重庆南开中学

分治法

南开中学信息学竞赛教练组

重庆南开中学

目录
contents

PART 01

作业讲评

- 讲不讲？讲！

作业3592 - 人数统计

题目大意

输入n个整数，m次询问，每次问有多少值为k的数。

数据范围$n \leq 100000, m \leq 80000$。

解析

- 课上讲过，用两次二分查找。
- 还可以用C++自带的二分查找函数
- lower_bound(...)
  - 找到第一个$\geq x$的数据；
- upper_bound(...)
  - 找到第一个$> x$的数据；
  - 在整数类型时等价于lower_bound传入x+1；
  - 在double、string等类型时不能用lower_bound代替。

作业3592 - 人数统计

题目大意

输入n个整数，m次询问，每次问有多少值为k的数。

数据范围$n \leq 100000, m \leq 80000$。

解析

- 函数参数：
  - 前两个参数和sort相同，例如a+1和a+1+n；
  - 第三个参数传入需要二分查找的值x；
  - 如果需要，可以在第四个参数传入比较函数的名字，和sort的比较函数相同。
- 返回值：
  - 返回值不是数组下标i，而是a+i；
  - 可以用返回值减去a得到数组下标i。
  - 注意，如果数组中没有符合条件的数，返回值不再数组之内，如a+1+n。

作业3592 - 人数统计

题目大意

输入n个整数，m次询问，每次问有多少值为k的数。

数据范围$n \leq 100000, m \leq 80000$。

解析

- 这道题怎么写？

```
int ans = upper_bound(a + 1, a + 1 + N, k) - lower_bound(a + 1, a + 1 + N, k);
//int ans = lower_bound(a + 1, a + 1 + N, k + 1) - lower_bound(a + 1, a + 1 + N, k);
printf("%d\n", ans);
```

- 可以将两次调用的返回值相减，得到的整数就是自己写二分查找时的两次数组下标之差。

作业3593 - 工资统计

题目大意

输入n个的能力，能力是互不相同的整数。如果按从小到大排序后，第i个人的工资是$a[i]-i$。m次询问，每次问工资为k的人数。数据范围$n \le 100000, m \le 80000$。

解析

- 输入之后先排序，然后第i个数减i；
- 是否需要再排序呢？
- $(a[i+1]-i-1)-(a[i]-i)=a[i+1]-a[i]-1 \ge 0$
- 数组仍然有序，不需要再次排序。

作业3144 - 何老板的淘宝店3.0

题目大意

有n个商品，m次询问，每次找出价格与x最接近的商品的价格。

数据范围$n, m \leq 300000$

解析

- 二分查找找到第一个$\geq x$的商品的数组下标i；
- 如果$a[i] = x$，直接输出；
- 如果$i > 1$，且$x - a[i-1] \leq a[i] - x$，输出$a[i-1]$，否则输出$a[i]$。

作业5455 - 苹果采摘

题目大意

一个长度为n的序列，统计有多少个连续段的和为m。

数据范围$n \leq 100000$。

解析

- 方法1：
- 在前缀和上二分查找：
- 设前缀和数组$s[0], s[1], \ldots, s[n]$，其中$s[0] = 0$；
- 从第i个到第j个苹果的连续段，有$s[j] - s[i-1]$个。
- 枚举j，在s数组上二分查找值为$s[j] - m$的数。
- 时间复杂度$O(n \log n)$。

作业5455 - 苹果采摘

题目大意

一个长度为n的序列，统计有多少个连续段的和为m。

数据范围$n \leq 100000$。

解析

- 方法2：
- 从小到大枚举j，同时另一个变量i始终保持$s[j] - s[i] \leq m$；
- 当从$j-1$变成j时，一步一步挪动i，直到满足条件。

作业5455 - 苹果采摘

题目大意

一个长度为n的序列，统计有多少个连续段的和为m。

数据范围$n \leq 100000$。

解析

- 手动模拟样例，$n = 8, m = 5$:

	0	1	2	3	4	5	6	7	8
s[ ]	0	3	5	9	10	15	17	18	20

作业5455 - 苹果采摘

题目大意

一个长度为n的序列，统计有多少个连续段的和为m。

数据范围$n \leq 100000$。

解析

- 手动模拟样例，$n = 8, m = 5$:

	0	1	2	3	4	5	6	7	8
s[ ]	0	3	5	9	10	15	17	18	20

- 初始化

作业5455 - 苹果采摘

题目大意

一个长度为n的序列，统计有多少个连续段的和为m。

数据范围$n \leq 100000$。

解析

- 手动模拟样例，$n = 8, m = 5$:

	0	1	2	3	4	5	6	7	8
s[ ]	0	3	5	9	10	15	17	18	20

i　　j

- 初始化
- j变成2，i不变。此时s[j]-s[i]==m，所以ans++。

作业5455 - 苹果采摘

题目大意

一个长度为n的序列，统计有多少个连续段的和为m。

数据范围$n \leq 100000$。

解析

- 手动模拟样例，$n = 8, m = 5$:

	0	1	2	3	4	5	6	7	8
s[ ]	0	3	5	9	10	15	17	18	20

i j

- 初始化
- j变成2，i不变。此时s[j]-s[i]==m，所以ans++。
- j变成3，i挪动到2。

作业5455 - 苹果采摘

题目大意

一个长度为n的序列，统计有多少个连续段的和为m。

数据范围$n \leq 100000$。

解析

- 手动模拟样例，$n = 8, m = 5$:

	0	1	2	3	4	5	6	7	8
s[ ]	0	3	5	9	10	15	17	18	20

i j

- 初始化
- j变成2，i不变。此时s[j]-s[i]==m，所以ans++。
- j变成3，i挪动到2。
- j变成4，i不变。此时s[j]-s[i]==m，所以ans++。

作业5455 - 苹果采摘

题目大意

一个长度为n的序列，统计有多少个连续段的和为m。

数据范围$n \leq 100000$。

解析

- 手动模拟样例，$n = 8, m = 5$:

	0	1	2	3	4	5	6	7	8
s[ ]	0	3	5	9	10	15	17	18	20

- 初始化
- j变成2，i不变。此时s[j]-s[i]==m，所以ans++。
- j变成3，i挪动到2。
- j变成4，i不变。此时s[j]-s[i]==m，所以ans++。
- j变成5，i挪动到4。此时s[j]-s[i]==m，所以ans++。

作业5455 - 苹果采摘

题目大意

一个长度为n的序列，统计有多少个连续段的和为m。

数据范围$n \leq 100000$。

解析

- 手动模拟样例，$n = 8, m = 5$:

	0	1	2	3	4	5	6	7	8
s[ ]	0	3	5	9	10	15	17	18	20

i j

- 初始化
- j变成2，i不变。此时s[j]-s[i]==m，所以ans++。
- j变成3，i挪动到2。
- j变成4，i不变。此时s[j]-s[i]==m，所以ans++。
- j变成5，i挪动到4。此时s[j]-s[i]==m，所以ans++。
- ……

作业5455 - 苹果采摘

题目大意

一个长度为n的序列，统计有多少个连续段的和为m。

数据范围$n \leq 100000$。

解析

- 手动模拟样例，$n = 8, m = 5$:

	0	1	2	3	4	5	6	7	8
s[ ]	0	3	5	9	10	15	17	18	20

i　　j

- 初始化
- j变成2，i不变。此时s[j]-s[i]==m，所以ans++。
- j变成3，i挪动到2。
- j变成4，i不变。此时s[j]-s[i]==m，所以ans++。
- j变成5，i挪动到4。此时s[j]-s[i]==m，所以ans++。
- ……

作业5455 - 苹果采摘

题目大意

一个长度为n的序列，统计有多少个连续段的和为m。

数据范围$n \leq 100000$。

解析

- 手动模拟样例，$n = 8, m = 5$:

	0	1	2	3	4	5	6	7	8
s[ ]	0	3	5	9	10	15	17	18	20

i　　j

- 初始化
- j变成2，i不变。此时s[j]-s[i]==m，所以ans++。
- j变成3，i挪动到2。
- j变成4，i不变。此时s[j]-s[i]==m，所以ans++。
- j变成5，i挪动到4。此时s[j]-s[i]==m，所以ans++。
- ……

作业5455 - 苹果采摘

题目大意

一个长度为n的序列，统计有多少个连续段的和为m。

数据范围$n \leq 100000$。

解析

- 方法2：
- 从小到大枚举j，同时另一个变量i始终保持$s[j] - s[i] \leq m$；
- 当从$j-1$变成j时，一步一步挪动i，直到满足条件。

- j从1到n的过程中，i也至多只会从0到n。
- 时间复杂度$O(n)$。

作业1934 - 外地人

题目大意

输入n条英文单词和方言单词的对应关系，然后进行m次查询，每次查询一个方言单词对应的英文单词。数据范围$n, m \leq 100000$，单词长度不超过10。

解析

- 将n条对应关系按方言单词的字典序排序，然后二分查找。
- 如果按照以前说的“对数组下标排序”的方式，怎么写？

作业1934 - 外地人

题目大意

输入n条英文单词和方言单词的对应关系，然后进行m次查询，每次查询一个方言单词对应的英文单词。数据范围$n, m \leq 100000$，单词长度不超过10。

解析

- 将n条对应关系按方言单词的字典序排序，然后二分查找。
- 如果按照以前说的“对数组下标排序”的方式，怎么写？

```
int N, M;
pair<string, string> dict[MAX_N];
int d[MAX_N];

bool cmp_by_dict(int x, int y) {
    return dict[x] < dict[y];
}
```

```
int main() {
    cin >> N >> M;
    for (int i = 1; i <= N; i ++) {
        cin >> dict[i].second >> dict[i].first;
        d[i] = i;
    }
    sort(d + 1, d + 1 + N, cmp_by_dict);
    for (int i = 1; i <= M; i ++) {
        cin >> dict[0].first;
        int j = lower_bound(d + 1, d + 1 + N, 0, cmp_by_dict) - d;
        if (dict[d[j]].first == dict[0].first) {
            cout << dict[d[j]].second << endl;
        } else {
            cout << "eh" << endl;
        }
    }
    return 0;
}
```

- 对数组下标排序
- 为了对下标lower_bound，需要用到操作一下
- 对下标lower_bound

作业1408 - 求方程的根

题目大意

输入正整数m,n,p,a,b，求方程$m^x+n^x-p^x=0$的根。

数据范围$m,n,p\le 1000$，$a,b\le 100$。

解析

- 函数$f(x)=m^x+n^x-p^x$
- 当$x=0$时，$f(0)=1+1-1=1$；
- 如果$p\le \max(m,n)$，则$f(x)$随x增大而增大，方程在a到b之间没有根；
- 如果$p>\max(m,n)$，则方程在$x>0$时有一个根，怎么计算？

作业1408 - 求方程的根

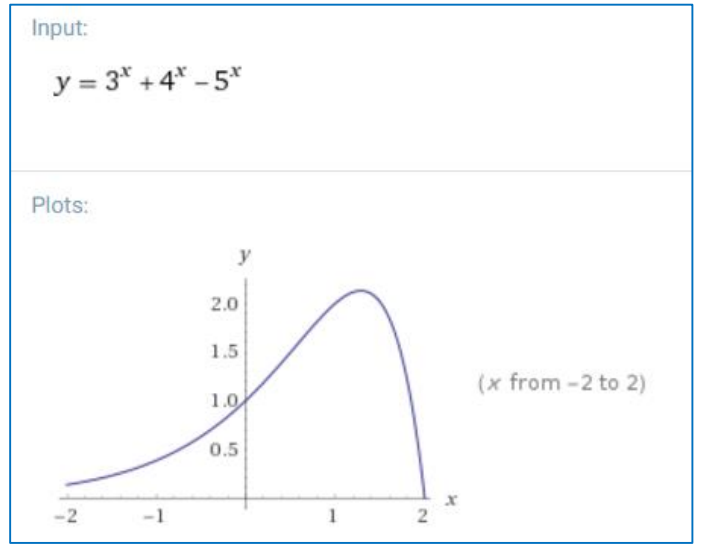
题目大意

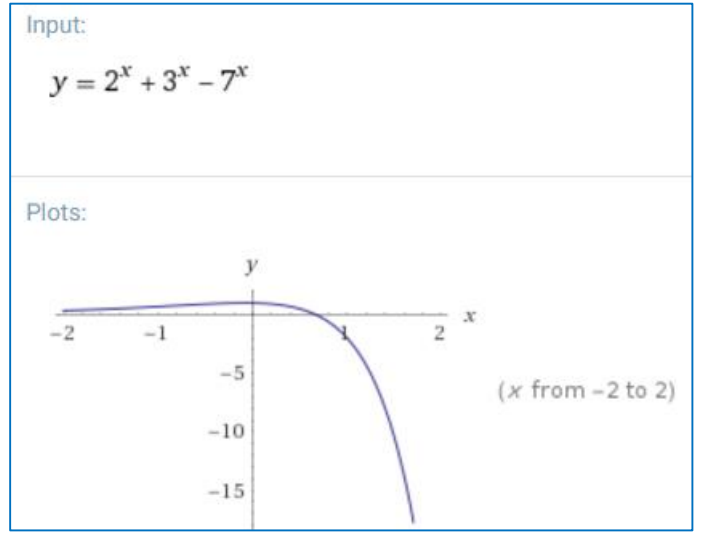
输入正整数m,n,p,a,b，求方程$m^x+n^x-p^x=0$的根。

数据范围$m,n,p\le 1000$，$a,b\le 100$。

解析

- 什么时候可以用二分法求根？
  - $f(x)$在范围$[a,b]$上是单调函数，且$f(a)f(b)<0$
- $f(x)$不单调怎么办？





作业1408 - 求方程的根

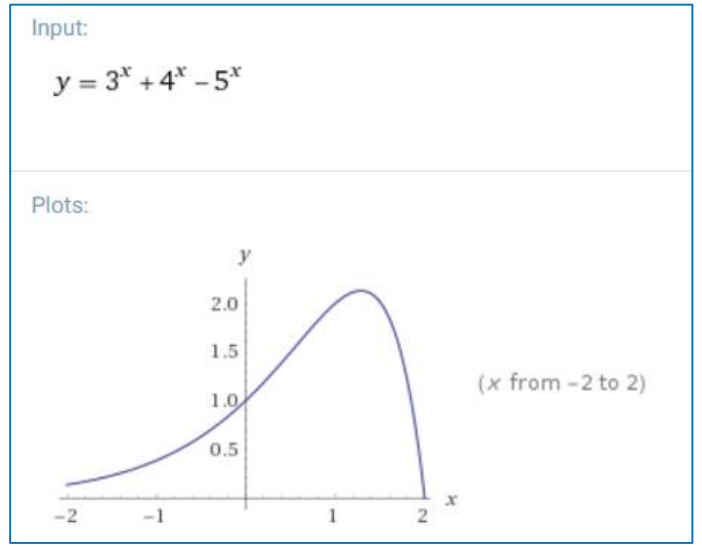
题目大意

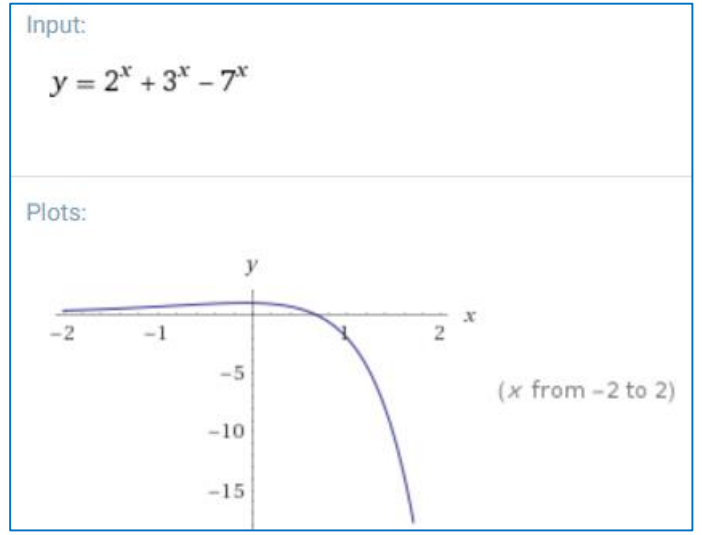
输入正整数m,n,p,a,b，求方程$m^x+n^x-p^x=0$的根。

数据范围$m,n,p\le 1000$，$a,b\le 100$。

解析

- 什么时候可以用二分法求根？
  - $f(x)$在范围$[a,b]$上是单调函数，且$f(a)f(b)<0$
- $f(x)$不单调怎么办？
- 事实上$f(x)$有右边的两种情况：先增后减、单调递减。
  - 严格的说，应该先找到$[a,b]$当中$f(x)$最大值的地方c；
  - 然后分别处理$[a,c]$和$[c,b]$。
- 但是这道题不需要拆分，因为$f(x)$只会有一个根。





作业1408 - 求方程的根

题目大意

输入正整数m,n,p,a,b，求方程$m^x+n^x-p^x=0$的根。

数据范围$m,n,p\le 1000$，$a,b\le 100$。

解析

- 二分法求根：
  - 设当前范围是$[l,r]$，且$f(l)f(r)<0$
  - 设$m=(l+r)/2$，如果$f(l)f(m)<0$，则根在$[l,m]$中，赋值$r=m$；
  - 否则根在$[m,r]$中，赋值$l=m$；
  - 当$r-l<\Delta$的时候停止，其中Δ用来控制二分的精度，这道题设为10^{-11}或10^{-12}为宜。
  - 循环结束后，返回$(l+r)/2$值。

PART 02

分治法

- 讲几个例题

例题

例题1：【NKOJ3590 循环赛日程表】

有n名选手，其中$n = 2^k$。设计一个满足以下要求的比赛日程表：

1. 每个选手必须与其他$n-1$个选手各赛一次；
2. 每个选手一天只能赛一次；
3. 循环赛一共进行$n-1$天。

按此要求，将比赛日程表设计成一个n行$n-1$列的二维表，其中第i行第j列表示和第i个选手在第j天比赛的选手。

数据范围：$1 \le k \le 8$

输入格式：

输入一个整数k。

输出格式：

一个$n \times n$的数字矩阵，表示赛程表，每行数字以空格作为间隔。第1列从上到下分别是1到n，后面$n-1$列是比赛日程表。比赛日程表的安排方式很多，请找出字典序最小的方案。具体样式参见样例数据。

样例输入：

```
3
```

样例输出：

```
1 2 3 4 5 6 7 8
2 1 4 3 6 5 8 7
3 4 1 2 7 8 5 6
4 3 2 1 8 7 6 5
5 6 7 8 1 2 3 4
6 5 8 7 2 1 4 3
7 8 5 6 3 4 1 2
8 7 6 5 4 3 2 1
```

例题

例题1：【NKOJ3590 循环赛日程表】

思考

- 为什么题目要让$n = 2^k$呢？
  - 肯定是要递归，调用2^{k-1}的函数得到2^k时的结果！

例题

例题1：【NKOJ3590 循环赛日程表】

思考

- 为什么题目要让$n = 2^k$呢？
  - 肯定是要递归，调用2^{k-1}的函数得到2^k时的结果！
- 如何递归？找规律。
  - $k = 0$

1

  - $k = 1$

1	2
2	1

  - $k = 2$

1	2	3	4
2	1	4	3
3	4	1	2
4	3	2	1

  - $k = 3$

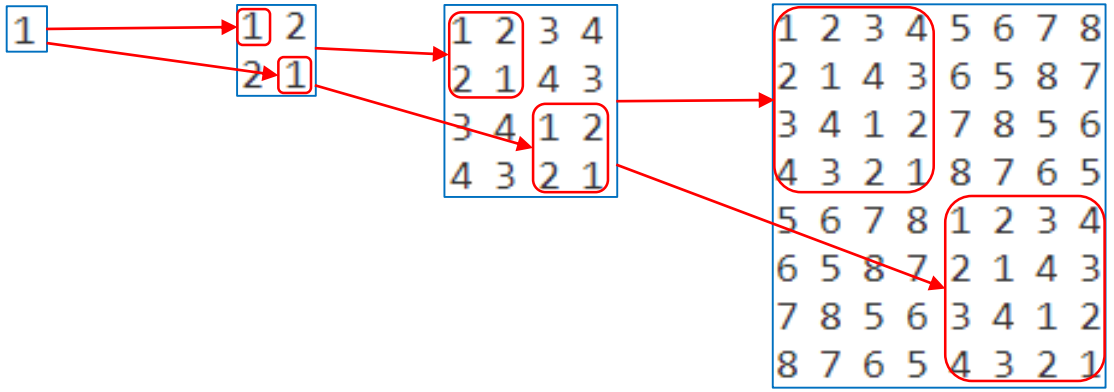
1	2	3	4	5	6	7	8
2	1	4	3	6	5	8	7
3	4	1	2	7	8	5	6
4	3	2	1	8	7	6	5
5	6	7	8	1	2	3	4
6	5	8	7	2	1	4	3
7	8	5	6	3	4	1	2
8	7	6	5	4	3	2	1

例题

例题1：【NKOJ3590 循环赛日程表】

思考

- 为什么题目要让$n = 2^k$呢？
  - 肯定是要递归，调用2^{k-1}的函数得到2^k时的结果！
- 如何递归？找规律。
  - $k = 0$ $k = 1$ $k = 2$ $k = 3$

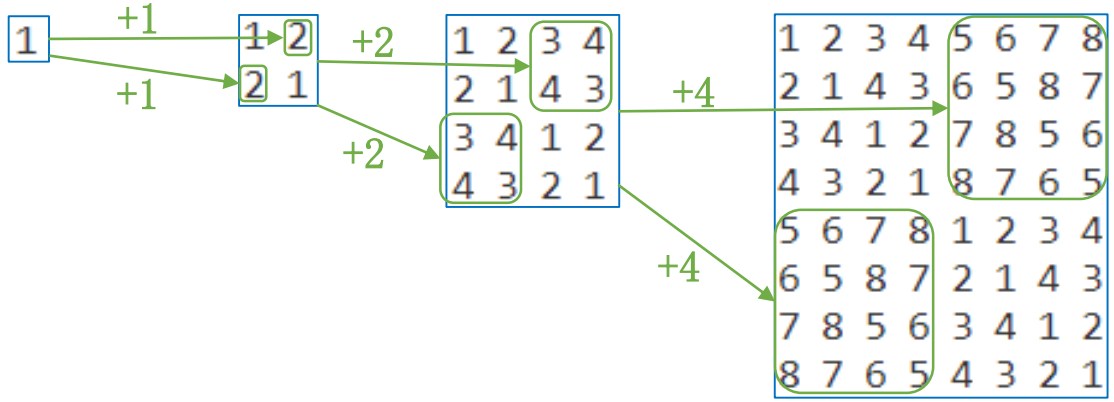


例题

例题1：【NKOJ3590 循环赛日程表】

思考

- 为什么题目要让$n = 2^k$呢？
  - 肯定是要递归，调用2^{k-1}的函数得到2^k时的结果！
- 如何递归？找规律。
  - $k = 0$ $k = 1$ $k = 2$ $k = 3$



例题

例题1：【NKOJ3590 循环赛日程表】

解析

- 分治算法：
- 开一个全局变量二维数组a[][]存储答案；
- 递归函数solve(k, x, y)的作用是，生成一个规模为$2^k \times 2^k$的矩阵，矩阵的左上角在a[][]数组的第x行第y列。

例题

例题1：【NKOJ3590 循环赛日程表】

解析

- 分治算法：
- 开一个全局变量二维数组a[][]存储答案；
- 递归函数solve(k, x, y)的作用是，生成一个规模为$2^k \times 2^k$的矩阵，矩阵的左上角在a[][]数组的第x行第y列。
- 函数实现：
  - 若k=0，即生成1×1的矩阵。令a[x][y]=1，返回。
  - 若k>0，设$m = 2^{k-1}$，先递归调用四次：
    - solve(k - 1, x, y)　　solve(k - 1, x, y + m)
    - solve(k - 1, x + m, y)　　solve(k - 1, x + m, y + m)
  - 然后将左下、右上区域每个数都+m。
- 主函数中调用solve(k, 1, 1)。

例题

例题1：【NKOJ3590 循环赛日程表】

解析

- 分治算法，写成程序：
- 可以用1 << x来计算2^x，其中x必须是非负整数
  - “<<”运算符优先级比“+”、“-”要低；
  - “<<”运算符优先级比“<”、“>”等要高；
  - 右边的程序中“1 << k - 1”计算2^{k-1}。

```
int a[MAX_N][MAX_N];
void solve(int k, int x, int y) {
    if (k == 0) {
        a[x][y] = 1;
    } else {
        int m = 1 << k - 1;
        solve(k - 1, x, y);
        solve(k - 1, x, y + m);
        solve(k - 1, x + m, y);
        solve(k - 1, x + m, y + m);
        for (int i = 0; i < m; i ++) {
            for (int j = 0; j < m; j ++) {
                a[x + i][y + j + m] += m;
                a[x + i + m][y + j] += m;
            }
        }
    }
}
```

例题

例题1：【NKOJ3590 循环赛日程表】

解析

- 分治算法，写成程序：
- 可以用1 << x来计算2^x，其中x必须是非负整数
  - “<<”运算符优先级比“+”、“-”要低；
  - “<<”运算符优先级比“<”、“>”等要高；
  - 右边的程序中“1 << k - 1”计算2^{k-1}。
- 需要手动给左下和右上$+m$，很多数据被多次处理。
- 有没有更“优雅”的写法？

```
int a[MAX_N][MAX_N];
void solve(int k, int x, int y) {
    if (k == 0) {
        a[x][y] = 1;
    } else {
        int m = 1 << k - 1;
        solve(k - 1, x, y);
        solve(k - 1, x, y + m);
        solve(k - 1, x + m, y);
        solve(k - 1, x + m, y + m);
        for (int i = 0; i < m; i ++) {
            for (int j = 0; j < m; j ++) {
                a[x + i][y + j + m] += m;
                a[x + i + m][y + j] += m;
            }
        }
    }
}
```

例题

例题1：【NKOJ3590 循环赛日程表】

解析

- 分治算法+：
- 开一个全局变量二维数组a[][]存储答案；
- 递归函数solve(k, x, y, **v**)的作用是，生成一个规模为$2^k \times 2^k$的矩阵，矩阵的左上角在a[][]数组的第x行第y列，**且a[x][y]=v**。
- 函数实现：
  - 若k=0，即生成1×1的矩阵。令a[x][y]=**v**，返回。
  - 若k>0，设$m = 2^{k-1}$，先递归调用四次：
    - solve(k - 1, x, y, **v**)　　solve(k - 1, x, y + m, **v + m**)
    - solve(k - 1, x + m, y, **v + m**)　　solve(k - 1, x + m, y + m, **v**)
- 主函数中调用solve(k, 1, 1, **1**)。

例题

例题1：【NKOJ3590 循环赛日程表】

解析

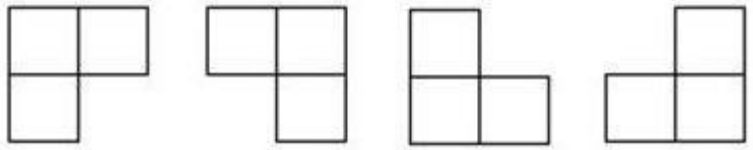
- 分治算法+，写成程序：
- 这样就避免了对一些区域的多次处理。

```
int a[MAX_N][MAX_N];
void solve(int k, int x, int y, int v) {
    if (k == 0) {
        a[x][y] = v;
    } else {
        int m = 1 << k - 1;
        solve(k - 1, x, y, v);
        solve(k - 1, x, y + m, v + m);
        solve(k - 1, x + m, y, v + m);
        solve(k - 1, x + m, y + m, v);
    }
}
```

例题

例题2：【NKOJ2307 棋盘覆盖问题】

在一个$2^k \times 2^k (1 \le k \le 10)$个方格组成的棋盘中，恰有一个方格是黑色的，其他方格都是白色。你的任务是要用图示的由三个方格构成的4种不同形态的L形骨牌覆盖所有的白色方格，且任何一个白色方格不能同时被两次或多个次覆盖。



输入格式：

第一行一个整数k，表示棋盘的大小为$2^k \times 2^k$。
第二行是两个整数，代表特殊方格所在行号和列号。

输出格式：

一个以空格为间隔的$2^k \times 2^k$数字矩阵，表示一种可行的覆盖方案。同一块骨牌用相同的数字表示，用数字0表示黑色方格。按矩阵从左到右，从上到下的顺序将数字由小到大输出，详情见样例。

样例输入1：
```
1
1 1
```

样例输出1：
```
0 1
1 1
```

样例输入2：
```
2
2 1
```

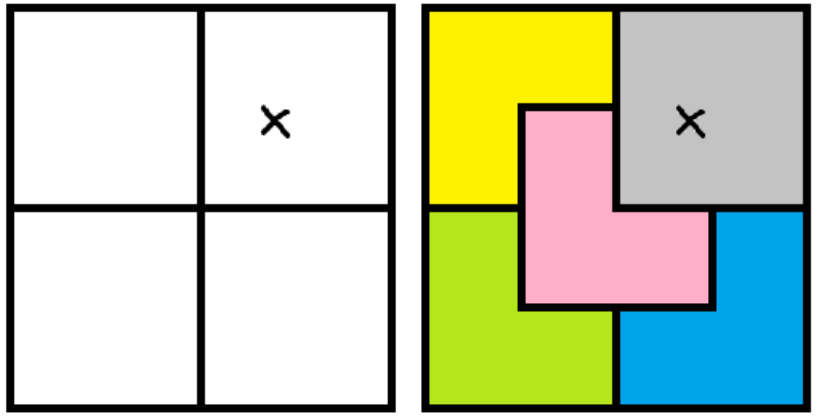
样例输出2：
```
1 1 2 2
0 1 3 2
4 3 3 5
4 4 5 5
```

样例输入3：
```
3
3 2
```

样例输出3：
（写不下，去OJ上看）

例题

例题2：【NKOJ2307 棋盘覆盖问题】

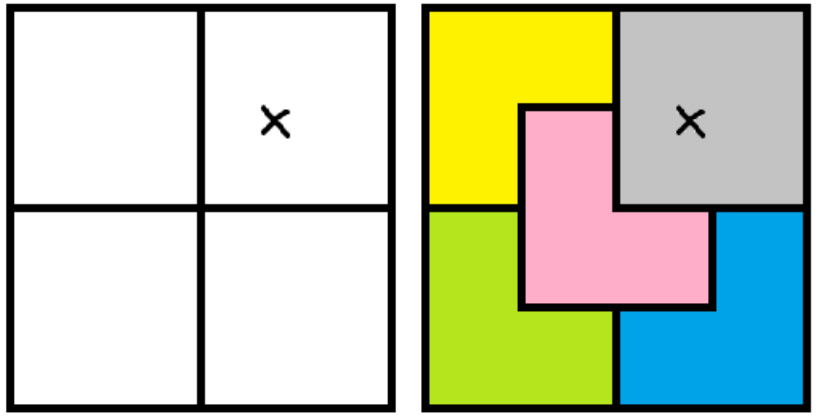


解析

- 如何分治？
- 如右图所示，将$2^k \times 2^k$的棋盘划分为四个区域。
- 不含黑格的三个区域组成规模为2^k的L形，
- $k > 1$时可以划分为四个规模为2^{k-1}的L形，递归处理（L形）；
- 含有黑格的一个区域，递归处理（正方形）。

例题

例题2：【NKOJ2307 棋盘覆盖问题】

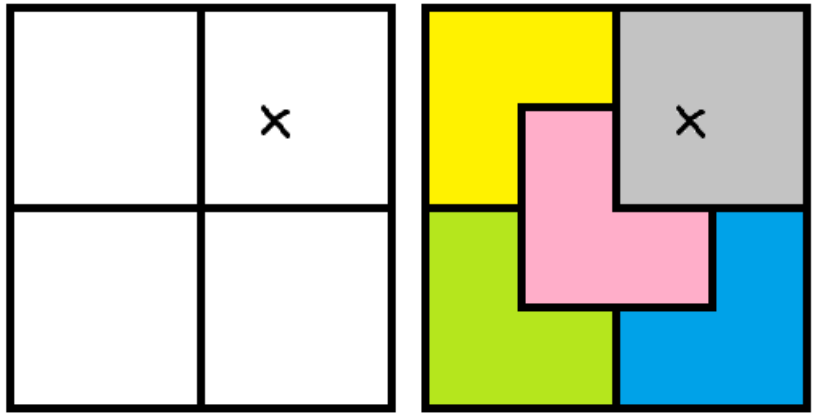


解析

- 怎么写程序？
- 方法1：分开写正方形递归和L形递归函数。

例题

例题2：【NKOJ2307 棋盘覆盖问题】

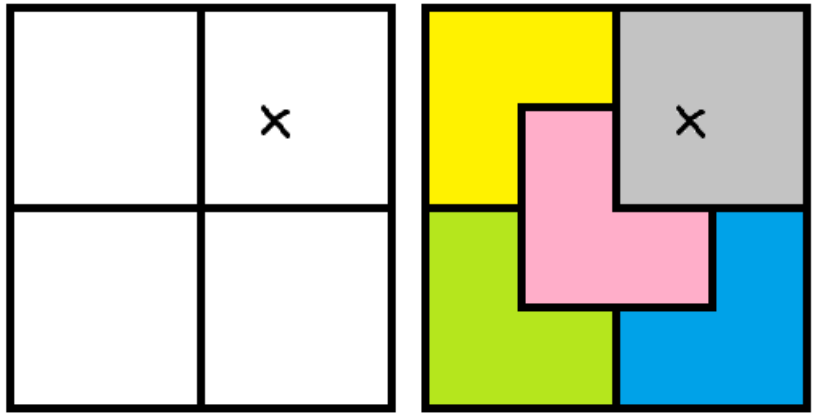


解析

- 怎么写程序？
- 方法1：分开写正方形递归和L形递归函数。
- 正方形递归的参数：
  - 参数k，表示当前的问题规模；
  - 参数x, y，表示当前正方形区域左上角的坐标；

例题

例题2：【NKOJ2307 棋盘覆盖问题】

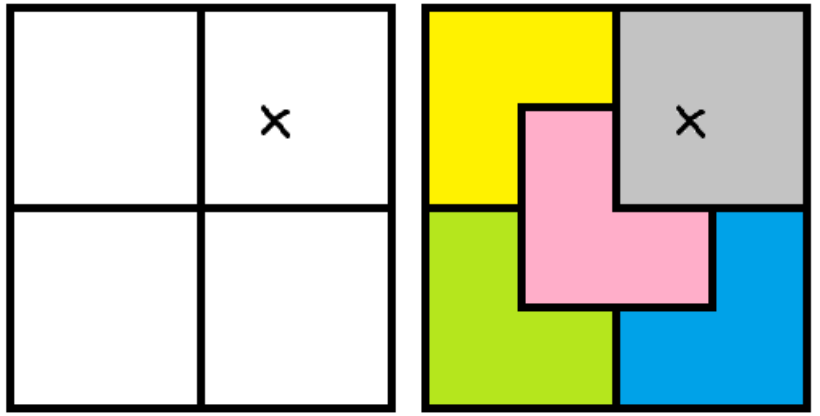


解析

- 怎么写程序？
- 方法1：分开写正方形递归和L形递归函数。
- 正方形递归的参数：
  - 参数k，表示当前的问题规模；
  - 参数x, y，表示当前正方形区域左上角的坐标；
- 正方形递归的过程：
  - 确定黑格在4个区域中的哪一个；
  - 调用L形递归函数处理规模为k的L形；
  - 若k>1，调用正方形递归函数处理规模为k-1的正方形。

例题

例题2：【NKOJ2307 棋盘覆盖问题】

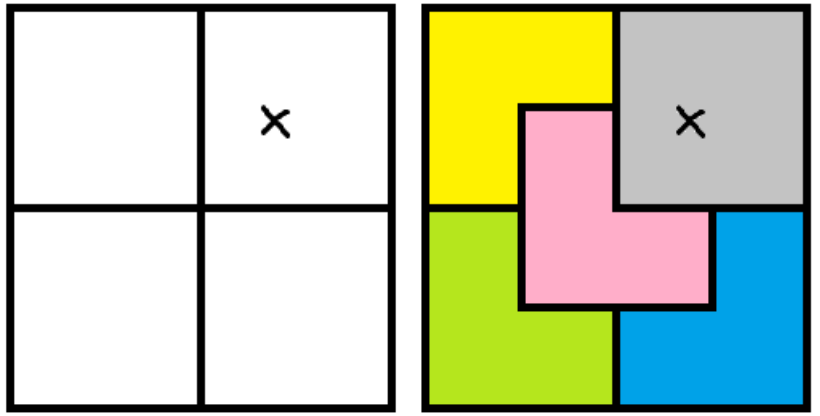


解析

- 怎么写程序？
- 方法1：分开写正方形递归和L形递归函数。
- L形递归的参数：
  - 参数k，表示当前的问题规模；
  - 参数x,y，表示当前正方形区域左上角的坐标；
  - 参数dir，表示L形的方向；

例题

例题2：【NKOJ2307 棋盘覆盖问题】

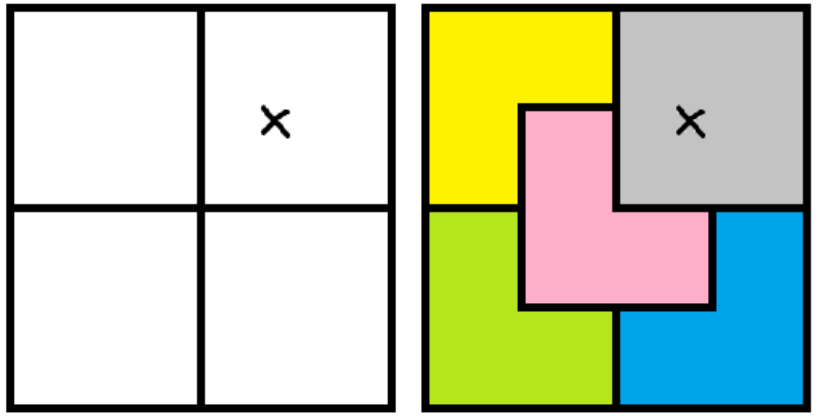


解析

- 怎么写程序？
- 方法1：分开写正方形递归和L形递归函数。
- L形递归的参数：
  - 参数k，表示当前的问题规模；
  - 参数x,y，表示当前正方形区域左上角的坐标；
  - 参数dir，表示L形的方向；
- L形递归的过程：
  - 如果k=1，全局变量id增加1，填入数据；
  - 如果k>1，分别调用四次L形递归，处理每个L形，周围的L形方向固定，中间的L形方向与当前一致。

例题

例题2：【NKOJ2307 棋盘覆盖问题】



解析

- 怎么写程序？
- 方法1：分开写正方形递归和L形递归函数。
- 递归过程中，按访问顺序临时编号，
- 最后还要再处理编号，按从上到下、从左到右的顺序。
- 开一个int数组，记录每个临时编号的最终编号。
- 按从上到下、从左到右输出，遇到没有分配最终编号的临时编号，就分配个新编号。

例题

例题2：【NKOJ2307 棋盘覆盖问题】

解析

- 怎么写程序？
- 方法2：写一个递归函数

例题

例题2：【NKOJ2307 棋盘覆盖问题】

解析

- 怎么写程序？
- 方法2：写一个递归函数
- 用同一个递归函数解决L形和正方形的问题
- L形需要参数dir表示方向，那么正方形时的dir规定为特殊的值。

例题

例题2：【NKOJ2307 棋盘覆盖问题】

解析

- 怎么写程序？
- 方法2：写一个递归函数
- 用同一个递归函数解决L形和正方形的问题
- L形需要参数dir表示方向，那么正方形时的dir规定为特殊的值。

```
int K, N, X, Y, id;
int a[MAX_N][MAX_N];
int new_id[MAX_N * MAX_N];

void solve(int k, int x, int y, int dir) {
    int m = 1 << k - 1;
    int d = dir >= 0 ? dir : (X >= x + m) * 2 + (Y >= y + m);
    if (k == 1) {
        ++ id;
        if (d != 0) a[x][y] = id;
        if (d != 1) a[x][y + 1] = id;
        if (d != 2) a[x + 1][y] = id;
        if (d != 3) a[x + 1][y + 1] = id;
    } else {
        if (d != 0) solve(k - 1, x, y, 3);
        if (d != 1) solve(k - 1, x, y + m, 2);
        if (d != 2) solve(k - 1, x + m, y, 1);
        if (d != 3) solve(k - 1, x + m, y + m, 0);
        solve(k - 1, x + m / 2, y + m / 2, d);
        if (dir < 0) {
            if (d == 0) solve(k - 1, x, y, -1);
            if (d == 1) solve(k - 1, x, y + m, -1);
            if (d == 2) solve(k - 1, x + m, y, -1);
            if (d == 3) solve(k - 1, x + m, y + m, -1);
        }
    }
}
```

例题

例题2：【NKOJ2307 棋盘覆盖问题】

解析

- 怎么写程序？
- 方法2：写一个递归函数
- 用同一个递归函数解决L形和正方形的问题
- L形需要参数dir表示方向，那么正方形时的dir规定为特殊的值。

```
int main() {
    scanf("%d%d%d", &K, &X, &Y);
    N = 1 << K;
    id = 0;
    solve(K, 1, 1, -1);
    id = 0;
    for (int i = 1; i <= N; i ++) {
        for (int j = 1; j <= N; j ++) {
            if (a[i][j] && !new_id[a[i][j]]) {
                new_id[a[i][j]] = ++ id;
            }
            printf("%d%c", new_id[a[i][j]], j < N ? ' ' : '\n');
        }
    }
    return 0;
}
```

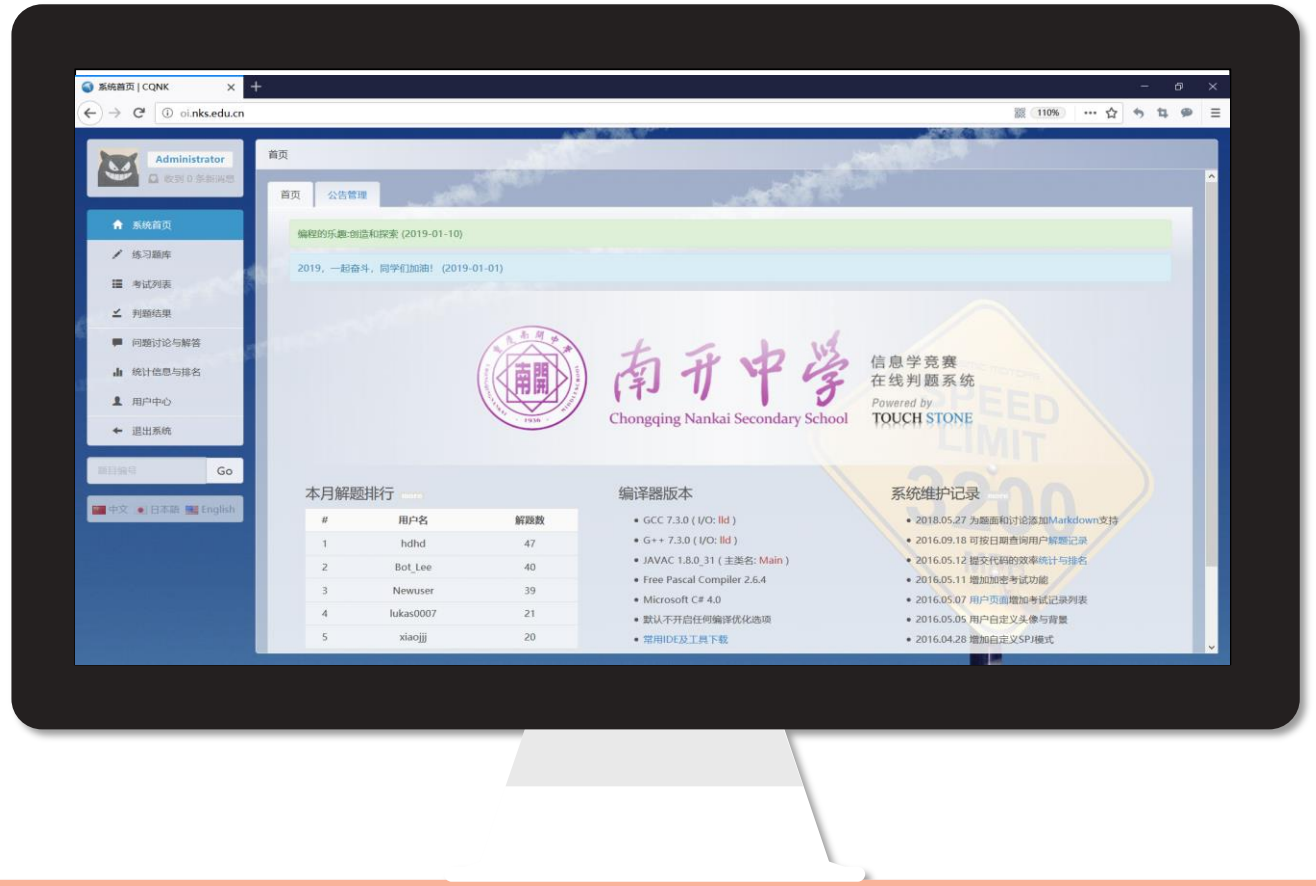
```
int K, N, X, Y, id;
int a[MAX_N][MAX_N];
int new_id[MAX_N * MAX_N];

void solve(int k, int x, int y, int dir) {
    int m = 1 << k - 1;
    int d = dir >= 0 ? dir : (X >= x + m) * 2 + (Y >= y + m);
    if (k == 1) {
        ++ id;
        if (d != 0) a[x][y] = id;
        if (d != 1) a[x][y + 1] = id;
        if (d != 2) a[x + 1][y] = id;
        if (d != 3) a[x + 1][y + 1] = id;
    } else {
        if (d != 0) solve(k - 1, x, y, 3);
        if (d != 1) solve(k - 1, x, y + m, 2);
        if (d != 2) solve(k - 1, x + m, y, 1);
        if (d != 3) solve(k - 1, x + m, y + m, 0);
        solve(k - 1, x + m / 2, y + m / 2, d);
        if (dir < 0) {
            if (d == 0) solve(k - 1, x, y, -1);
            if (d == 1) solve(k - 1, x, y + m, -1);
            if (d == 2) solve(k - 1, x + m, y, -1);
            if (d == 3) solve(k - 1, x + m, y + m, -1);
        }
    }
}
```

课后练习

习题

NKOJ3590	循环赛日程表
NKOJ2307	棋盘覆盖问题
NKOJ1407	地毯填补问题
NKOJ3520	分形之城


系统首页 | CQNK
oi.nks.edu.cn
Administrator
首页
公告管理
编程的乐趣:创造和探索 (2019-01-10)
2019，一起奋斗，同学们加油！ (2019-01-01)
南开中学
Chongqing Nankai Secondary School
信息学竞赛
在线判题系统
Powered by
TOUCH STONE
本月解题排行
编译器版本
系统维护记录


Noi


oi.nks.edu.cn