

姓名								
2	0	1						
班级								

题号	[1]	[2]	[3]	[4]	[5]	[6]	$\Sigma$
得分							
题分	8	6	56	8	6	16	100

## 1. B-树

3 + 5

考查包含2018个关键码的16阶B-树，约定根节点常驻内存，且在各节点内部采用顺序查找。

a) 在单次成功查找的过程中，至多可能需要读多少次磁盘？请列出估算的依据。

b) 在单次成功查找的过程中，至多可能有多少个关键码需要与目标关键码做比较？请列出估算的依据。

## 2. 理想随机

2 x3

本课程所介绍的一些算法与数据结构，乃是针对实际应用中普遍存在的非随机数据集而设计的；反过来，只要数据集是理想随机的，则大可不必采用。试举三个这样的案例，列出讲义页码，并作简要说明（各不超过两行）。

- ☐ 在某节点被删除后AVL树的高度即便下降了, 这次操作期间也未必做过旋转调整。
- ☐ 在图DFS()算法中的default分支, 将  $\text{dTime}(v) < \text{dTime}(u)$  改为  $\text{dTime}(v) < \text{fTime}(u)$  同样可行。
- ☐ 有向图经DFS后若共有 $k$ 条边被标记为BACKWARD, 则它应恰有 $k$ 个环路。
- ☐ 左式堆中每一对兄弟节点的高度尽管未必“左大右小”, 但左兄弟至少不低于右兄弟的一半。
- ☐ 对于同一无向图, 起始于顶点 $s$ 的DFS尽管可能得到结构不同的DFS树, 但 $s$ 在树中的度数必然固定。
- ☐ 采用Crane算法将左式堆 $A$ 与 $B$ 合并为左式堆 $H$ ,  $H$ 右侧链上的节点未必都来自 $A$ 或 $B$ 的右侧链。
- ☐ 采用单向平方策略的散列表, 只要长度 $M$ 不是素数, 则每一组同义词在表中都不会超过 $\lfloor M/2 \rfloor$ 个。
- 
- ☐ 经快速划分 ( LGU版 ) 之后, 后缀 $G$ 中的雷同元素可能调换相对次序, 但其余部分的雷同元素绝不会。
- ☐ PFS过程中, 尽管每一步迭代都可能多次调用prioUpdater(), 但累计不过 $O(e)$ 次。
- ☐ 只要底层的排序算法是正确且稳定的, 则radixSort()也必然是正确且稳定的。
- ☐ 相对于KMP算法而言, BM算法更适合于大字符集的应用场合。
- ☐ 若调用BST::remove( $e$ )将节点 $x$ 从常规BST中删除, 则所需的时间为被删除之前 $x$ 的深度。
- ☐ 在存有 $n$ 个词条的跳转表中, 各塔高度的期望值为 $\Theta(\log n)$ 。
- ☐ 将 $n$ 个词条逐个插入一个容量为 $M$ 、采用线性试探策略、初始为空的散列表,  $n < M$ , 则无论它们的插入次序如何, 最终的平均成功查找长度都必然一样。
- 
- ☐ 红黑树的插入或删除操作, 都有可能导致 $\Omega(\log n)$ 个节点的颜色反转。
- ☐ 只有在访问序列具有较强的局部性时, 伸展树才能保证分摊 $O(\log n)$ 的性能。
- ☐ 将 $\{0, 1, 2, \dots, 2018\}$ 插入一棵空的伸展树后若树高为2018, 则上述词条必是按单调次序插入的。
- ☐ 相对于同样规模的完全二叉堆, 多叉堆delMax()操作的时间成本更低。
- ☐ 在插入操作后若红黑树黑高度增加, 则在双红修复过程中仅做过重染色, 而无任何结构调整。
- ☐ 最底层的叶节点一旦被访问 ( 并做过splay调整 ) 之后, 伸展树的高度必然随即下降。
- ☐ 若输入序列包含 $\Omega(n^2)$ 个逆序对, 则快速排序算法 ( LUG版 ) 至少需要执行 $\Omega(n \log n)$ 元素交换操作。
- 
- ☐ 胜者树的根节点即是冠军, 而败者树的根节点即是亚军。
- ☐ 采用12-C节中介绍的任何一种增量序列, shellSort()最后的1-sorting都只需要 $O(n)$ 时间。
- ☐ B-树的任一非叶节点内, 每个关键码都存在直接后继, 且必然来自某个叶节点。
- ☐ 无论是单独借助BC[]表或GS[]表, BM算法在最好情况下都只需要 $O(|T|/|P|) = O(n/m)$ 时间。
- ☐ ShellSort()每按照某个增量做过逐列排序, 序列中逆序对的总数都会减少 ( 或持平 ), 但绝不致增加。
- ☐ 对规模为 $n$ 的AVL树做一次插入操作, 最坏情况下可能引发 $\Omega(\log n)$ 次局部重构。
- ☐ 若用完全二叉堆来实现PFS, 则各顶点在出堆之前, 深度只可能逐步减少 ( 或保持 ) 而不致增加。

某散列表 $\mathcal{H}[0, M = 2^s]$ 采用封闭散列策略 ( 初始令 $c = d = 0$ ) : 对于任何 $key$ , 首先试探 $\mathcal{H}[key \% M]$ ; 以下, 只要冲突, 就令 $c \leftarrow c + 1$ 再 $d \leftarrow d + c$ , 并继而试探 $\mathcal{H}[(key + d) \% M]$ 。以 $M = 2^4 = 16$ 为例, 关键码 $key = 27$ 的前五个试探位置依次是: 11、12、14、1、5。但如同对于平方试探策略, 我们首先需要确认, 这种试探序列是否总能覆盖所有桶单元。若是, 请给出证明; 否则, 试举一 (  $s$ 和 $key$ 组合的 ) 反例。

## 5. 多产

2 x3

计算机科学家往往在多个方面同时有所建树。试以讲义上介绍的算法或数据结构为例, 列举出其中的三位, 以及他们各自的两项贡献。请注明在讲义上对应的页码, 并作简要说明 ( 每人每项不超过一行 )。

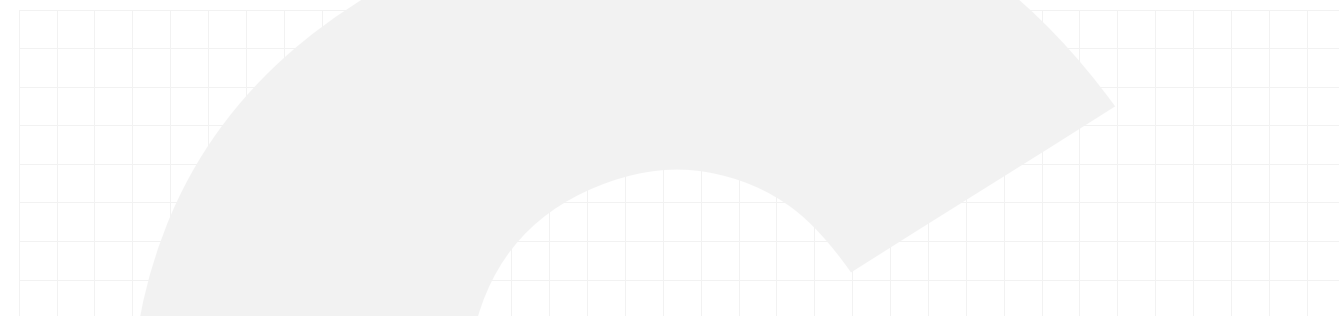
所谓的斐波那契串 ( Fibonacci Strings ), 系由字符集  $\Sigma = \{ '0', 'X' \}$  生成:  $\phi_0 = "0"$ ,  $\phi_1 = "X"$ ; 对于  $k \geq 2$ , 有  $\phi_k = \phi_{k-1}\phi_{k-2}$ , 比如:  $\phi_2 = "X0"$ ,  $\phi_3 = "X0X"$ ,  $\phi_4 = "X0XX0"$ , .....。

1) 以下考查KMP算法的改进版。试列出  $\phi_8$ , 并计算其对应的查询表。

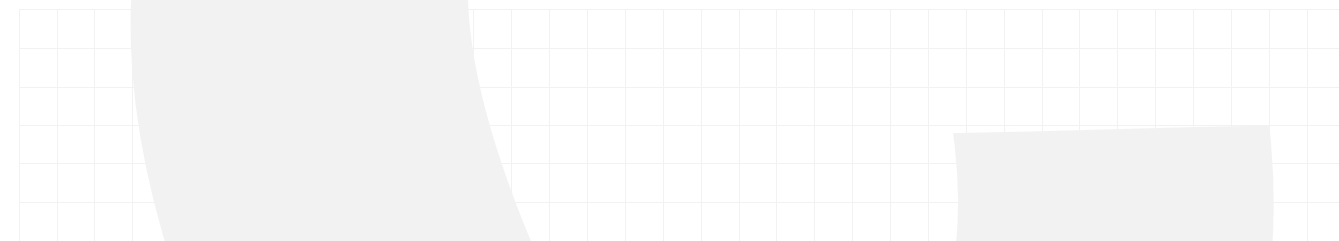
$j$	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
$\phi_8[j]$																					
$improved\_next[j]$																					

2) 若  $|\phi| \geq 2$ , 则将  $\phi$  末尾的两个字符翻转, 得到的串可记作  $\phi'$ 。比如,  $\phi'_5 = "X0XX0XX0"$ 。

试证明:  $\forall k \geq 2, \phi_{k-2}\phi_{k-1} = \phi'_k$



3) 试证明: 若以  $\phi_k$  作为模式串, 文本串的某个  $T[i]$  可能参与  $\Omega(k)$  次比较。



4) 试证明, 对于任何模式串  $P$ , 文本串的每一字符至多会与  $P$  中的  $\mathcal{O}(\log m)$  个字符做比对,  $m = |P|$ 。

