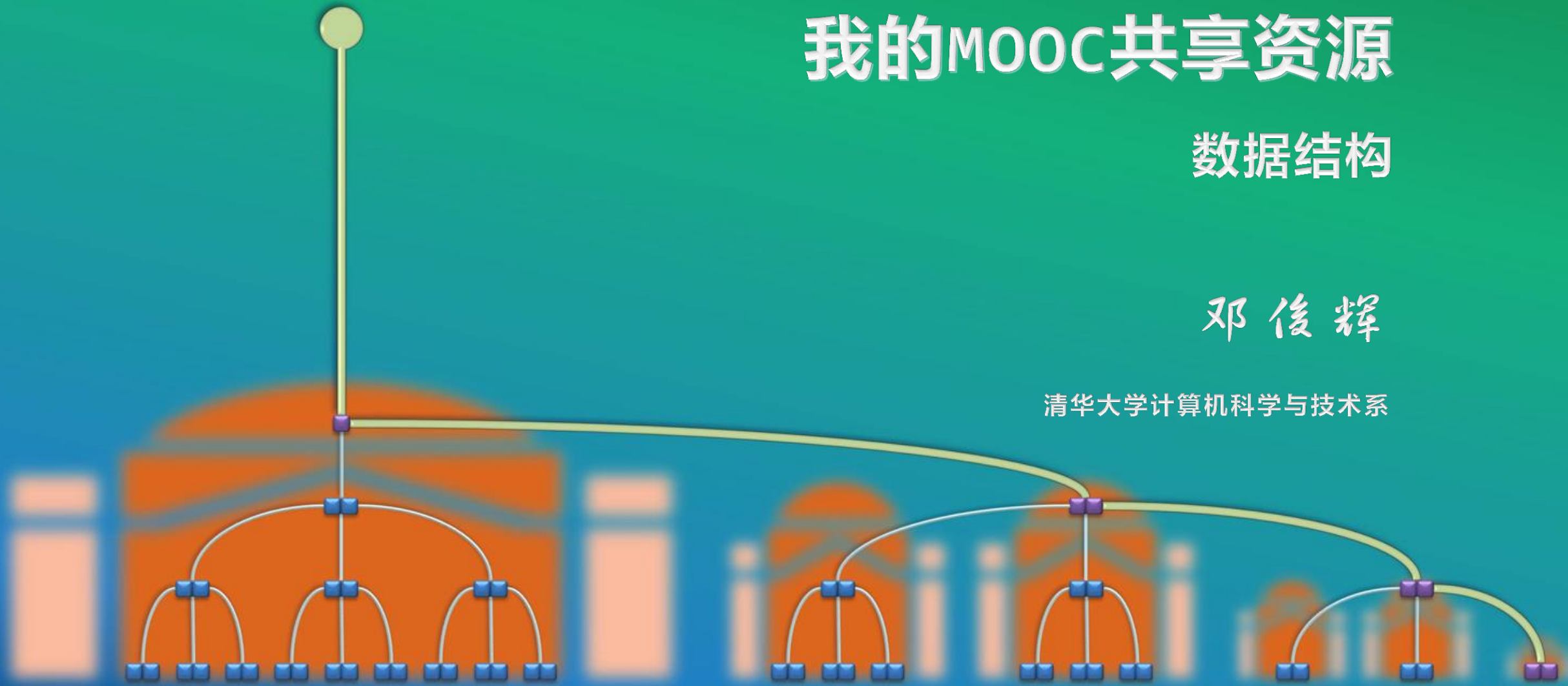
我的MOOC共享资源

数据结构

邓俊辉

清华大学计算机科学与技术系



兄弟院校
SPOC

校内传统课堂

校内SPOC

XueTangX + EDX
MOOC

教材

讲义

试卷

习题集

视频 + 字幕
36hr / 501

段间测试
250

测验题库
120

演示库

编程训练
3 x 4

一体化讲义、教材

论坛/wiki/FAQ

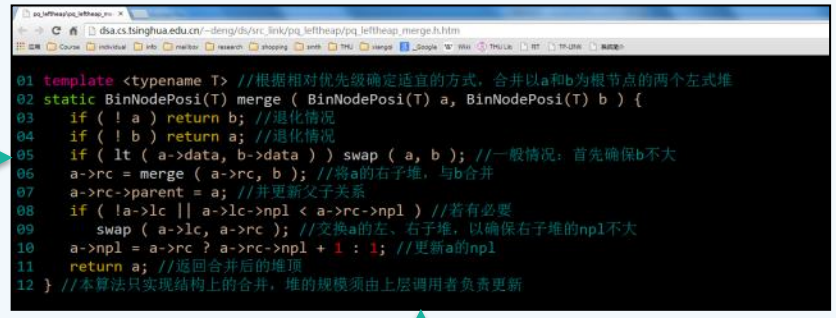
微信公众号

教材 + 讲义

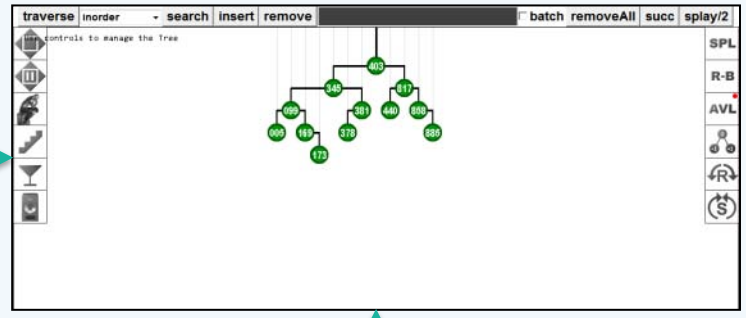
- ✓ http://dsa.cs.tsinghua.edu.cn/~deng/dsacpp
- ✓ http://pan.baidu.com/s/1i3IWISd
  密码：158w

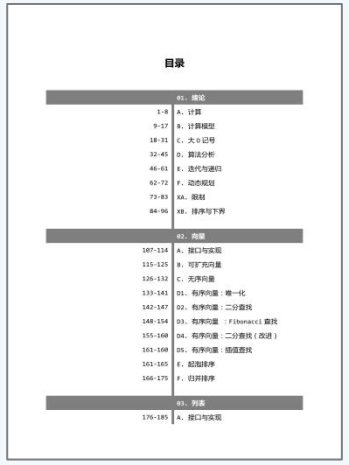

清华大学 计算机系列教材
邓俊辉 编著
数据结构(C++语言版)
(第3版)

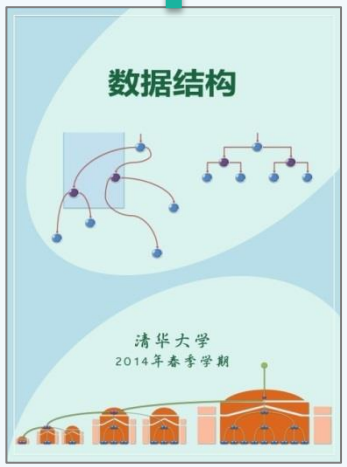

清华大学 计算机系列教材
邓俊辉 编著
数据结构习题解析
(第3版)


dsa.cs.tsinghua.edu.cn/~deng/ds/src_link/pq_leftheap/pq_leftheap_merge.h.htm
template <typename T> //根据相对优先级确定适宜的方式，合并以a和b为根节点的两个左式堆
static BinNodePosi(T) merge (BinNodePosi(T) a, BinNodePosi(T) b) {
if (! a) return b; //退化情况
if (! b) return a; //退化情况
if (lt (a->data, b->data)) swap (a, b); //一般情况：首先确保b不大
a->rc = merge (a->rc, b); //将a的右子堆，与b合并
a->rc->parent = a; //并更新父子关系
if (!a->lc || a->lc->npl < a->rc->npl) //若有必要
swap (a->lc, a->rc); //交换a的左、右子堆，以确保右子堆的npl不大
a->npl = a->rc ? a->rc->npl + 1 : 1; //更新a的npl
return a; //返回合并后的堆顶
} //本算法只实现结构上的合并，堆的规模须由上层调用者负责更新


LeftHeap
template <typename T> //基于二叉树，以左式堆形式实现的优先级队列
class PQ_LeftHeap : public PQ<T>, public BinTree<T> {
public:
void insert(T); //（按比较器确定的优先级次序）插入元素
T getMax() { return _root->data; } //取出优先级最高的元素
T delMax(); //删除优先级最高的元素
}; //主要接口，均基于统一的合并操作实现...
template <typename T>
static BinNodePosi(T) merge(BinNodePosi(T), BinNodePosi(T));
Data Structures (Spring 2014), Tsinghua University


traverse
inorder
search
insert
remove
batch
removeAll
succ
splay/2
controls to manage the Tree
SPL
R-B
AVL


目录
01. 绪论
02. 向量
03. 列表


数据结构
清华大学
2014年春季学期

■ 实现

依照以上思路，可针对列表实现插入排序算法如代码3.19所示。

```
1 template <typename T> //列表的插入排序算法：对起始于位置p的n个元素排序
2 void List<T>::insertionSort ( ListNodePosi(T) p, int n ) { //valid(p) && rank(p) + n <= size
3    for ( int r = 0; r < n; r++ ) { //逐一为各节点
4       insertA ( search ( p->data, r, p ), p->data ); //查找适当的位置并插入
5       p = p->succ; remove ( p->pred ); //转向下一节点
6    }
7 }
```

代码3.19 列表的插入排序

按3.4.2节的约定，有多个元素命中时search()接口将返回其中最靠后者，排序之后重复元素将保持其原有次序，故以上插入排序算法属于稳定算法。

■ 复杂度

插入排序算法共由n步迭代组成，故其运行时间应取决于，各步迭代中所执行的查找、删除及插入操作的效率。根据此前3.3.5节和3.3.7节的结论，插入操作insertA()和删除操作remove()均只需$\mathcal{O}(1)$时间；而由3.4.2节的结论，查找操作search()所需时间可在$\mathcal{O}(1)$至$\mathcal{O}(n)$之间浮动（从如表3.3所示的实例，也可看出这一点）。

不难验证，当输入序列已经有序时，该算法中的每次search()操作均仅需$\mathcal{O}(1)$时间，总体运行时间为$\mathcal{O}(n)$。但反过来，若输出序列完全逆序，则各次search()操作所需时间将线性递增，累计共需$\mathcal{O}(n^2)$时间。在等概率条件下，平均仍需要$\mathcal{O}(n^2)$时间（习题[3-10]）。

3.5.3 选择排序

选择排序（selectionsort）也适用于向量与列表之类的序列结构。

■ 构思

与插入排序类似，该算法也将序列划分为无序前缀和有序后缀两部分；此外，还要求前缀不大于后缀。如此，每次只需从前缀中选出最大者，并作为最小元素转移至后缀中，即可使有序部分的范围不断扩张。

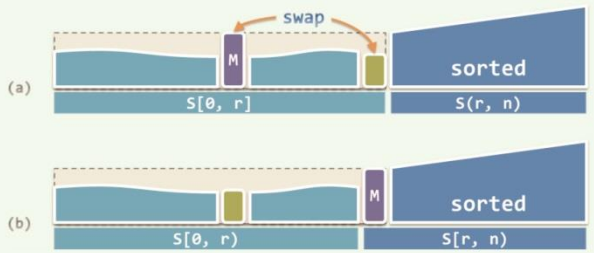


图3.6 序列的选择排序

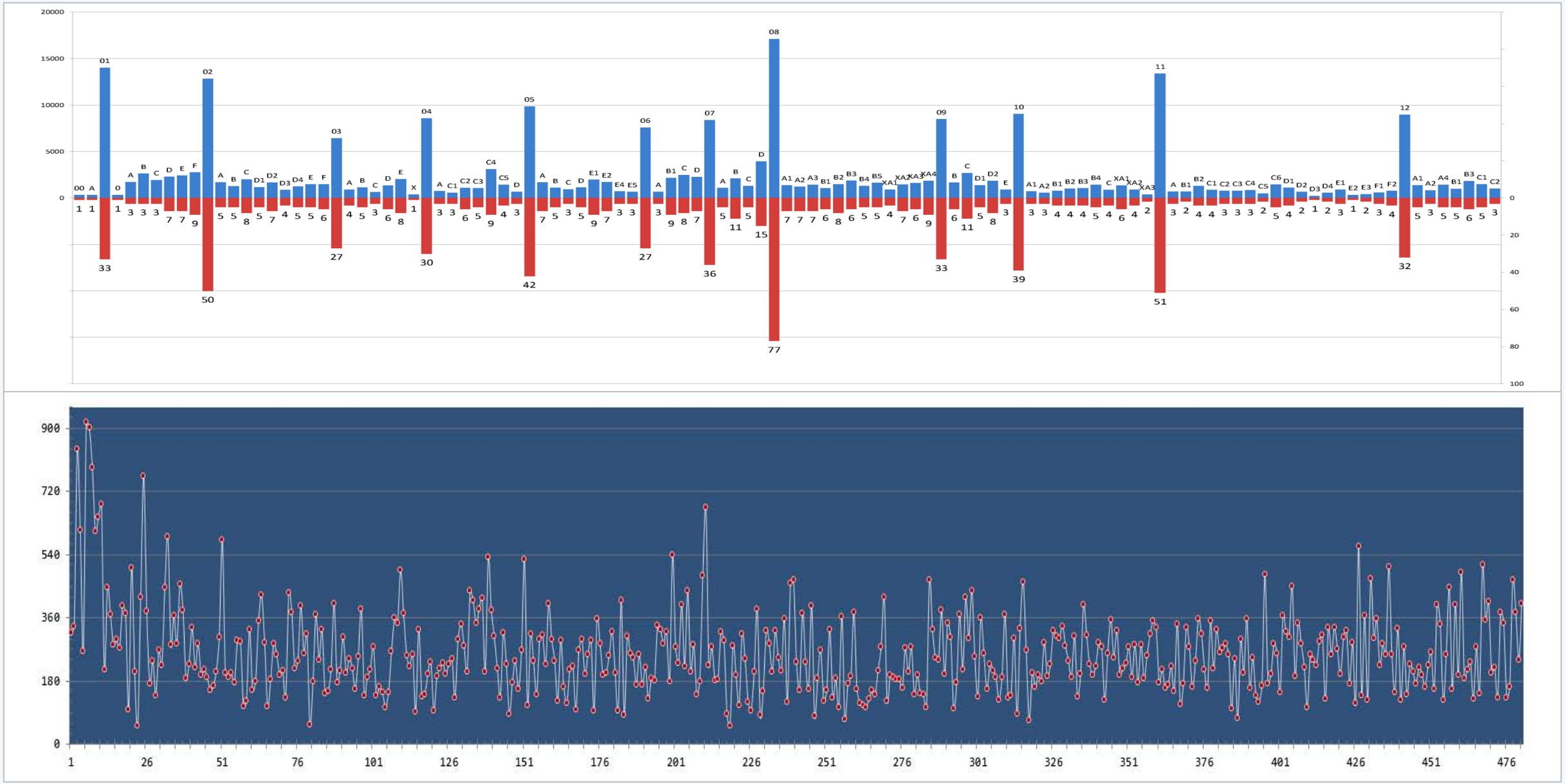
同样地，上述描述也给出了选择排序算法过程所具有的不变性：

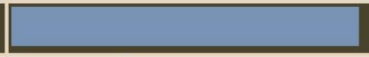
W

X

讲授视频

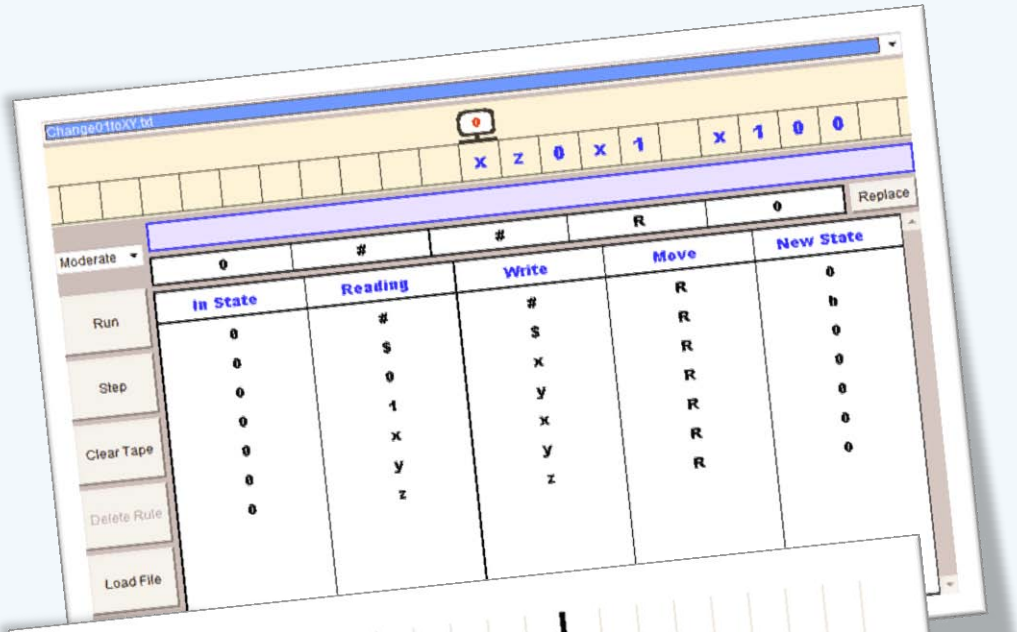
- ✓ 2200分钟 = 36小时 / 501段
- ✓ 104节 / 12章 / 16讲
- ✓ http://pan.baidu.com/s/1eQ8PG8U
  密码：eb67

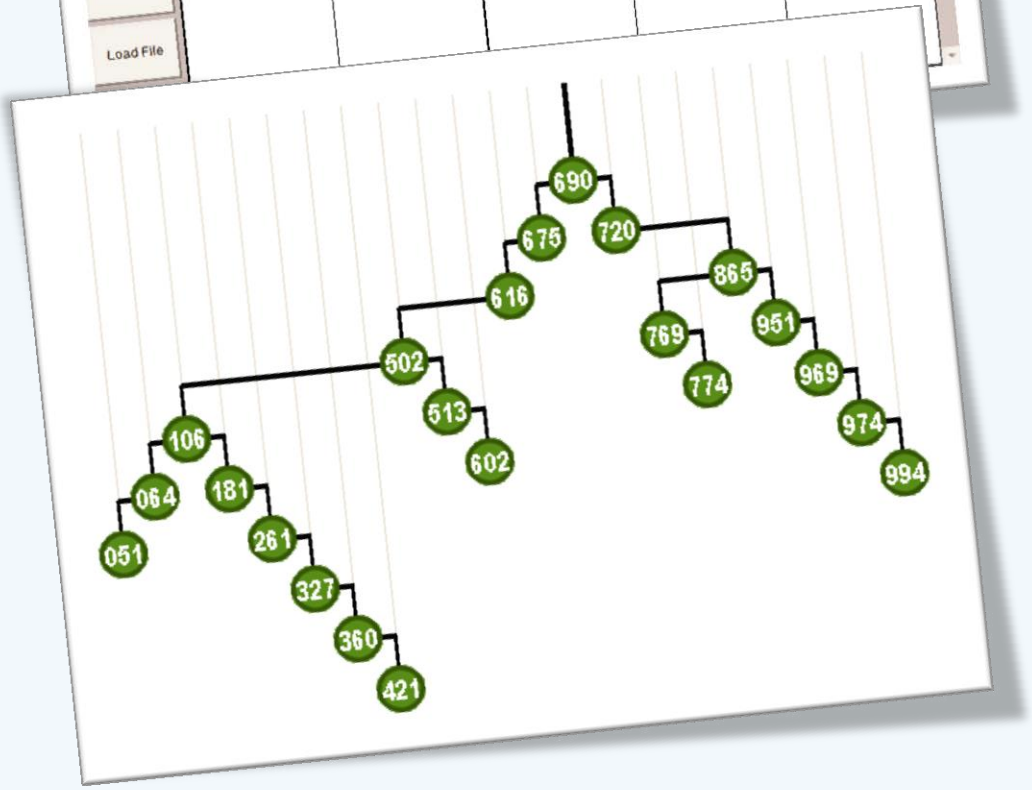

20000
15000
10000
5000
0
100
80
60
40
20
0
900
720
540
360
180
0
1
26
51
76
101
126
151
176
201
226
251
276
301
326
351
376
401
426
451
476

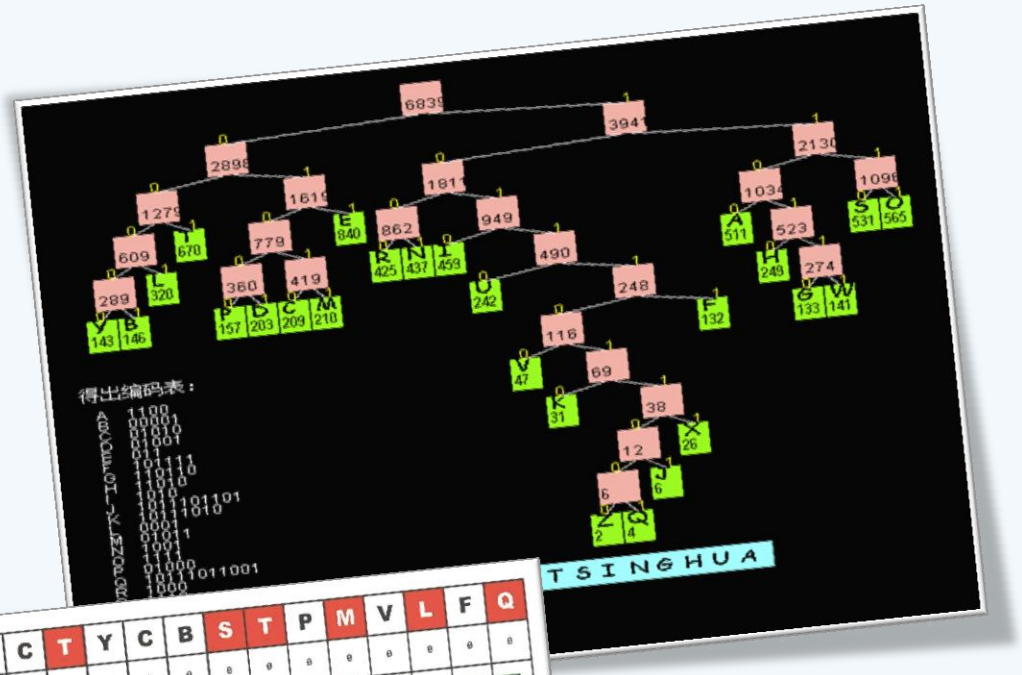
讲次		章、节主题	视频段数	视频时长	
第01讲.		绪论（上）	23	112'39	
00		OJ简介	1	6'28	
01-0		课程简介	1	5'35	
01-A		计算	6	29'2	
01-B		计算模型	8	42'10	
01-C		大O记号	7	35'52	
第02讲.		绪论（下）	26	142'8	
01-D		算法分析	7	43'36	
01-E		迭代与递归	9	46'47	
01-XC		动态规划	10	51'45	
第03讲.		向量（上）	30	134'45	
02-A		接口与实现	5	29'29	
02-B		可扩充向量	5	21'27	
02-C		无序向量	8	35'13	
02-D1		有序向量：唯一化	5	19'58	
02-D2		有序向量：二分查找	7	28'38	
第04讲.		向量（下）	25	110'55	
02-D3		有序向量：Fib查找	4	15'21	
02-D4		有序向量：二分查找（改进）	5	20'54	
02-D5		有序向量：插值查找	5	19'50	

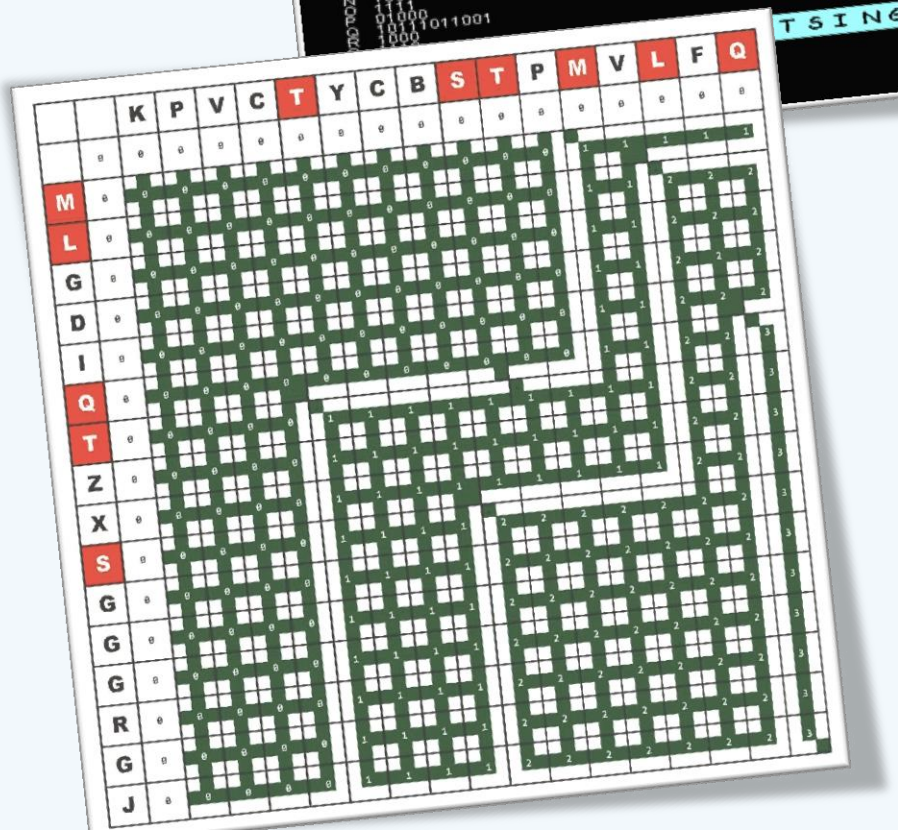
算法演示库

- ✓ 200余个演示
- ✓ 多为Java Applet
- ✓ 部分Excel格式
- ✓ http://dsa.cs.tsinghua.edu.cn/~deng/ds/demo


Change01to0Y.txt
x z 0 x 1 x 1 0 0
Replace
Moderate
Run
Step
Clear Tape
Delete Rule
Load File
In State
Reading
Write
Move
New State


690
675
720
616
865
502
769
951
513
774
969
602
974
106
994
064
181
051
261
327
360
421


得出编码表：
TSINGHUA


K P V C T Y C B S T P M V L F Q
M
L
G
D
I
Q
T
Z
X
S
G
G
G
R
G
J

在线编程实验平台（1/2）

- ✓ http://dsa.cs.tsinghua.edu.cn/oj
- ✓ **配有帮助文档，提供视频教程**

  http://pan.baidu.com/s/1i35tlFR

  密码：gp22

Tsinghua Online Judge

Department of Computer Science and Technology, Tsinghua University

New here? CREATE AN ACCOUNT

Welcome!

A quick guide to online judge is available here (video guide).

10817 registered users, **11** active courses, and **25** closed courses.

1534 test runs in the last 24 hours.
12929 test runs during last week.
422898 test runs since 09/01/2012.

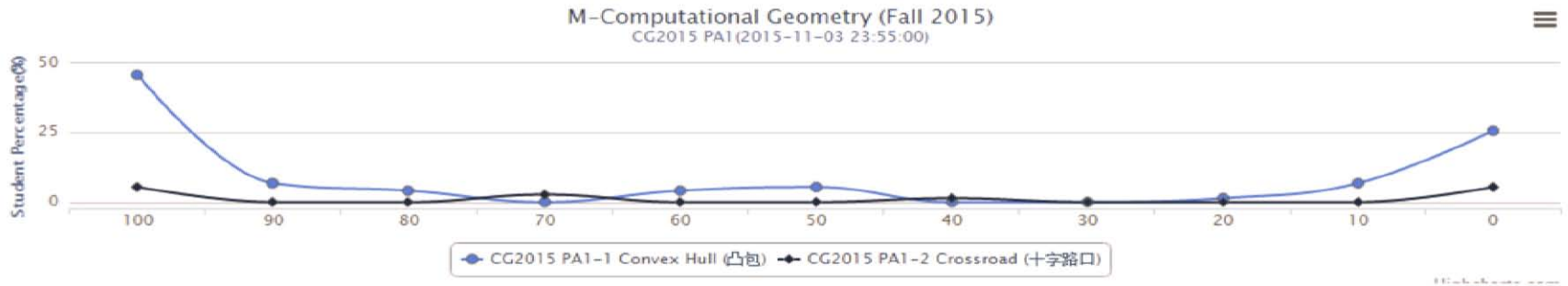
Username
deng@tsinghua.edu.cn

Password
••••••

Remember me

Sign in

Forgot my password!



在线编程实验平台（2/2）

- ✓ 100余道分级习题，自定义部署
- ✓ 自动评测、统计和发布成绩
- ✓ 实时监测进度与状态

第56题 股票（Stock） 30240184@2012S-2

“天下大势，分久必合，合久必分”，股市亦是如此：新股不断发行，旧股相继退出。从发行到退出之间的时段，称作股票的活跃期。

为简化起见，这里不妨假定活跃期均以交易日为基本单位，并以股市开市之（第零）日作为统一起点。比如，若股票 A “在第 123 天发行并于第 130 天退出”，则其活跃期为第 123 至 129 天共计 7 天，记作[123, 130)。另外，不妨假定各股票在其活动期内价格不变——否则，可等效于原股票退出，同时发行一支定价更新的股票。

Q 王国的股市虽不能脱俗，却亦有其与众不同之处。在该国的股市中，禁止不同股票的活跃期相互包含。比如，无论是[100, 150)和[120, 130)、[120, 130)和[120, 150)、[100, 130)和[120, 130)，甚至是[120, 130)和[120, 130)，都不会同时作为股票的活跃期。也就是说，每天发行和退市的股票累计不超过一支。

Maximum（卖个死萌）先生是 Q 王国公认的股神，他所采用的策略既稳妥也简单。所谓稳妥是指，任何时候他的投资总额都保持固定，如此可以保证风险有限。所谓简单是指，他每天都将所有的投资用于购入当日价格最低的一支股票（为此可能需要先卖出原持有股票，且交易所需时间忽略不计），如此他的持股量每天都能保持最大——在 Q 王国，这可是评估股市收益的首要指标。

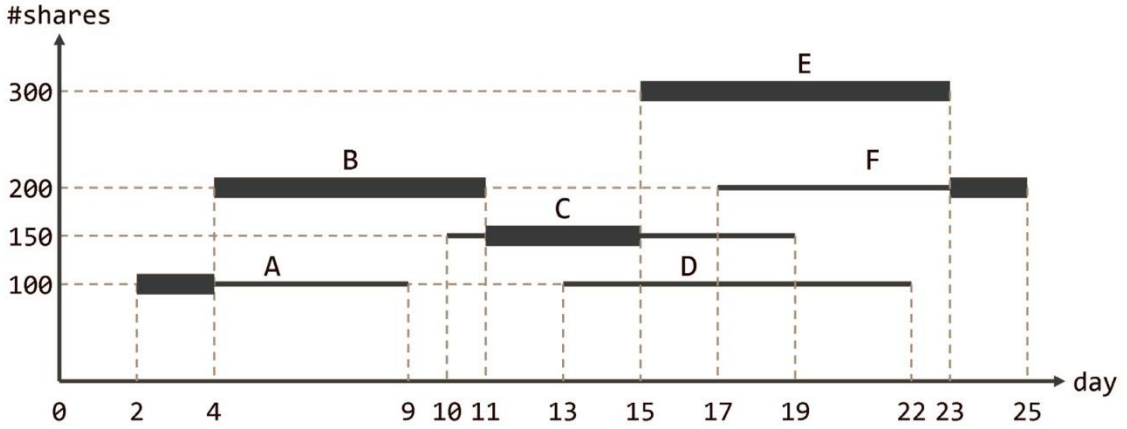
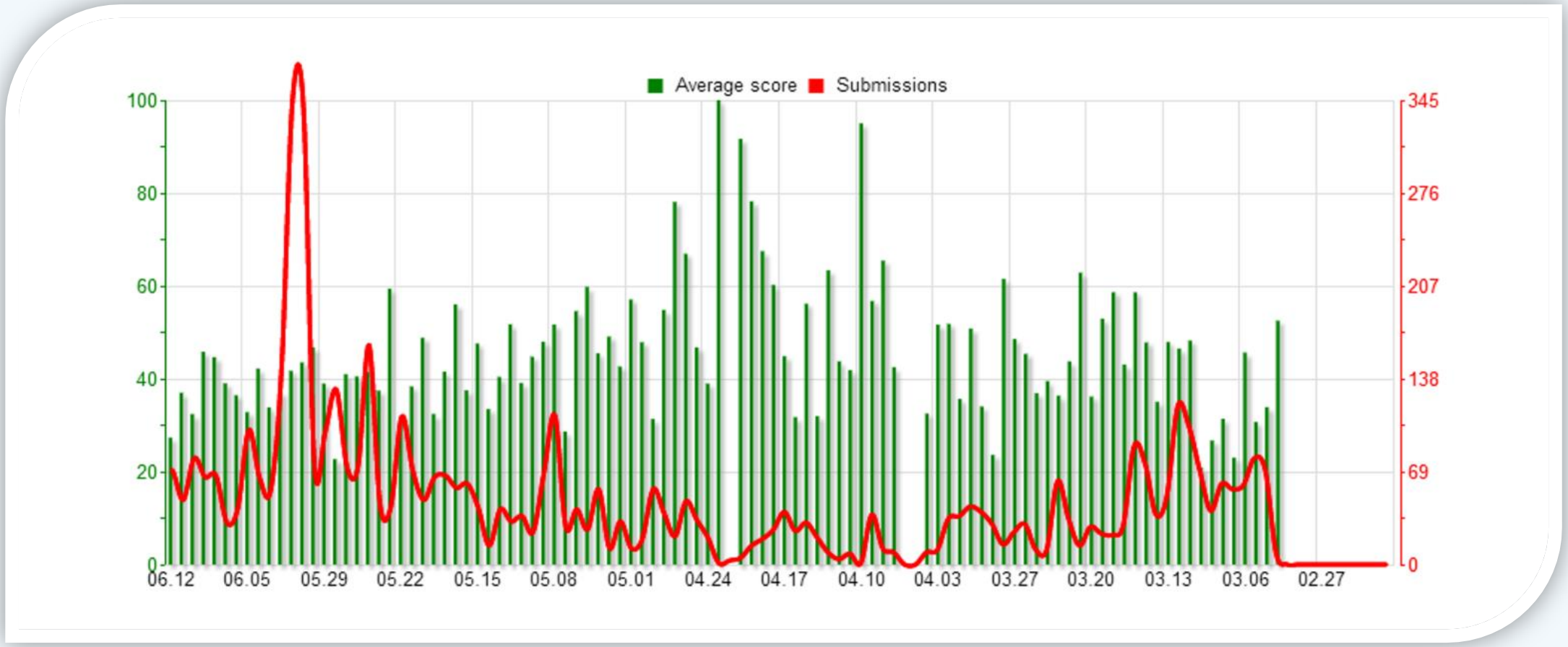
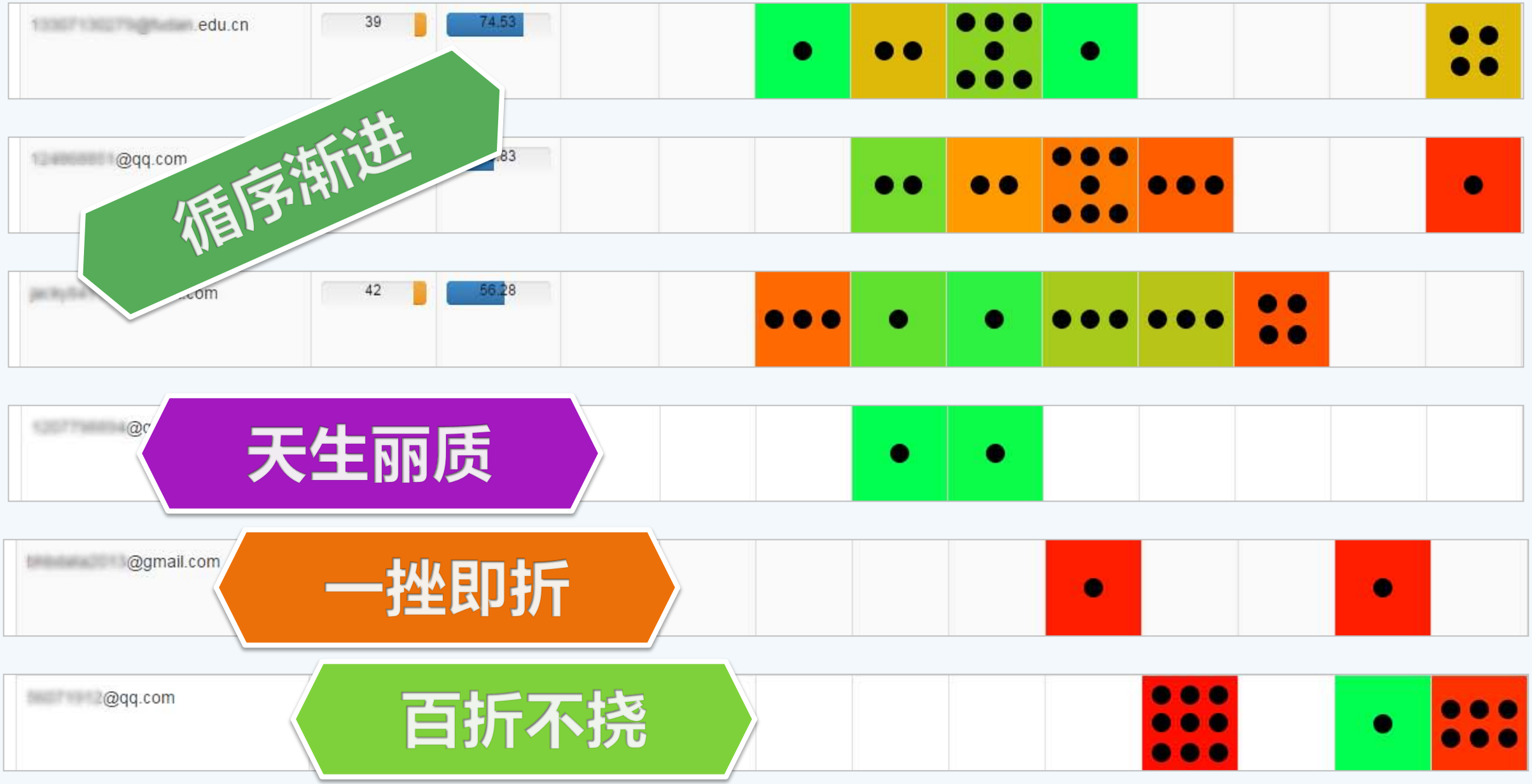


图1. 实例：股市共持续25天，先后发行6支股票，Maximum先生日均持有200股


Average score
Submissions
100
80
60
40
20
0
345
276
207
138
69
0
06.12
06.05
05.29
05.22
05.15
05.08
05.01
04.24
04.17
04.10
04.03
03.27
03.20
03.13
03.06
02.27


edu.cn
39
74.53
@qq.com
83
com
42
56.28
@gmail.com
@qq.com
循序渐进
天生丽质
一挫即折
百折不挠

编程作业查重系统

- ✓ 操作流程简捷，配有使用文档
- ✓ 自动生成报表，可视定位对比
- ✓ 捕捉各种作弊形式
- ✓ 智能忽略经典算法

COSIDE 首页

COSIDE查重系统

1.在右上角注册/登录

2.上传待查重的代码包

3.在个人上传的历史记录中查看结果

下载说明手册

地址：北京市海淀区清华大学

登录

Honor Code

补充下面各栏的内容，本声明需要在每道题目分别签署。
否则，你的作业中的该题目的提交将被视作无效而不能参与评测，对应的题目成绩按0分计。

在完成本作业的过程中，我从以下各位那里得到过以下方面的启发和帮助：

感谢[redacted]同学启发我使用lazy_tag的更新方法

在我提交的程序中，还在对应的位置以注释形式记录了他们的具体帮助。

此外，我也参考了以下资料：

讲义上二叉树的部分实现

在我提交的程序中，还在对应的位置以注释形式记录了具体的参考内容。

我独立完成了本程序除以上方面之外的所有工作，包括分析、设计、编码、调试与测试。
我清楚地知道，从以上方面获得的信息在一定程度上降低了作业难度，可能影响起评分。

我从未抄袭或盗用过他人的程序，不管是原封不动地复制，还是经过了某些等价转换。
我未曾也不会向同一课程（包括此后各届）的同学复制或公开我这份程序的代码，我有义务妥善保管好它们。
我编写这个程序无意于破坏或妨碍任何计算机系统的正常运转。
我清楚地知道，以上情况均为本课程纪律所禁止，若违反，对应的作业成绩将按-100计。

[redacted] 2014.12.22 2014-12-22

Close

File 1	File 2	Fingerprint Matched
201301 (99%)	201301 (99%)	184
201301 (58%)	201301 (59%)	106
201301 (30%)	201301 (27%)	98
201301 (15%)	201301 (23%)	87
201301 (19%)	201301 (22%)	41
201301 (19%)	201301 (10%)	15
201301 (9%)	201301 (18%)	15
201301 (14%)	201301 (17%)	19
201301 (12%)	201301 (16%)	54
201301 (16%)	201301 (10%)	13
201301 (16%)	201301 (6%)	13
201301 (16%)	201301 (8%)	13
201101 (%)	201301 (14%)	26
201301 (14%)	201301 (13%)	26


9 21
6 101
26~29
130~133
33~37
138~142
40~48
31~43
template <typename T> void Vector<T>::shrink() { //装填因子过小时压缩向量所占空间
if (_capacity < DEFAULT_CAPACITY << 1) return; //不致收缩到DEFAULT_CAPACITY以下
if (_size << 2 > _capacity) return; //以25%为界
T* oldElem = _elem; _elem = new T[_capacity >>= 1]; //容量减半
for (int i = 0; i < _size; i++) _elem[i] = oldElem[i]; //复制原向量内容
delete[] oldElem; //释放原空间
}
template <typename T> //在有序向量的区间[lo, hi)内，确定不大于e的最后一个节点的秩
Rank Vector<T>::search(T const& e, Rank lo, Rank hi) const { //assert: 0 <= lo < hi <= _s
return binSearch(_elem, e, lo, hi);
}
//二分查找算法（版本C）：在有序向量的区间[lo, hi)内查找元素e，0 <= lo <= hi <= _size
template <typename T>
static Rank binSearch(T* A, T const& e, Rank lo, Rank hi) {
while (lo < hi) { //每步迭代仅需做一次比较判断，有两个分支
Rank mi = (lo + hi) >> 1; //以中点为轴点
(A[mi] > e) ? hi = mi : lo = mi + 1; //经比较后确定深入[lo, mi)或(mi, hi)
} //成功查找不能提前终止
return --lo; //循环结束时，lo为大于e的元素的最小秩，故lo - 1即不大于e的元素的最大
} //有多个命中元素时，总能保证返回秩最大者；查找失败时，能够返回失败的位置
template <class T> //排序!
void Vector<T>::sort(Rank lo, Rank hi)

考查评分

- ✓ 习题解析，300+
- ✓ 段间练习，250+
- ✓ 章节测验，120+
- ✓ 编程训练，3 x4

02-B-2 QUIZ (1 point possible)

是否可以将视频里向量扩容代码中的：

```
for (int i = 0; i < _size; i++) _elem[i] = oldElem[i];
```

替代为：

```
memcpy(_elem, oldElem, _size * sizeof(T));
```

P.S.本题涉及C++的相关知识

- 是，二者是等价的，不会有任何问题。
- 否，因为二者复制的元素区间范围不同。
- 否，因为二者的效率不同。
- 否，因为后者能否达到目的与元素类型T有关。

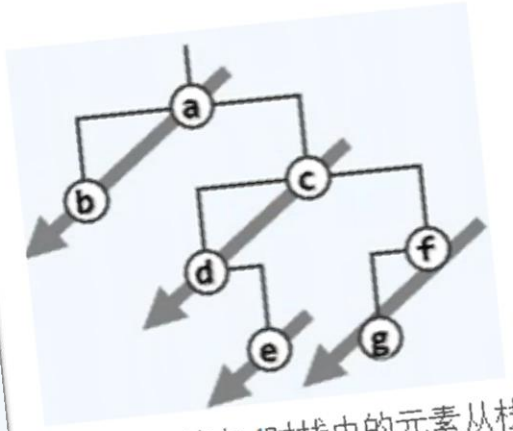
QUIZ (1 point possible)

以现在普通计算机的速度，直接用定义以递归的方式计算fib(100)需要多少时间（不考虑溢出）：

- 一小时之内
- 大约一天
- 十年
- 这辈子看不到啦

高度为3的AVL树至少包含几个节点？

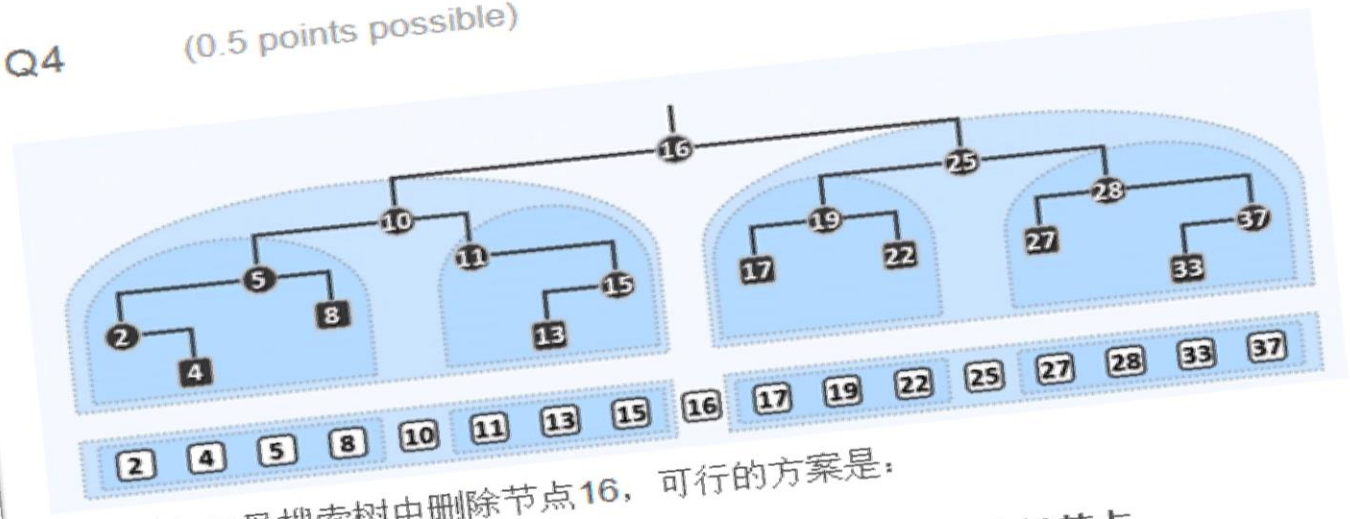
对以下二叉树进行先序遍历：



刚访问完节点d时栈中的元素从栈顶到栈底

- c,e
- g,f
- e,f,g
- e,f

Q4 (0.5 points possible)



欲在以上二叉搜索树中删除节点16，可行的方案是：

- 将16摘除后令25为10的右子，从而10是新的根节点
- 以16为轴进行一次zig操作使之不再是根节点，再直接摘除16
- 将节点16和节点15的关键码互换，摘除新的节点16并令13为11的右子
- 将节点16和节点33的关键码互换，再摘除新的节点16