机锯片的最大的整数高度H，使得他能得到木材至少为M米。**换句话说，如果再升高1米，则他将得不到M米木材。

输入数据：

第1行：2个整数N和M，N表示树木的数量，M表示需要的木材总长度

第2行：N个整数表示每棵树的高度，值均不超过1000000000。

输出数据：1个整数，表示设置的砍树最高高度。

1<=N<=1000000

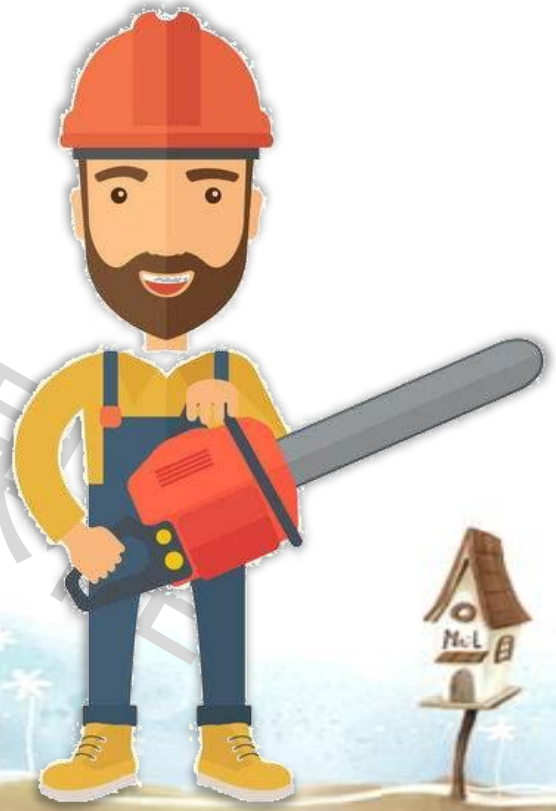
1<=M<=2000000000

输入样例：

```
5 20
4 42 40 26 46
```

输出样例：

```
36
```



引例：砍树 问题分析

输入样例：
5 20
4 42 40 26 46
输出样例：
36

发现规律1：伐木机的高度H设置得越低，砍下的木材越多。
伐木机得高度H设置得越高，砍下的木材越少。

发现规律2：伐木机的高度H设置的范围在0到最高那棵树的树高之间。
以样例数据为例，H的的范围在[0,46]之间

思考：在至少获得M米木材的前提下，伐木机的高度H最高是多少呢？ 我们通过**二分来枚举答案**！

初始时，设置答案可行的上下边界值Lt=0，Rt=46

二分枚举答案mid	验证答案	记录可行的答案和更新枚举的上下边界
mid=(Lt+Rt)/2=23	讨论H=23是否可行?	砍下的木材总量为62，62>20,可行！Ans=23 Lt= Mid+1 = 24
mid=(Lt+Rt)/2=35	讨论H=35是否可行?	砍下的木材总量为23，23>20,可行！Ans=35 Lt= Mid+1 = 36
mid=(Lt+Rt)/2=41	讨论H=41是否可行?	砍下的木材总量为6，6<20不可行！ Rt= Mid-1 = 40
mid=(Lt+Rt)/2=38	讨论H=38是否可行?	砍下的木材总量为14，14<20不可行！ Rt= Mid-1 = 37
mid=(Lt+Rt)/2=36	讨论H=36是否可行?	砍下的木材总量为20，20==20,可行！Ans=36 Lt= Mid+1 = 37
mid=(Lt+Rt)/2=37	讨论H=37是否可行?	砍下的木材总量为17，17<20不可行！ Rt=Mid-1=36

此刻：Lt=37,Rt=36,Lt>Rt,结束，答案即是Ans

引例：砍树 参考代码

```cpp
long long m,n,a[1000005];

long long check(long long x) //验证值为x的答案是否可行
{
    long long sum=0;
    for(int i=1;i<=n;i++)
    {
        if(a[i]>x)
        sum=sum+(a[i]-x);  //sum记录得到的木材总长度
    }
    return sum;
}
```

```cpp
int main()
{
    long long tot,ans=0,temp=0;
    scanf("%d%d",&n,&m);
    for(int i=1;i<=n;i++)
    {
        scanf("%d",&a[i]);
        temp=max(temp,a[i]);     //temp用来记录最高的树的高度
    }


    long long mid,Lt=0,Rt=temp;   //把右边界设成最高的树的高度
    while(Lt<=Rt)                 //二分的进行条件
    {
        mid=(Lt+Rt)/2;
        tot=check(mid);
        if(tot<m)Rt=mid-1;
        else { ans=mid; Lt=mid+1;}
    }
    printf("%d",ans);
}
```

时间复杂度$O(nlog_2{}^v)$

n为树木总棵数
v为最高树木的高度

引例：砍树 解题小结

二分答案可求解问题的特点：
1.答案的上下界已知
2.随着答案的变化，问题的可行性会随之发展变化

例1：跳房子游戏

奶牛们在举办一种特殊的跳房子游戏，在这个游戏中，大家小心翼翼地在河中的岩石上跳。这个游戏在一条**笔直的河中进行，以一块岩石表示开始，以另一块距离起点L单位长度的岩石表示结束。**在这两块岩石**中间还有N块岩石，第i块岩石距离起点是Di个单位长度。**

每头牛从开始的那块岩石想办法要跳到表示结束的那块岩石上。中间只能在从某块岩石跳跃到另一块岩石，反复的这样跳。为了增加游戏难度，农夫约翰计划移除一些岩石，从而增加奶牛在跳跃时需要的最短距离。他要移除M块岩石，但是不能移除开始和结束的两块岩石。

求**移除了M块岩石后，奶牛从开始跳到结束的岩石，每次跳跃的最短距离至多可以增加到多少。**

输入数据：

第一行：三个用空格分开的整数，分别是 L，N 和 M

第二行：N个整数，其中第i个数表示第i块岩石的位置Di，没有两块岩石处于同一位置。

输出数据：

一个整数表示移除某M块岩石后，相邻岩石间距最小值的最大可能情况。

输入样例：

```
25 5 2
2 14 11 21 17
```

输出样例：

```
4
```

样例解释：

没有移除任何岩石之前，最少需要跳2个单位长度，从0到2。当移除了位于 2 和 14 的两块岩石后，需要的最短跳跃距离就变成了4。（从 17 到 21 或 从 21 到 25）。

```
1 <= L <= 1,000,000,000
0 <= N <= 50,000
0 <= M <= N
```

例1：跳房子游戏 分析

提炼题意：给出n+2块(+2是包含了起点和终点)岩石，从中移除m块，使得相邻两块岩石的间距尽可能大。

发现规律：移除的石头越多，间距增加得越大。移除的石头越少，间距增加得越小。
也就是，要使得间距增加得越大，需要移除越多得岩石。

我们考虑二分岩石间的间距mid。
先按到起点的距离，由小到大排序。
对于相邻两块岩石Dx,Dy，若Dy-Dx<mid，就将岩石Dy移除。

输入样例：
25 5 2
2 14 11 21 17
输出样例：
4

初始时，将岩石按距离排序0 2 11 14 17 21 25 设置答案可行的上下边界值Lt=0, Rt=25

二分枚举答案mid	验证答案	记录可行的答案和更新枚举的上下边界
mid=(Lt+Rt)/2=12	讨论间距=12是否可行?	需要移除4块岩石，4>2,不可行! Rt= Mid-1 = 11
mid=(Lt+Rt)/2=5	讨论间距=5是否可行?	需要移除3块岩石，3>2,不可行! Rt= Mid-1 = 4
mid=(Lt+Rt)/2=2	讨论间距=2是否可行?	需要移除0块岩石，0<2可行! Ans=2 Lt= Mid+1 = 3
mid=(Lt+Rt)/2=3	讨论间距=3是否可行?	需要移除1块岩石，1<2可行! Ans=3 Lt= Mid+1 = 4
mid=(Lt+Rt)/2=4	讨论间距=4是否可行?	需要移除2块岩石，2==2可行! Ans=4 Lt= Mid+1 = 5

此刻：Lt=5,Rt=4,Lt>Rt,结束，答案即是Ans

例1：跳房子游戏 参考程序

提炼题意：给出n+2块（+2是包含了起点和终点）岩石，从中移除m块，使得相邻两块岩石的间距尽可能大。

```cpp
int main(){
    int L, n, m, Lt = 0, Rt, ans=0;
    cin >> L >> n >> m;
    Rt = L;
    for (int i = 1; i <= n; i++)cin >> d[i];

    sort(d + 1, d + 1 + n);
    while (Lt <= Rt)
    {
        int mid = (Lt + Rt) / 2;
        if (check(n, m, mid))
        {
            ans=mid; Lt = mid + 1;
        } else Rt = mid - 1;

    }
    cout << ans;
}
```

```cpp
int d[50007];

bool check(int n, int m, int k)
{
    //last记录上一块保留的岩石的位置，cnt记录已被移除的岩石的数量
    int last = 0, cnt = 0;
    for (register int i = 1; i <= n; i++)
    {
        if (d[i] - last < k)  //第i号岩石与上一块岩石的间距小于k,i需要被移除
        {
            cnt++;
            if (cnt > m) return false;
        } else last = d[i];  //第i块岩石被保留，记录下最新保留的岩石的位置

    }
    return true;
}
```

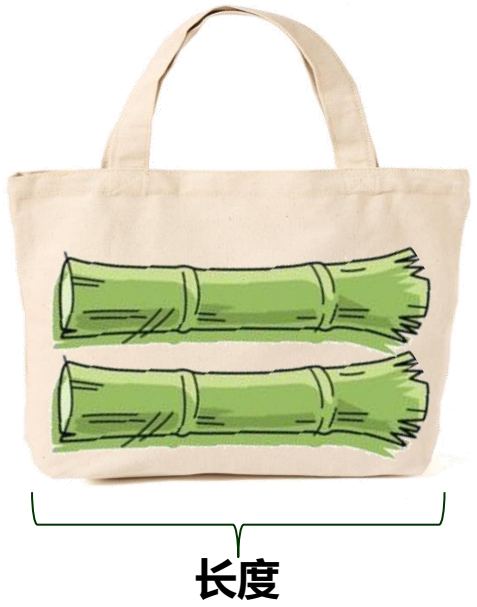
例题2：砍甘蔗

何老板买了一根很长的甘蔗，总共有N节(1<=N<=100,000)。为方便带回家，他打算把甘蔗砍成M(1<=M<=N)段。只能从相邻两小节的交界处砍断。何老板测量了每一节的长度，他想知道至少需要一个长度为多少的购物袋，才能装得下这些甘蔗，请你帮他算一下。（也就是：希望砍完后，最长的一段尽可能短）

输入数据：第一行输入两个用空格隔开的正整数N和M

以下N行每行一个不超过10000正整数，依次表示从左往右每一节甘蔗的长度。

输出数据：输出购物袋的最小长度。



输入样例

```
7 5
100
400
300
100
500
101
400
```

输出样例

```
500
```

样例说明

采取下面的方案可以使购物袋的长度不多于500。这个答案不能再少了。

```
 100 400      300 100      500      101      400      每一节甘蔗的长度
<-------1 <-------2    <---3 <---4    <---5    砍成5段
   500          400       500      101      400      每一段的长度
```

分析问题：

购物袋长度**可能**的范围值：

最少 500

最多 100+400+300+100+500+101+400=1901

也就是说，答案肯定在100到1901之间

怎么讨论答案呢？ 二分枚举 Lt=100, Rt=1901

mid=(Lt+Rt)/2=1000,	讨论1000是否可行?	可行! Ans=1000 Rt=Mid-1 =999
mid=(Lt+Rt)/2=549,	讨论549是否可行?	可行! Ans=549 Rt=Mid-1=548
mid=(Lt+Rt)/2=324,	讨论324是否可行?	不可行! Lt=Mid+1=325
mid=(Lt+Rt)/2=436,	讨论436是否可行?	不可行! Lt=Mid+1=437
mid=(Lt+Rt)/2=492,	讨论492是否可行?	不可行! Lt=Mid+1=493
mid=(Lt+Rt)/2=520,	讨论520是否可行?	可行! Ans=520 Rt=Mid-1=519
mid=(Lt+Rt)/2=506,	讨论506是否可行?	可行! Ans=506 Rt=Mid-1=505
mid=(Lt+Rt)/2=499,	讨论499是否可行?	不可行! Lt=Mid+1=500
mid=(Lt+Rt)/2=502,	讨论502是否可行?	可行! Ans=502 Rt=Mid-1=501
mid=(Lt+Rt)/2=500,	讨论500是否可行?	可行! Ans=500 Rt=Mid-1=499

此刻：Lt=500,Rt=499,Lt>Rt,结束，答案即是Ans

输入样例

```
7 5
100
400
300
100
500
101
400
```

输出样例

```
500
```

例题2：砍甘蔗 分析

验证：
按顺序检查每一节甘蔗，记录当前购物袋剩余的长度；
如果长度不超过当前购物袋，将这节甘蔗放入购物袋；
否则使用一个新的购物袋，并将它放进新购物袋中。

【参考代码】验证

```
bool Check(int x)                    //x为当前二分的袋子长度
{
    int y = x, cnt = 1;    //y记录当前袋子的剩余长度，cnt记录甘蔗段数
    for (int i = 1; i <= n; i++)
        if (y >= a[i]) y -= a[i];
        else { cnt++, y = x - a[i];}
    return cnt <= m;
}
```

【二分答案，代码模板】

```
Lt=答案可能的下限;
Rt=答案可能的上限;
while(Lt<=Rt)
{
        Mid=(Lt+Rt)/2;
        if(Check( Mid )){ Ans=Mid;Rt=Mid-1;}//自定义函数check( )用于判定当前答案是否可行
        else Lt=Mid+1;
}
printf("%d\n",Ans);
```