**《算法竞赛入门到进阶》：勘误与改进**

出版社：清华大学出版社

作者：罗勇军 郭卫斌

联系方式：QQ 15512356，QQ群：567554289

**最近更新**：2020.3，第 5 次印刷。

在扉页中可以查到是第几次印刷。

本书在多次重印过程中，进行了持续勘误和改进。本文记录了这些细节，包括两方面的内容：

（1）勘误。本书的印刷或内容错误，每次新印刷时，会修改新发现的问题。

（2）新内容。增加的新内容。

本文下载地址：

（1）https://github.com/luoyongjun999/code

（2）QQ 群：567554289 群文件

1、第 2 次印刷（2019.8）

以下已经改正：

（1）55 页，把(c,r)改成(r,c)



（2）294 页，把 k 改成 n。

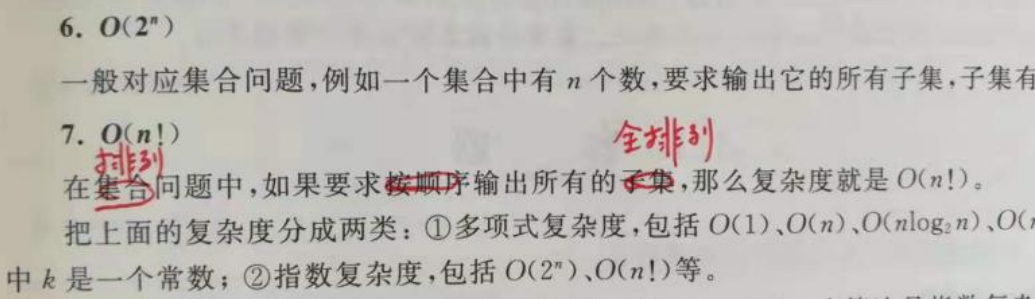


2、第3次印刷 （2019.9）

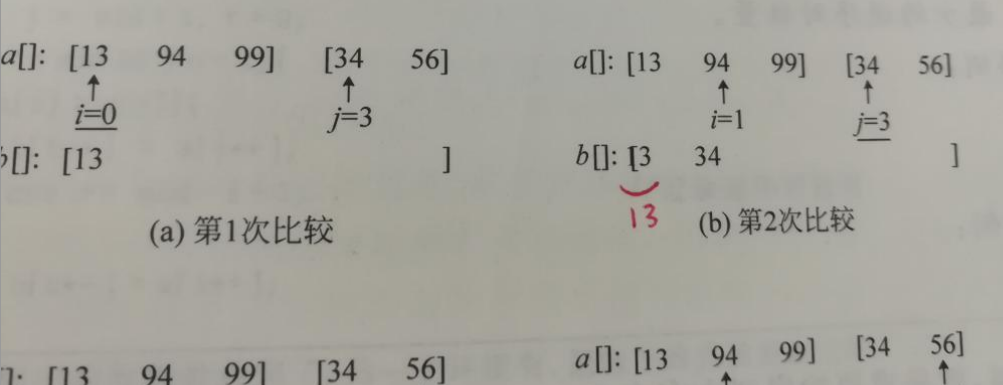
改进：前面2印是120分钟视频；从第3印开始，增加到350分钟视频。

下面是修改：

（1）23 页，修改一句。



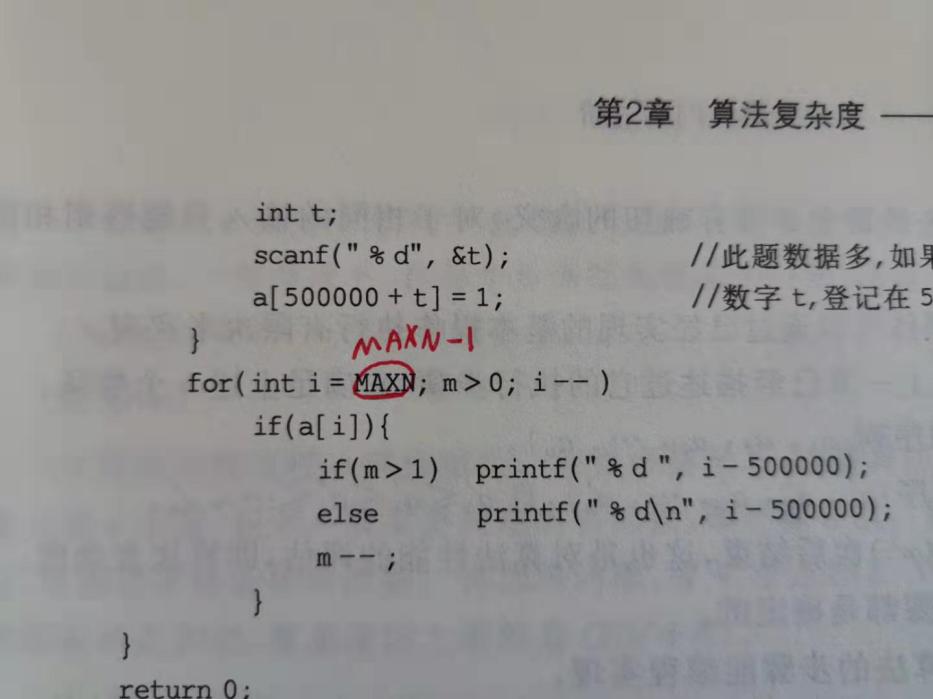
（3）109 页，应该是 13。



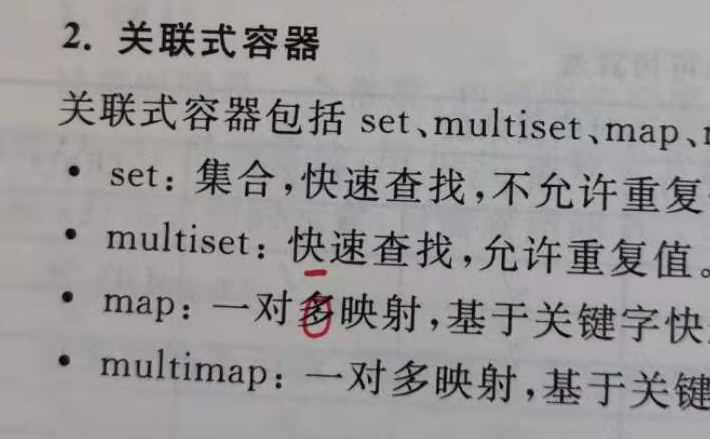
1. 第4次印刷（2019.11）

以下内容已经修改：

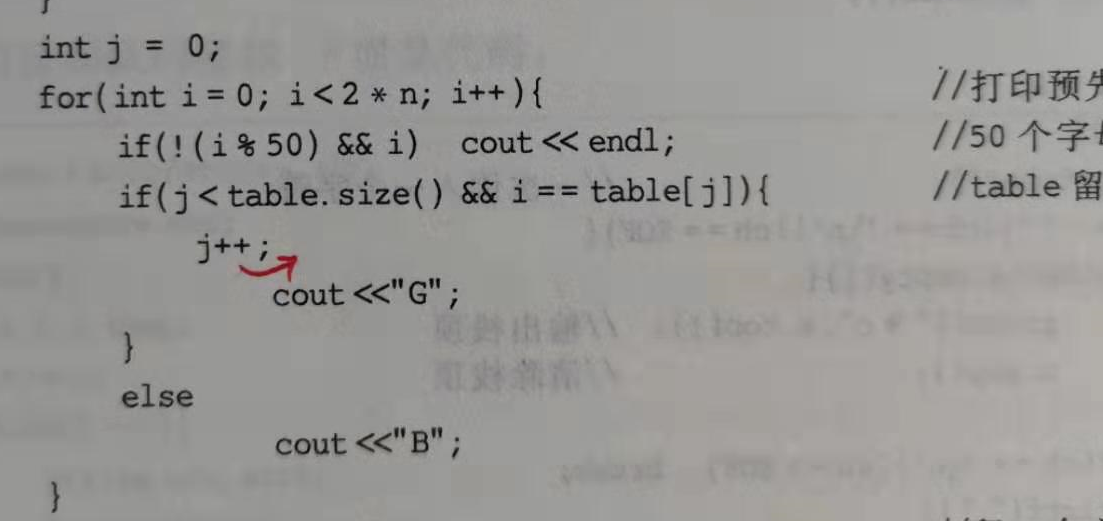
（1）21页。把 i=MAXN改为 i=MAXN-1



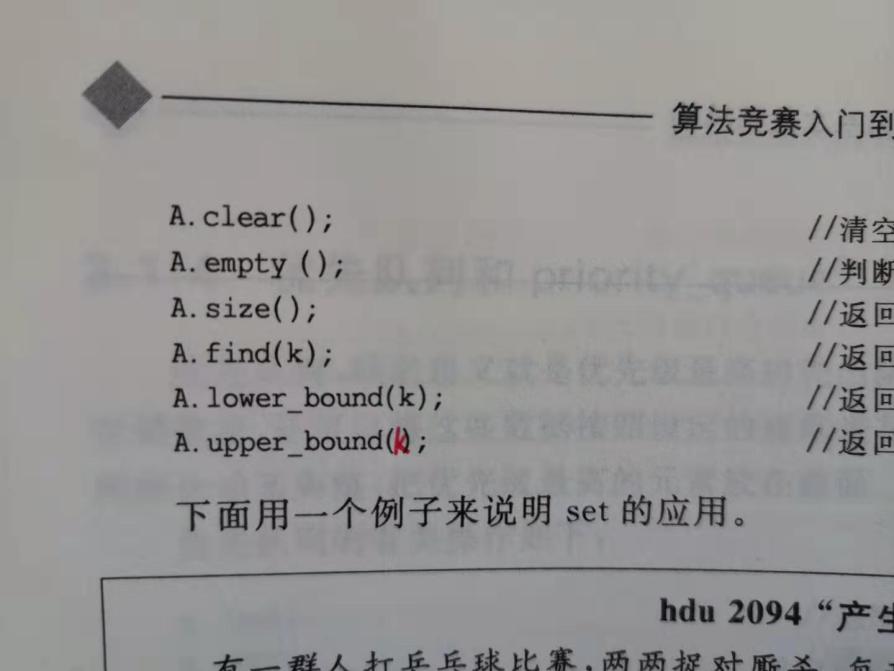
（2）24 页。把“map：一对多映射”改为“map：一对一映射”。



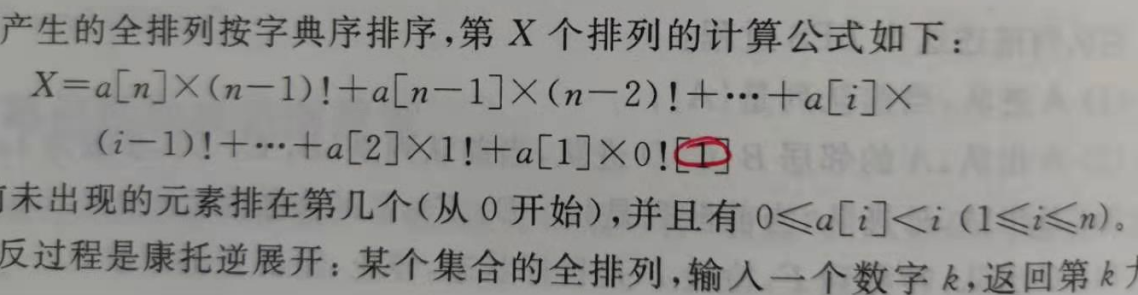
（3）27页。j++ 没对齐，往后退几格对齐。



（4）32页。第6行括号里面少一个 k



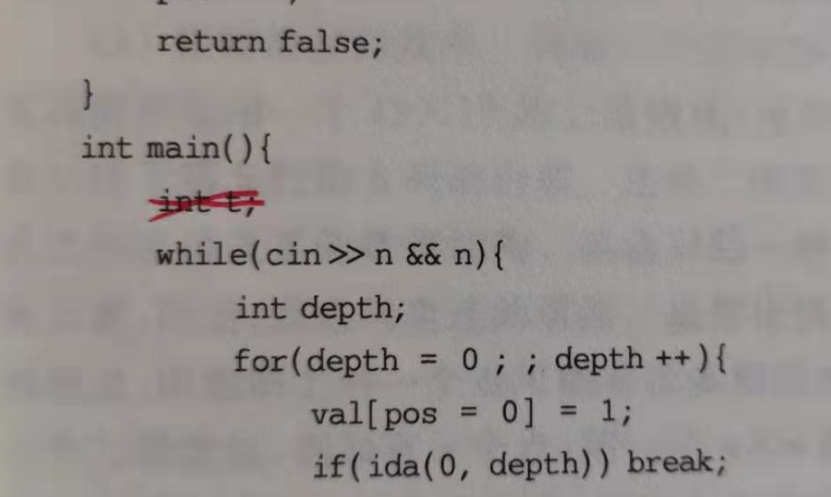
（5）48页。删除公式末尾的 [1]



（6）48页。把倒数第2行的“a[i]为当前未出现的元素排在第几个”改为“a[i]表示原数的第i位在当前未出现的元素中排在第几个”



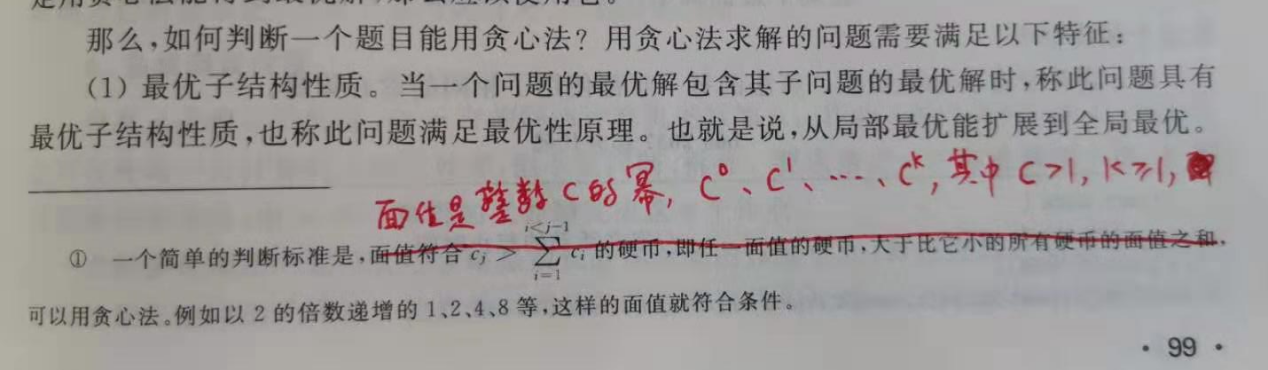
（7）59页。删除 int t;



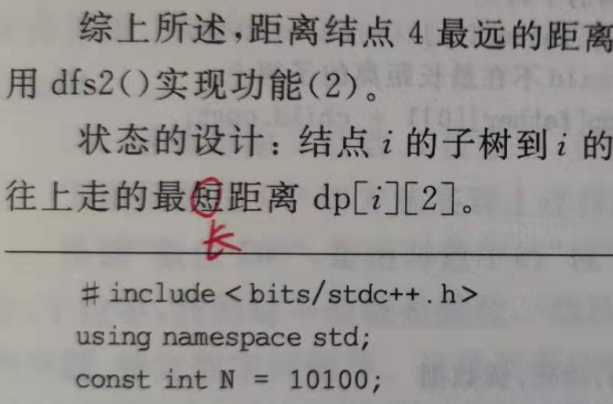
（8）99页。

最下面的脚注，原来的说法是错误的。现在改为：

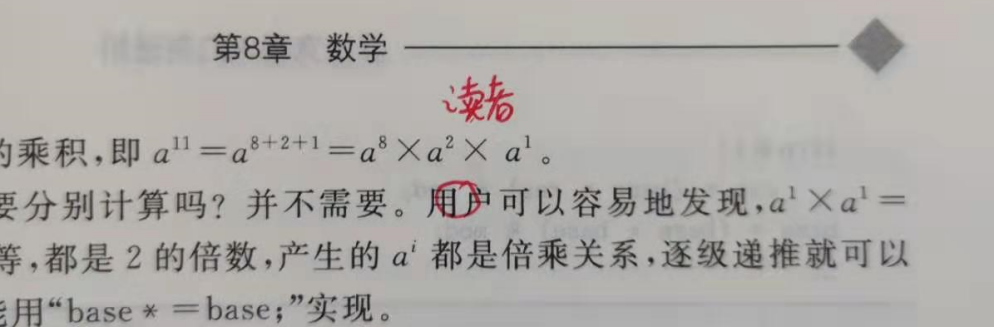
一个简单的判断标准是，面值是整数c的幂，c0、c1、... 、ck，其中c>1，k≥1，可以用贪心法。例如以2的倍数递增的1、2、4、8等，这样的面值就符合条件。



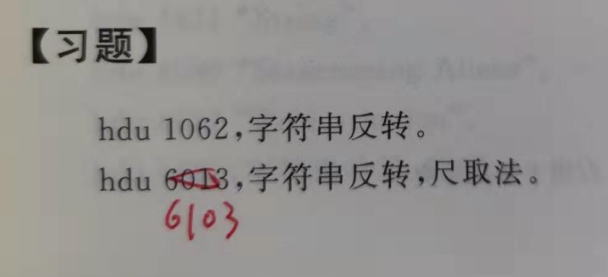
（9）143页。把“最短距离”改为“最长距离”。



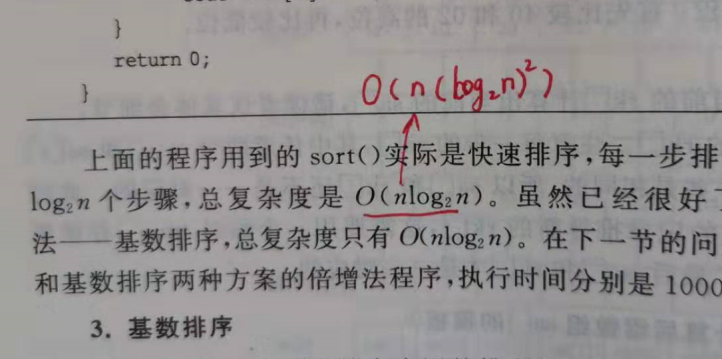
（10）157页，第2行，“用户”改为“读者”



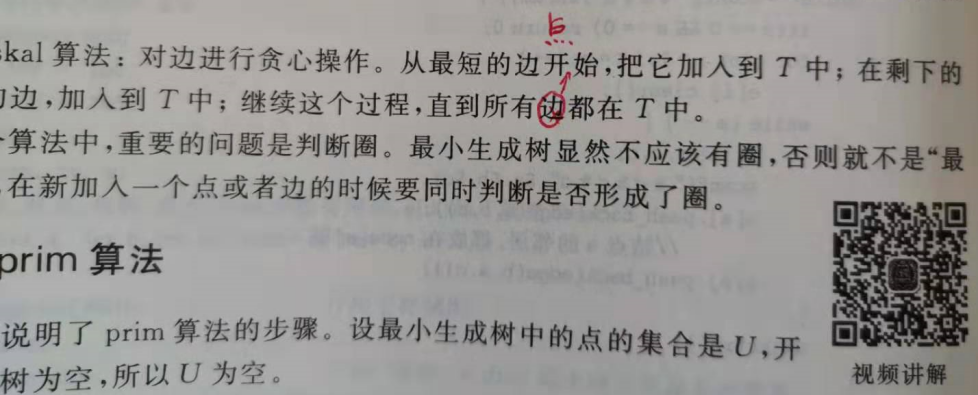
（11）193页。把6013改为6103



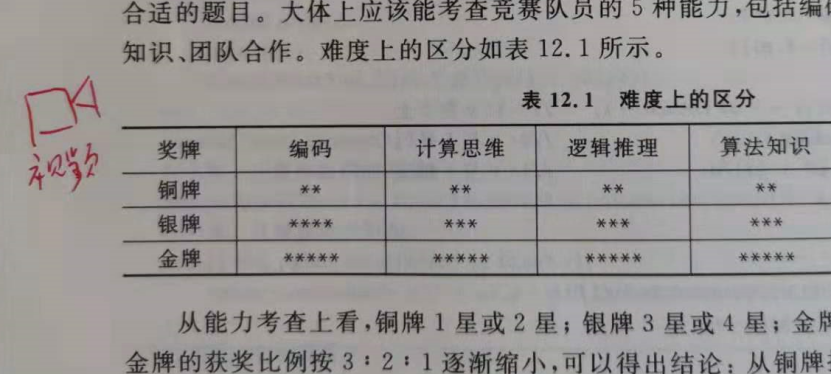
（12）210 页。改为：总复杂度是 *O*(*n*(log*2n*)*2*)



（13）254页。kruskal算法，把“所有边都在T中”，改为“所有点都在T中”

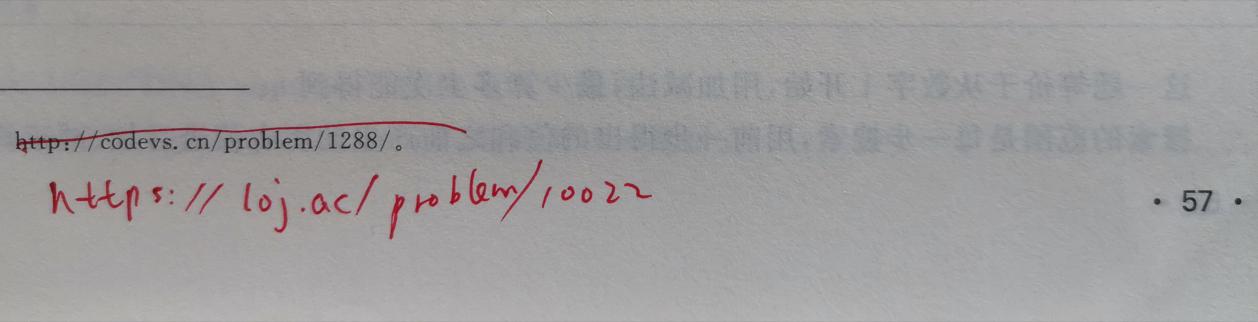


（14）316 页。加一个视频：“第12章-表 12.1.mp4”。



（15）57页，脚注的 http://codevs.cn/problem/1288 连不上了，改为：

https://loj.ac/problem/10022

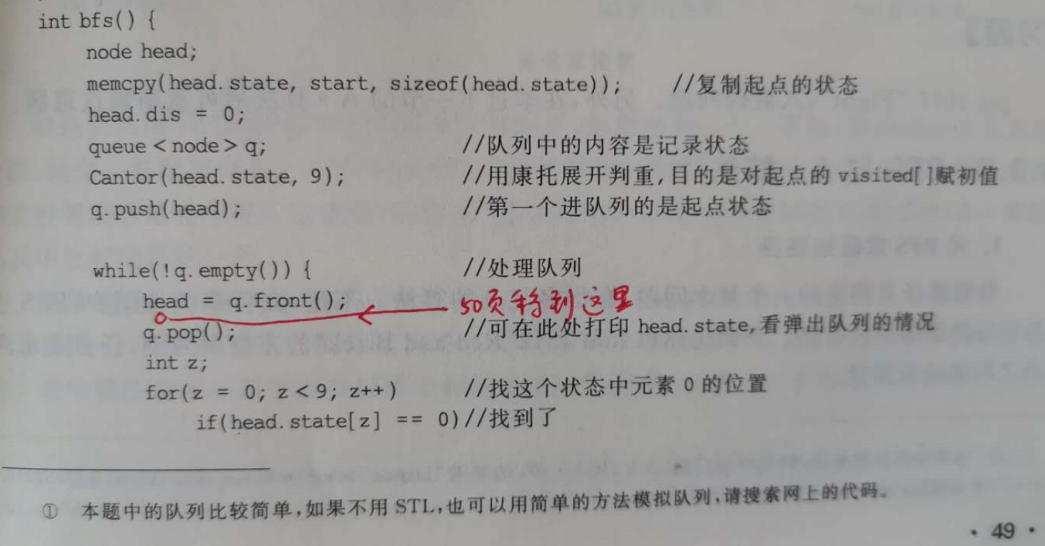
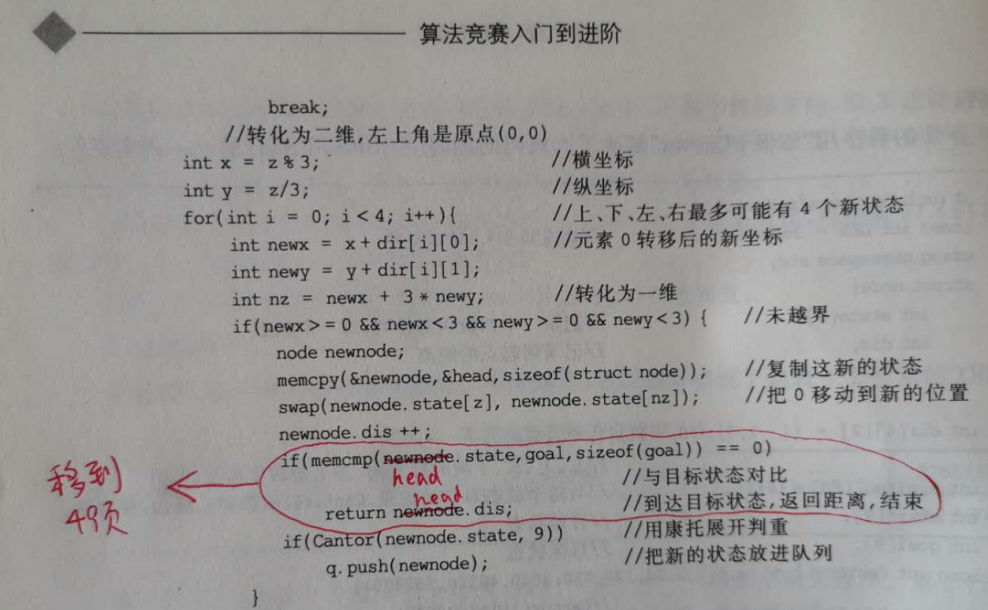


4、第5次 印刷

（1）把50页的这2句代码移动到49页，并把其中的newnode改为head

if(memcmp(head.state,goal,sizeof(goal)) == 0) //与目标状态对比

return head.dis; //到达目标状态，返回距离，结束

的判断放在了后面，当初始棋盘和目标棋盘一样时，输出了2，但是  
应该输出0。

修改：49页的代码把 if（memcmp）.....这句挪到取队头元素的下面就好了，当然里面的 newnode.state要换成head.state

int bfs() {

node head;

memcpy(head.state, start, sizeof(head.state)); //复制起点的状态

head.dis = 0;

queue <node> q; //队列中放状态

Cantor(head.state, 9); //用康托展开判重，目的是对起点的visited[]赋初值

q.push(head); //第一个进队列的是起点状态

while(!q.empty()) { //处理队列

head = q.front();

if(memcmp(head.state,goal,sizeof(goal)) == 0) //与目标状态对比

return head.dis; //到达目标状态，返回距离，结束

q.pop(); //可在此处打印head.state，看弹出队列的情况

int z;

for(z = 0; z < 9; z++) //找这个状态中元素0的位置

if(head.state[z] == 0) //找到了

break;

//转化为二维，左上角是原点(0, 0)。

int x = z%3; //横坐标

int y = z/3; //纵坐标

for(int i = 0; i < 4; i++){ //上、下、左、右最多可能有4个新状态

int newx = x+dir[i][0]; //元素0转移后的新坐标

int newy = y+dir[i][1];

int nz = newx + 3\*newy; //转化为一维

if(newx>=0 && newx<3 && newy>=0 && newy<3) {//未越界

node newnode;

memcpy(&newnode,&head,sizeof(struct node));//复制这新的状态

swap(newnode.state[z], newnode.state[nz]);//把0移动到新的位置

newnode.dis ++;

if(Cantor(newnode.state, 9)) //用康托展开判重

q.push(newnode); //把新的状态放进队列

}

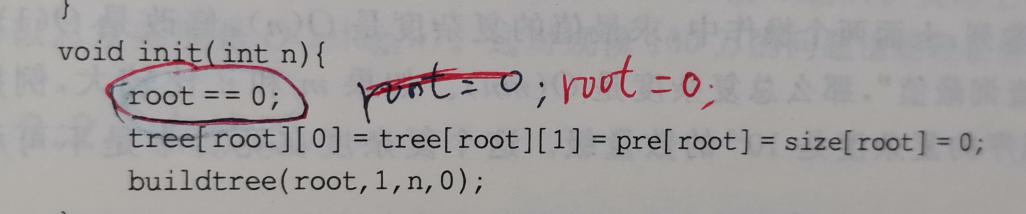
}

}

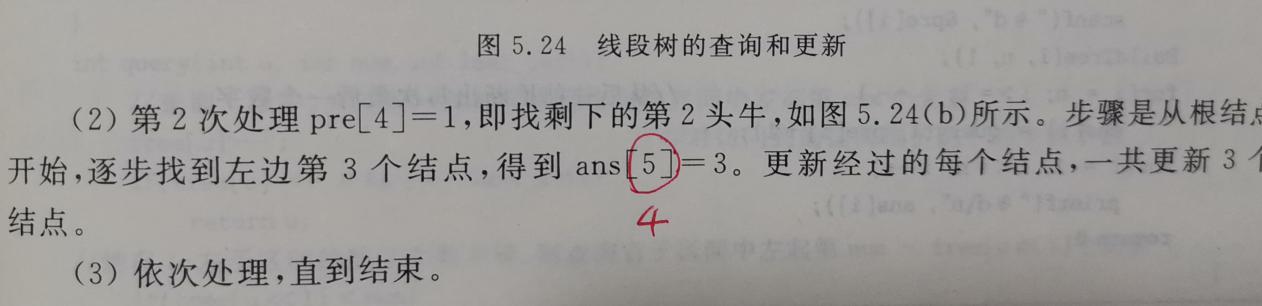
return -1; //没找到

}

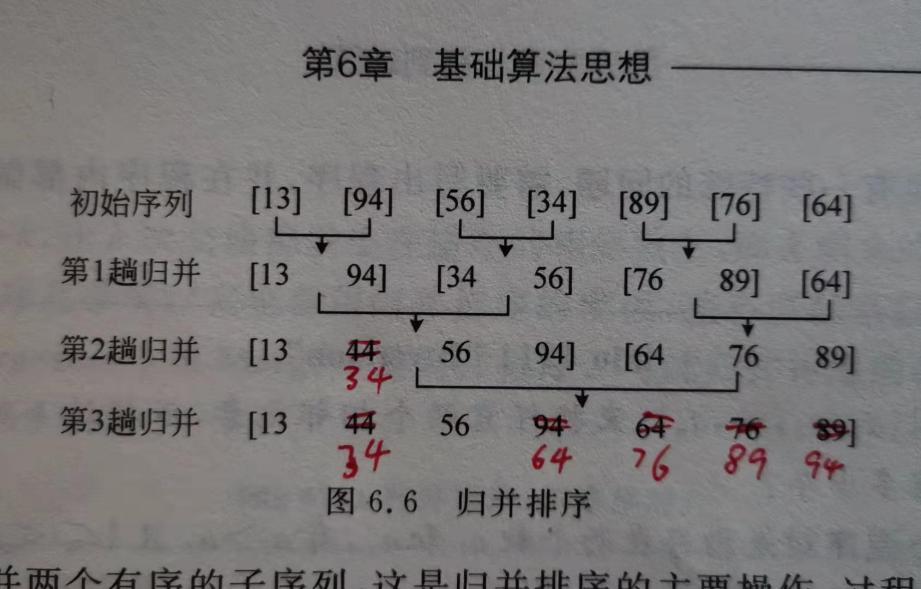
（2）83页，init()中root==0,应该为root=0



（3）87页，ans[5]改成ans[4]



（4）109页，图6.6，错了很多数字。



（5）在原书p145 hdu2089题的代码中，这一段：

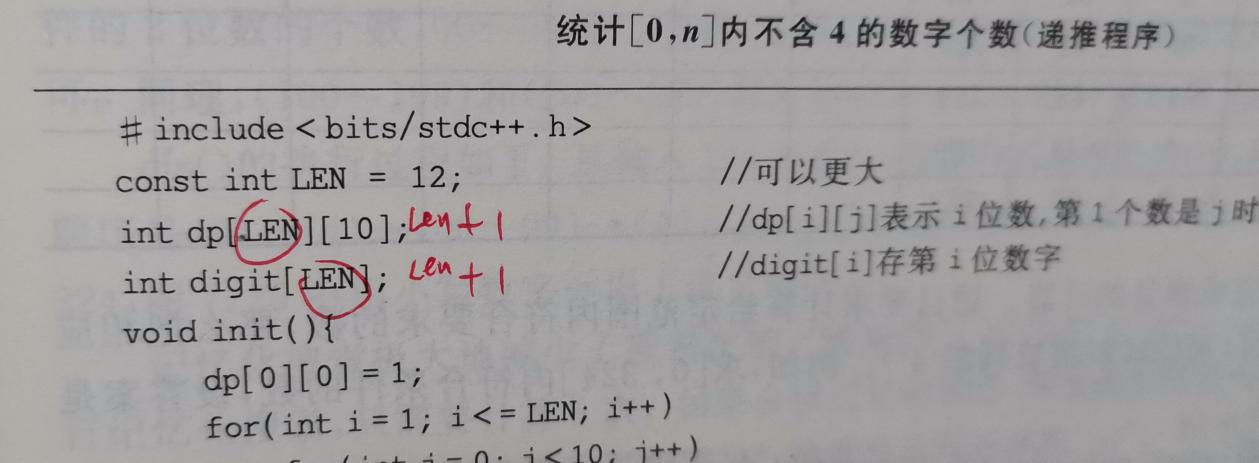
int dp[LEN][10];

int digit[LEN];

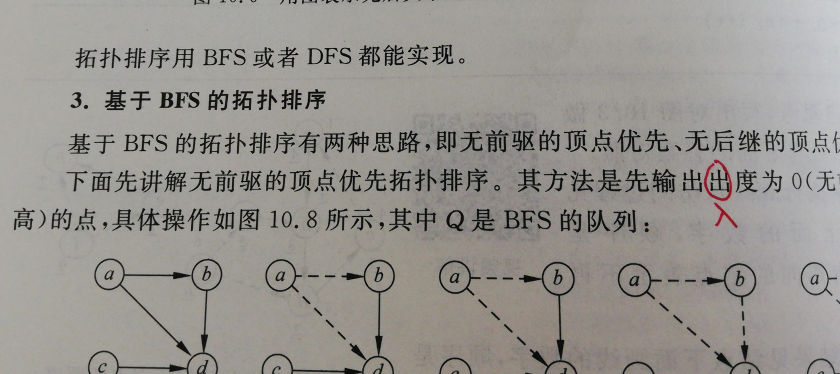
改为：

int dp[LEN+1][10];

int digit[LEN+1];

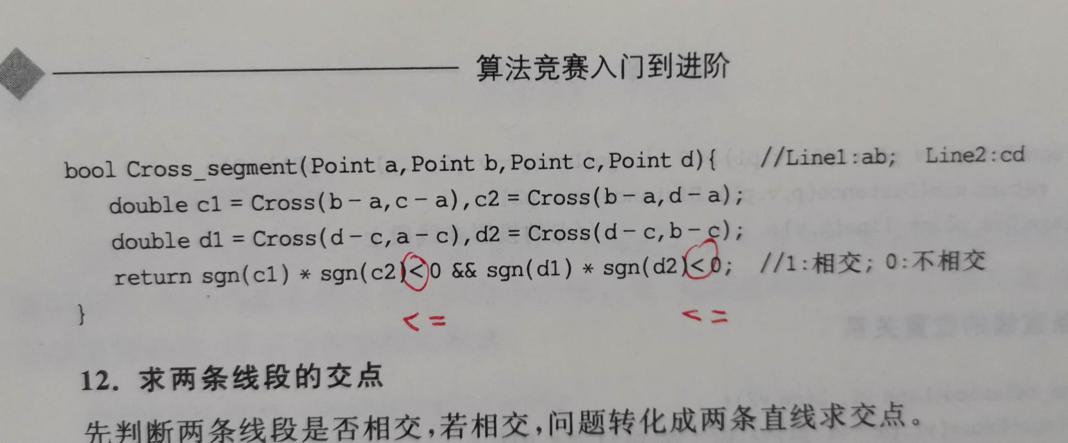


（6）220页。应该是“先输出入度为0”



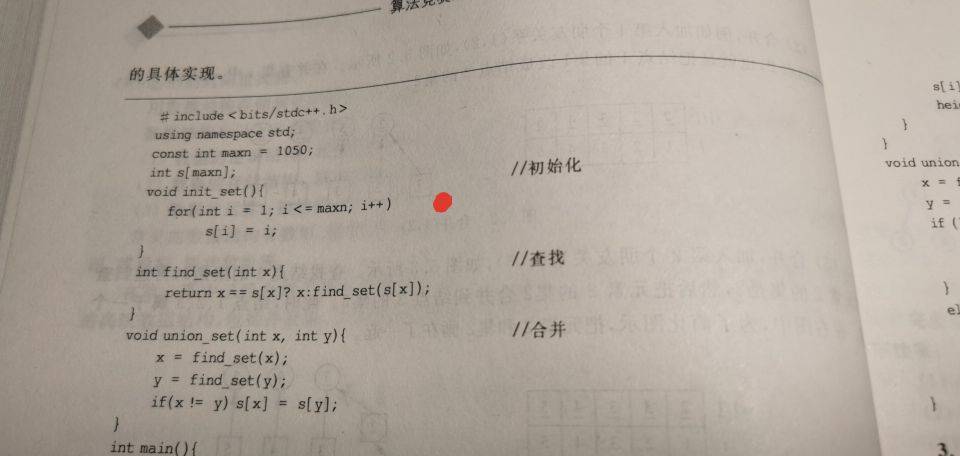
(7) 279页那个11：判断两个线段是否相交；不应该只判断两个线段都满足一个线段的两端在令一条线段的两端，如果两条线段相交的话，还有一种情况为一个线段的端点在另一条线段上。

把280页第4行的<0 >0改为 <=0 >=0



5 第6次印刷

（1）64页，好几个地方。应该改为s[maxn+1]



西南交大 文源 1825491283

我加了一句 这个 if(x==y) return;                //同一组不做修改，只进行set内调整

这个 代码 有些bug

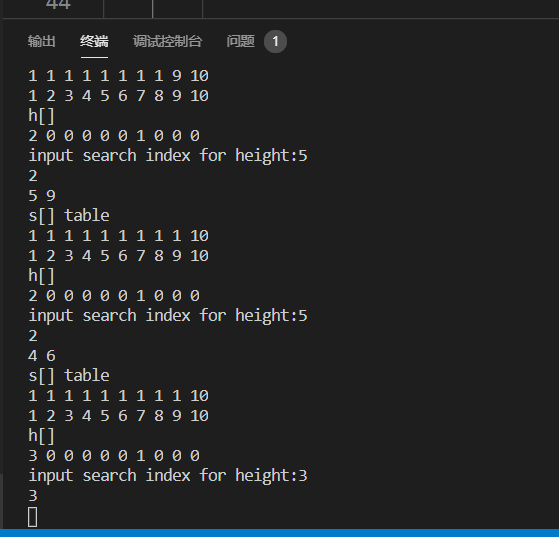
同时考虑 路径压缩和 合并优化之后

如果 输入 不同set的元素，虽然h[]数组不是完全对应

但是 没有bug

但是 一旦输入同一组的元素 高度就会无限递增

以后合并就会混乱



（2）113页，快速排序，加一个注释：

测试数据故意卡快速排序的极端情况，例如测试数据是100000个完全一样的数字，按本文的写法会超时。

例如洛谷P1177题<https://www.luogu.com.cn/problem/P1177>

参考这篇网文，很好地回答了这个问题。

<https://www.luogu.com.cn/blog/adamding/solution-p1177>

（3）181页

公式的最后一行少了一个dp。见下面图片中第2行。

