

信奥算法

双游标

two pointers

快快编程
kkcoding.net

现场挑战 快快编程991

快快编程
kkcoding.net

本题是计数问题

快快编程
kkcoding.net

纯暴力枚举

枚举第1个人i

枚举第2个人j

若这两人智商达标

答案计数器加1

时间复杂度
 $O(n^2)$

纯暴力枚举

猜猜得
几分?

60分

8
ans可以用int吗

```
cin>>n>>m;
for(int i=1;i<=n;i++)cin>>x[i];
long long ans=0;
for(int i=1;i<=n-1;i++)
    for(int j=i+1;j<=n;j++)
        if(x[i]+x[j]>=m)
            ans++;
cout<<ans<<endl;
```

【数据规模与约定】

50%数据， $n \leq 1000$

100%数据， $n \leq 200000$

快快编程
kkcoding.net

纯枚举两个人
计算量约 200000^2
一定会超时
如何优化？

只枚举一个人
巧算另一人的范围

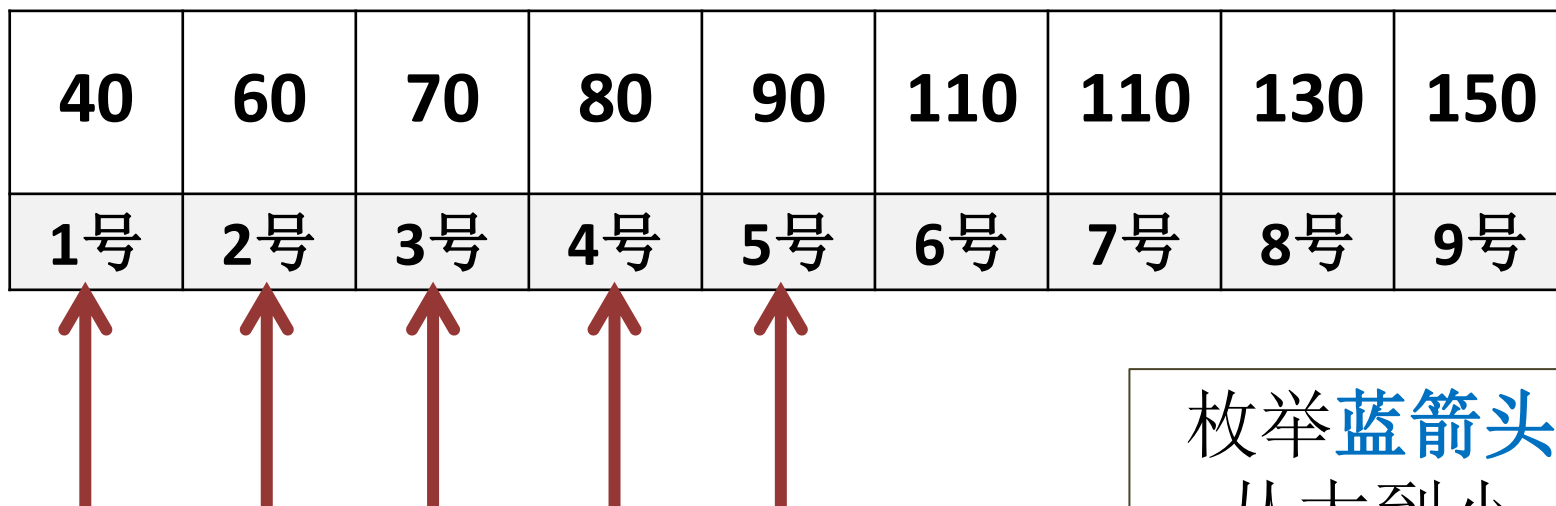
减少枚举对象
优化枚举顺序

双游标

先按照智商从小到大排序

目标
凑满
200

40	60	70	80	90	110	110	130	150
1号	2号	3号	4号	5号	6号	7号	8号	9号



$$\text{ans} = 0 + (9 - 2) + (8 - 3) + (7 - 5) + (6 - 5)$$

枚举**蓝箭头**
从大到小
红箭头右移
配合**蓝箭头**

先按照智商从小到大排序

主要枚举右端点
较高的智商

巧算左端点的范围：
较低的智商至少要多少才能达标

双游标

```
8   cin>>n>>m;
9   for(int i=1;i<=n;i++)cin>>x[i];
10  sort(x+1,x+1+n);

12  int i=1,j=n;
13  while(i<j){
14      while(i<j&& x[i]+x[j]<m)
15          ++i;
16      ans+=j-i;
17      --j;
18  }
19  cout<<ans<<endl;
```

请跟着老师翻译
理解每一行

时间复杂度
 $O(n)$

现场挑战 快快编程968

快快编程
kkcoding.net

本题是最优化问题

快快编程
kkcoding.net

纯暴力枚举

枚举左边桩子l

枚举右边桩子r

计算横幅面积

更新最优答案

时间复杂度
 $O(n^2)$

纯暴力枚举

猜猜得
几分?

80分

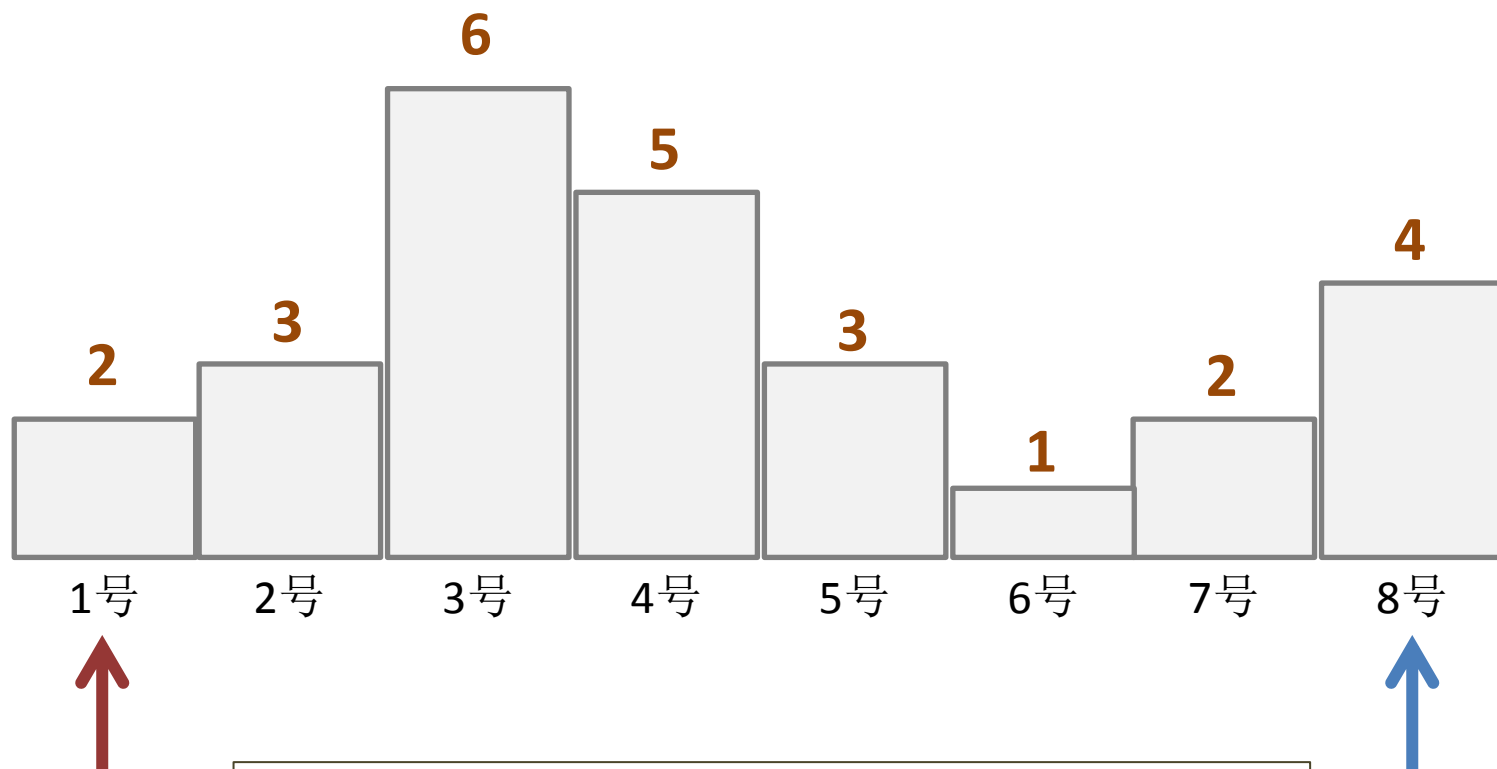
```
9      cin>>n;
10     for(ll i=1;i<=n;i++)cin>>h[i];
11     for(ll l=1;l<=n;l++)
12         for(ll r=l+1;r<=n;r++){
13             ll height=min(h[l],h[r]);
14             ll width=r-l+1;
15             ll area=height*width;
16             ans=max(ans,area);
17         }
18     cout<<ans<<endl;
```

纯枚举两端木桩
计算量约 200000^2
一定会超时
如何优化？

只枚举一个人
巧算另一人的范围

枚举顺序上的优化

双游标



面积 $(8-1+1)*\min(h[1],h[8])=8*2=16$

此时哪个游标该能移动呢?

左板更短，左游标固定的最优解，已经找到

所以左游标应该向右移动

双游标

```
9      cin>>n;  
10     for(ll i=1;i<=n;i++)cin>>h[i];  
11     ll l=1,r=n,ans=0;  
12     while(l<r){  
13         ll height=min(h[l],h[r]);  
14         ll width=r-l+1;  
15         ll area=height*width;  
16         ans=max(ans,area);  
17         if(h[l]<h[r])l++;  
18         else r--;  
19     }  
20     cout<<ans<<endl;
```

跟着老师翻译
理解每一行

时间复杂度
 $O(n)$

双游标：优化原理

双游标相对于暴力枚举两端优化的核心是什么？

避免了无效枚举

最优性
剪枝

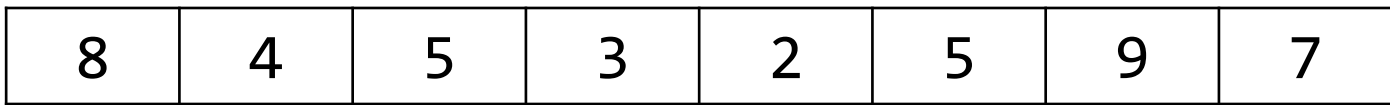
若不可能成为最优解
就避免枚举

快速排序

quick sort

算法可视化网址:

visualgo.net/en/sorting

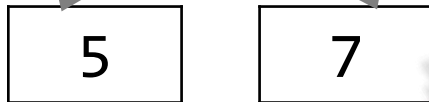
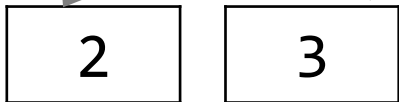
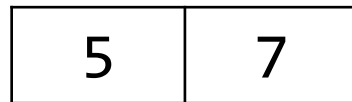


基准数5



基准数3

基准数8



快速排序

quick sort

请同学们总结快速排序的算法步骤

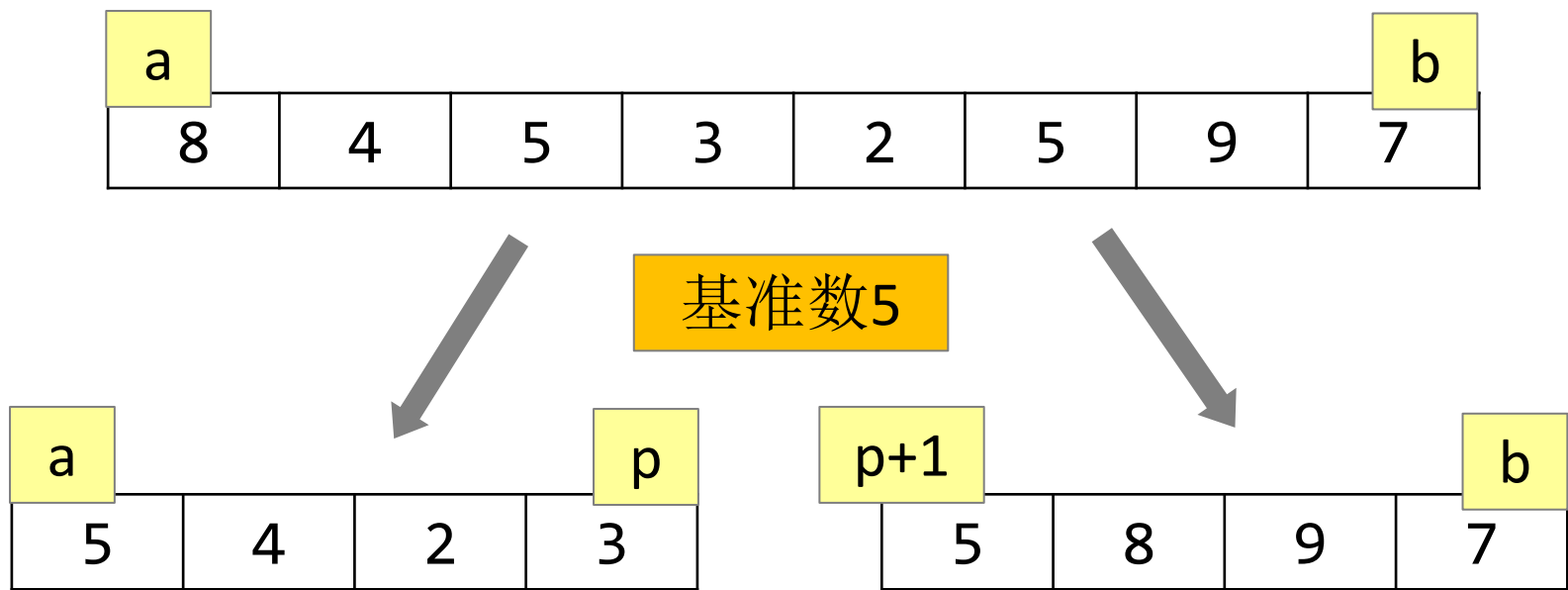
找一个基准数(pivot)

把数组分成左右两部分(partition)

左边：都不超过基准数

右边：都不小于基准数

左右继续递归操作



```
15 void qsort(int a, int b){  
16     if(a >= b) return;  
17     int p = partition(a, b);  
18     qsort(a, p);  
19     qsort(p+1, b);  
20 }
```

递归

用基准数分组

```
5 int partition(int a,int b){  
6     int pivot=x[(a+b)/2];  
7     int i=a-1,j=b+1;  
8     while(1){  
9         while(x[++i]<pivot);  
10        while(x[--j]>pivot);  
11        if(i>=j)return j;  
12        swap(x[i],x[j]);  
13    }  
14 }
```

取基准数

左右双游标

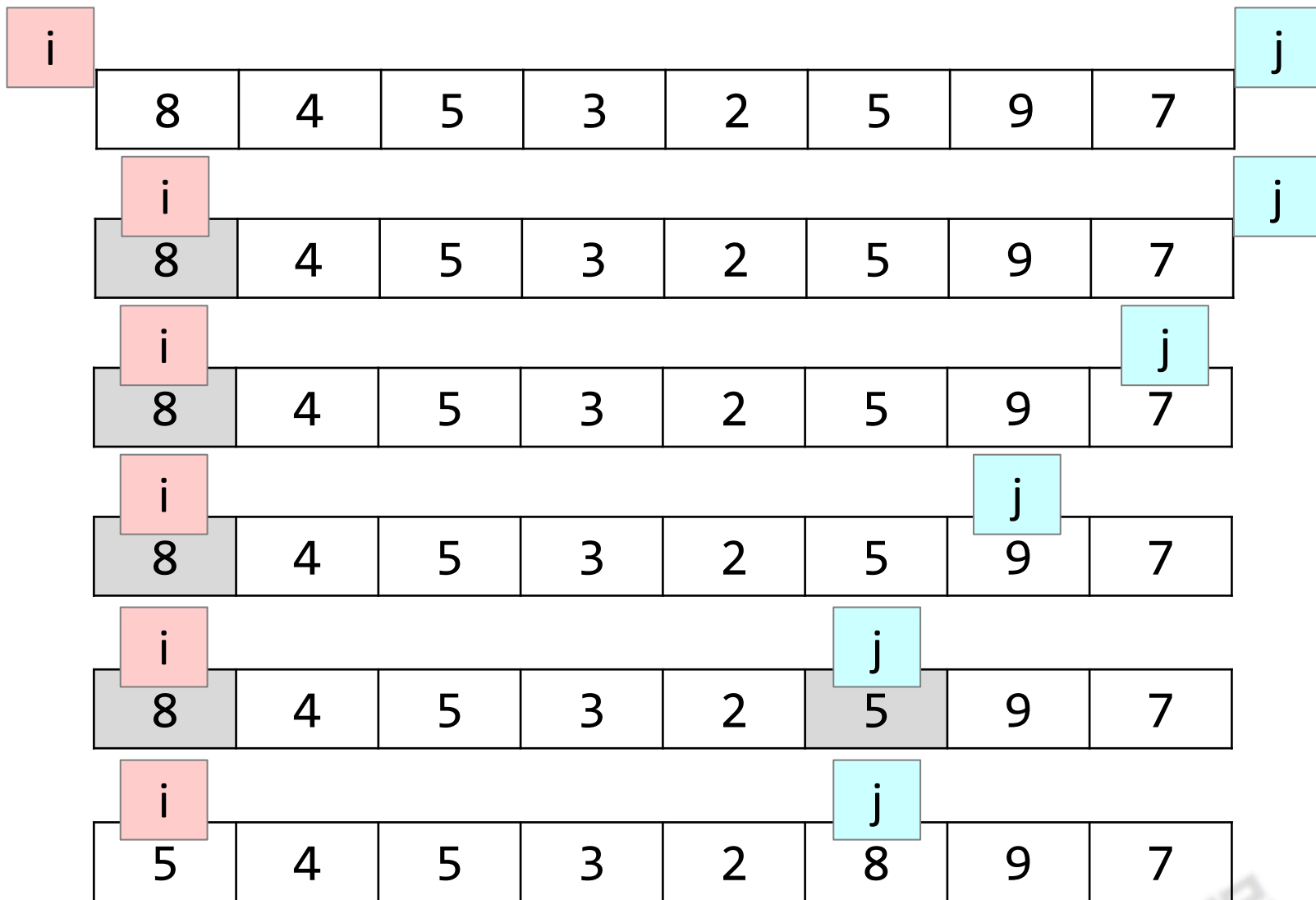
循环停止时 $x[i] \geq \text{pivot}$

循环停止时 $x[j] \leq \text{pivot}$

左右游标相遇

交换变量位置

基准数
5



左游标 i 右移
直到发现可以
放右半部分的元素

右游标 j 左移
直到发现可以
放左半部分的元素

基准数
5

i	5	4	5	3	2	j	8	9	7				
	5	i	4	5	3	2	j	8	9	7			
	5	4		i	5	3	2	j	8	9	7		
	5	4			i	5	3	2	j	8	9	7	
	5	4				i	5	3	2	j	8	9	7

左游标i右移
直到发现可以
放右半部分的元素

右游标j左移
直到发现可以
放左半部分的元素

基准数
5

		i		j			
5	4	2	3	5	8	9	7

			i	j			
5	4	2	3	5	8	9	7

				i	j			
5	4	2	3	5	5	8	9	7
左半部分全部 大于等于基准数					右半部分全部 小于等于基准数			

左游标i右移
直到发现可以
放右半部分的元素

右游标j左移
直到发现可以
放左半部分的元素

每次分组的复杂度 $O(b-a+1)$

```
5 int partition(int a,int b){  
6     int pivot=x[(a+b)/2];  
7     int i=a-1,j=b+1;  
8     while(1){  
9         while(x[++i]<pivot);  
10        while(x[--j]>pivot);  
11        if(i>=j)return j;  
12        swap(x[i],x[j]);  
13    }  
14 }
```

快速排序的平均复杂度 $O(n\log n)$

```
15 void qsort(int a,int b){  
16     if(a>=b)return;  
17     int p=partition(a,b);  
18     qsort(a,p);  
19     qsort(p+1,b);  
20 }
```

递归

最差情况的复杂度 $O(n^2)$

最差情况：基准数一直选中最小数

思考题

给定无序数组x共n个数字
求出其中第k小数字

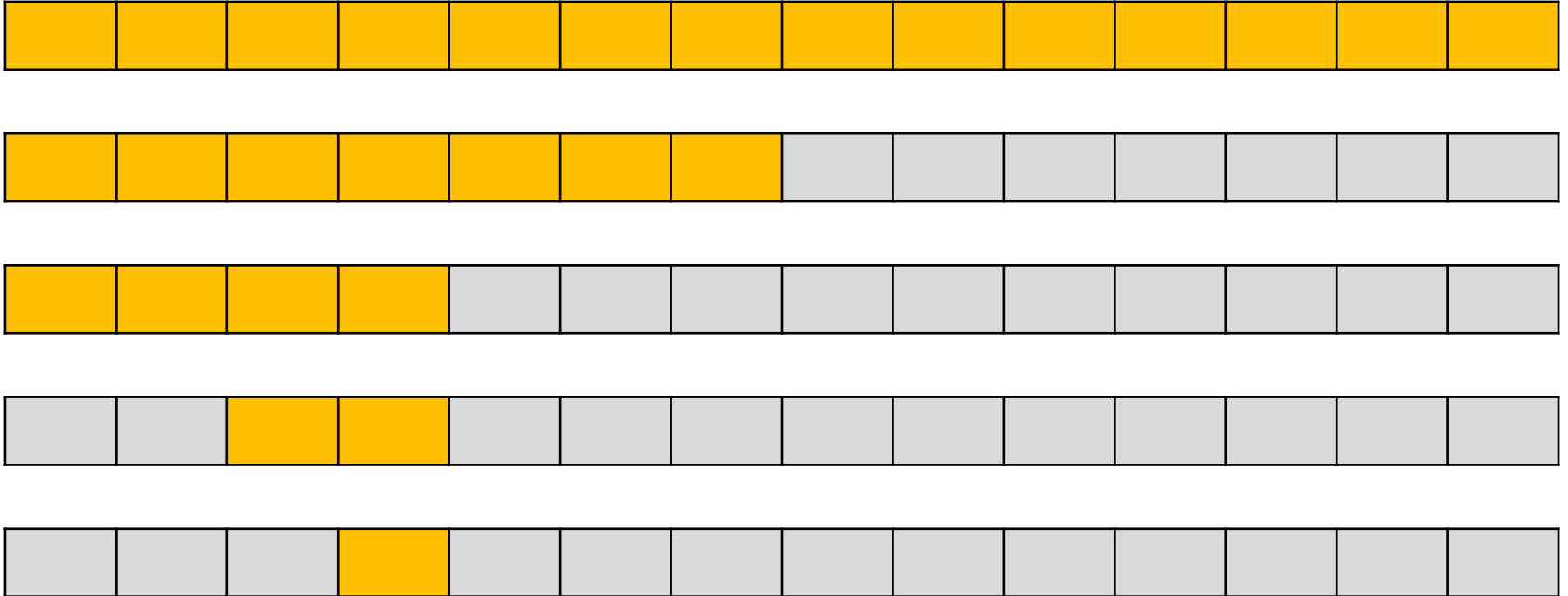
能否比 $O(n\log n)$ 更快?

第 $m+1$ 小数，即 m 号

```
15 □ int qsort(int a,int b){  
16     if(a>=b)return x[b];  
17     int p=partition(a,b);  
18     if(m<=p)return qsort(a,p);  
19     else return qsort(p+1,b);  
20 }
```

复杂度 $O(N)$
为什么

例：求第4小数



复杂度 $O(N)$
为什么

快快编程作业

991

968

612

用快速排序实现

拓展题

977,656