

C++ 算法

二分查找框架

浮点数答案

例题：开二次方根

输入正数 a ，输出 \sqrt{a} ，保留两位小数。 $a \leq 1000$

样例输入

4

样例输出

2.00

样例输入

10

样例输出

3.16

样例输入

66.66

样例输出

8.16

思考1：

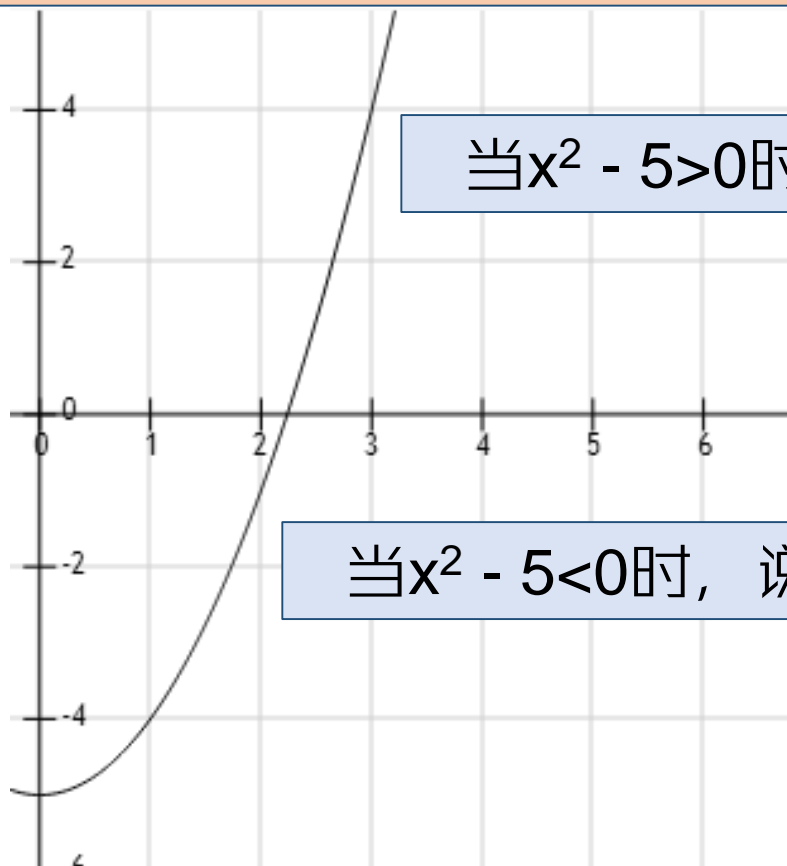
如何使用二分查找算法
计算开二次方根？

思考2：

如何确保两位小数的精确度？

例题：开二次方根

如何计算 $\sqrt{5}$ 的数值，保留两位小数



当 $x^2 - 5 > 0$ 时，说明 x 太大了

当 $x^2 - 5 < 0$ 时，说明 x 太小了

$f(x) = x^2 - 5$ 的函数图像
与 x 正半轴有唯一相交点，其横坐标就是 $\sqrt{5}$

```
1 #include<iostream>
2 #include<cmath>
3 #include<iomanip>
4 #define ERR 0.000001
5 using namespace std;
```

设置误差范围

```
6 double a;
7 bool tooSmall(double x){return x*x<a;}
```

```
8 int main() {
```

```
9     cin>>a;
```

```
10     double l=0,r=1000;
```

```
11     while(r-l>ERR){
```

```
12         double mid=l+(r-l)/2;
```

```
13         if(tooSmall(mid)) l=mid;
```

```
14         else r=mid;
```

```
15     }
```

```
16     cout<<fixed<<setprecision(2)<<r<<endl;
```

```
17     cout<<fixed<<setprecision(2)<<sqrt(a)<<endl;
```

```
18     return 0;
```

```
19 }
```

二分查找框架

二分查找框架：浮点数答案

```
int l= 初始化左端点, 即答案可能的最小值
int r= 初始化右端点, 即答案可能的最大值
int ans= 初始化答案
while(r-l>ERR){ 当待查找范围不够精确
    int mid=l+(r-l)/2; 二分范围: 中点为mid
    if(OK(mid))
    else
        根据mid是否为可行解
        更新范围端点和答案
        l=mid
        r=mid
}
cout<<ans<<endl;
```

汇总：二分查找问题

显性数组元素查找

身高查询

狙击装备

僵尸幸存者2

单调函数求零点

开二次方

特殊三次方程

最大化分组和的最小值

臭皮匠2

最小化分组和的最大值

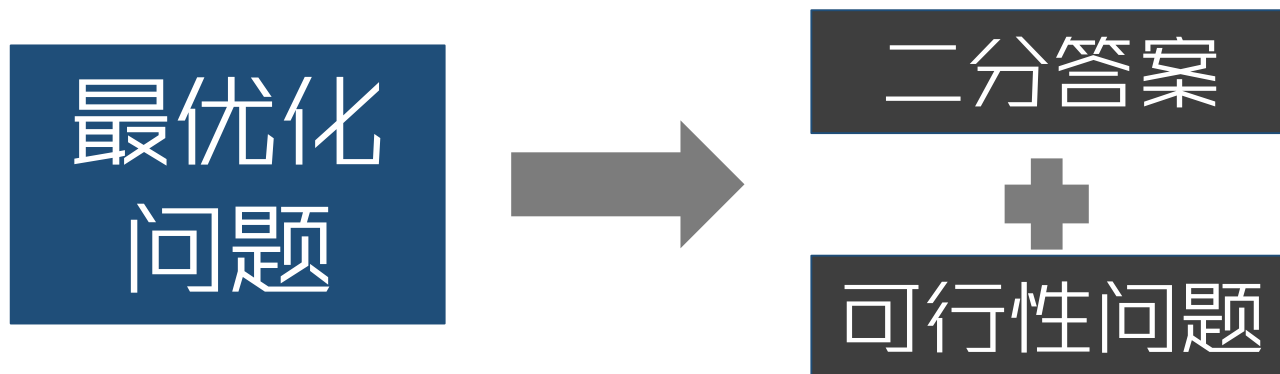
高智商罪犯2

秘籍修复

最大化位置间隔最小值

高智商罪犯3

典型二分查找问题



最大化分组和的最小值

臭皮匠2

最小化分组和的最大值

高智商罪犯2

秘籍修复

最大化位置间隔最小值

高智商罪犯3

最大化平均值问题

送礼就要体面

送礼讲究“体面”，所以很多礼品包装都非常漂亮。而你的送礼哲学是“体面”体现在“体积”。从 n 个礼品里你需要挑选 k 个，第 i 件的体积是 v_i ，价格是 p_i 。请问对于选出的 k 件礼品，单位价格的体积最大是多少？（也就是总体积除以总价格希望最大化）
输入第一行是 n 和 k ，之后 n 行每行是 v_i 和 p_i 。 $1 \leq k \leq n \leq 100000$
输出有个浮点数，保留两位小数

输入样例：

3 2

2 2

3 5

1 2

输出样例：

0.75

思考：能否使用贪心算法

送礼就要体面

错误的贪心算法：

1. 计算每件礼品*i*的性价比： v_i 除以 p_i
2. 将所有礼品按照性价比排序
3. 依次挑选性价比最高的*k*件礼品

输入样例：

3 2

2 2

3 5

1 2

输出样例：

0.75

礼品性价比列表：

$$2.0/2.0=1.0$$

$$3.0/5.0=0.6$$

$$1.0/2.0=0.5$$

如果选取最高性价比的两件

答案

$$= 5.0/7.0$$

$$= 0.71$$

正确算法：二分答案+判断可行性

二分枚举可能的答案：

答案 x 初始化范围为 $[0, \max_v / \min_p]$ 里的浮点数

对于特定的 x , 判断可行性：

能否选出 k 件礼品性价比不低于 x ?

不断二分缩小 x 的可能范围


直到待查找范围大小小于误差允许的范围


判断可行性


对于特定的 x , 判断可行性:
能否选出 k 件礼品性价比不低于 x ?

$$\frac{\sum_{i \in S} v_i}{\sum_{i \in S} p_i} \geq x$$

是否存在礼品集合 S , 含有 k 个礼品:
总体性价比满足左侧不等式


$$\sum_{i \in S} v_i \geq x \sum_{i \in S} p_i$$


$$\sum_{i \in S} v_i - x \sum_{i \in S} p_i \geq 0$$


$$\sum_{i \in S} (v_i - x \cdot p_i) \geq 0$$

是否存在礼品集合 S , 含有 k 个礼品:
它们的 $(v_i - x \cdot p_i)$ 数值总和不小于0

容易判断!

判断可行性

对于特定的 x , 判断可行性:
能否选出 k 件礼品性价比不低于 x ?

$$\sum_{i \in S} (v_i - x \cdot p_i) \geq 0$$

是否存在礼品集合 S , 含有 k 个礼品:
它们的 $(v_i - x \cdot p_i)$ 数值总和不少于0

贪心算法判断可行性:

1. 计算每件礼品 i 的剩余价值 $z_i = (v_i - x \cdot p_i)$
2. 将所有礼品按照剩余价值排序
3. 依次挑选剩余价值最高的 k 件礼品
4. 判断它们剩余价值总和是否非负

送礼就要体面

```
10 bool OK(double x){  
11     for(int i=0;i<n;i++)z[i]=v[i]-x*p[i];  
12     sort(z,z+n);  
13     double sum=0;  
14     for(int i=n-k;i<n;i++)sum+=z[i];  
15     return sum>=0;  
16 }
```

```
20     double maxv=*max_element(v,v+n);  
21     double minp=*min_element(p,p+n);  
22     double l=0,r=maxv/minp,ans=0;  
23     while(r-l>ERR){  
24         double mid=l+(r-l)/2;  
25         if(OK(mid)) ans=l=mid;  
26         else r=mid;  
27     }  
28     cout<<ans<<endl;
```

作业

现场
挑战

427

现场
挑战

446

447

拓展题

448

拓展题

449