北京大学信息科学技术学院考试试卷

**考试科目：** 数据结构与算法A  **姓名：**  **学号：**

**考试时间：** 2016 年 1 月 6 日 任课**教师:**

**北京大学考场纪律**

1、考生进入考场后，按照监考老师安排隔位就座，将学生证放在桌面上。无学生证者不能参加考试；迟到超过15分钟不得入场。在考试开始30分钟后方可交卷出场。

2、除必要的文具和主考教师允许的工具书、参考书、计算器以外，其它所有物品（包括空白纸张、手机、或有存储、编程、查询功能的电子用品等）不得带入座位，已经带入考场的必须放在监考人员指定的位置。

3、考试使用的试题、答卷、草稿纸由监考人员统一发放，考试结束时收回，一律不准带出考场。若有试题印制问题请向监考教师提出，不得向其他考生询问。提前答完试卷，应举手示意请监考人员收卷后方可离开；交卷后不得在考场内逗留或在附近高声交谈。未交卷擅自离开考场，不得重新进入考场答卷。考试结束时间到，考生立即停止答卷，在座位上等待监考人员收卷清点后，方可离场。

4、考生要严格遵守考场规则，在规定时间内独立完成答卷。不准交头接耳，不准偷看、夹带、抄袭或者有意让他人抄袭答题内容，不准接传答案或者试卷等。凡有违纪作弊者，一经发现，当场取消其考试资格，并根据《北京大学本科考试工作与学术规范条例》及相关规定严肃处理。

5、考生须确认自己填写的个人信息真实、准确，并承担信息填写错误带来的一切责任与后果。

学校倡议所有考生以北京大学学生的荣誉与诚信答卷，共同维护北京大学的学术声誉。

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 题号 | 一  8分 | 二  15分 | 三  7分 | 四  10分 | 五  25分 | 六  35分 | **总分** |
| 分数 |  |  |  |  |  |  |  |
| 阅卷人 |  |  |  |  |  |  |  |

装订线内 不要答题

以下以下为答题纸，共 页

以下为试题和答题纸(**答案写在答题纸上**)，包括本页共 6 页。

|  |
| --- |
| 得分 |
|  |

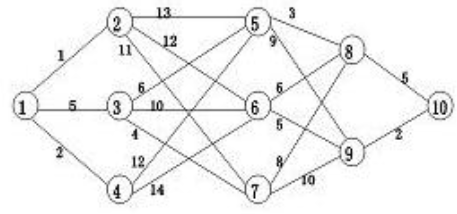
## 第一题 填空题（每空1分，共8分）

1. 设无向图G = (V,E)中V={1, 2, 3, 4, 5, 6, 7}，E={(1,2), (1,3), (2,3), (4,5), (3,6), (4,7), (5,7)}。则图G包含的不同生成森林的个数为\_\_\_\_\_9\_\_\_\_。

注释：其中若两个生成森林的边集不同，则认为它们是不同的生成森林。

1. 若对下面的无向图以1为起点运行单源最短路径的Dijkstra算法，则在Dijkstra算法的过程中，每一步依次被选出的顶点依次是\_1,2,4,3,7,5,6,8,9,10\_\_\_。若忽略边权，以6为起点进行一次广度优先周游，将得到一棵广度优先搜索的生成树。设仅包含一个结点的树高度为1，则该生成树的高度为\_3\_，树的最深层结点集合为\_1,5,7,10\_\_。

第一空回答2,4,3,7,5,6,8,9,10也算对



1. 用某种排序方法对序列（25, 84, 21, 47, 15, 27, 68, 35, 20）进行排序时，若序列的变化情况如下， 则所采用的排序方法为\_\_快速排序\_\_。

25, 84, 21, 47, 15, 27, 68, 35, 20

20, 15, 21, 25, 47, 27, 68, 35, 84

15, 20, 21, 25, 35, 27, 47, 68, 84

15, 20, 21, 25, 27, 35, 47, 68, 84

1. 假设Shell排序的增量序列是(2k, 2k-1, … , 2, 1)，其中k=[log2 n]。若待排序序列是**正序的**（已经排好）或**逆序的**，则Shell排序的时间复杂度**分别是 \_\_**O(nlogn)\_、O(nlogn)\_。

各0.5分

1. 设输入的关键码满足k1>k2>…>kn，缓冲区大小为m。在n=150, m=25的情况下，用最小值堆进行置换-选择排序方法可产生初始归并段的个数为\_\_\_6\_\_\_\_。
2. 假定有一棵B+树，它的内部结点可以存储多达20个子结点，叶结点可以存储多达15条记录（本题中的B+树把所有记录存放在叶结点上）。那么2层的B+树能够存储的**最多**记录个数和**最少**记录个数**分别为**\_300\_\_、\_\_16\_\_。

需要说明根节点是第一层，叶节点是第二层

否则是6000，160

各0.5分

|  |
| --- |
| 得分 |
|  |

## 第二题 简要和辨析题（每小题3分，共15分）

1. 设d是带权无向图G的相邻矩阵，G的任意两个顶点之间至多只有1条边，即：

若i=j，则d[i][j]=0；

若i与j之间有边权为w的边，则d[i][j]=w；

若i与j之间无边，则d[i][j]=+∞；

下面有一段用于求无向图中任意两点间最短路径的伪代码：

for i ← 1 to n do

for j ← 1 to n do

for k ← 1 to n do

d[i][j] ← min(d[i][j], d[i][k]+d[k][j]);

请判断这段伪代码的正确性。若正确，请给出证明；若错误，请分析错误原因并修正。

错误。（1分）

Floyd的算法本质是：d(k)[i][j]表示从点i到点j经过编号不超过k的点作为中转时的最短路，故d(k)[i][j] = min{ d(p)[i][k] + d(q)[k][j] | 1<=p,q<k }。k是该算法的一个阶段，最外层应以k循环，在循环开始时，d[i][j]存储的是途经点<k时的最短路径，在循环中使用点k为中转更新d数组，循环结束时就得到了途经点<=k时的最短路。以此直到k=n。故应该改为：

for k ← 1 to n do

for i ← 1 to n do

for j ← 1 to n do

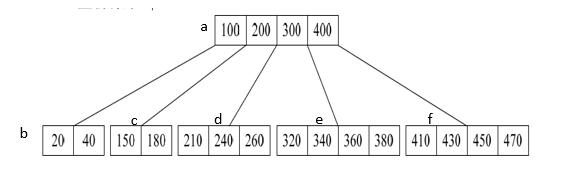
d[i][j] ← min(d[i][j], d[i][k]+d[k][j]);

注：

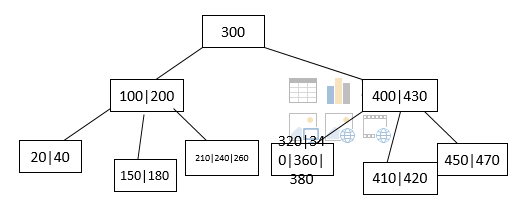
答出循环的顺序有问题（1分）

答出k作为中转时的最短路，或者其他详细解释（1分）

1. 请画出往下图的5 阶B 树中插入一个关键码420 后得到的B 树，以及再删除关键码40 后得到的B 树。并请分别分析插入和删除时的读写页块次数。（假设删除关键码时，如果需要从兄弟结点借关键码的话，先查看左边的、再查看右边的兄弟。在操作之前B 树所有的结点页块均在外存，在插入和删除过程中，新近被访问过的外存磁盘块都被缓冲在内存缓冲区中，不必重新访外，abcdef是对各个结点的标记）

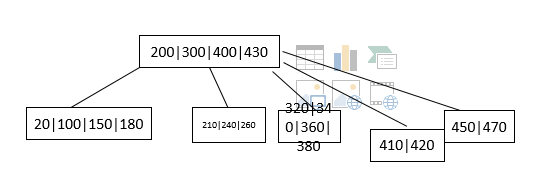


插入420（画图0.5分）



读2次，写5次。（0.5分）

再删除40后（画图0.5分）



考虑到之前已经缓存了第一层和第二层的所有结点，这次删除只需要读取2次（否则5次），另外需要写2次（各0.5分）

1. 已知一个线性表(38，25，74，63，52，48)，采用的散列函数为H(Key) = Key mod 7，将元素散列到表长为7的散列表中存储。
2. 若采用线性探查的冲突解决策略，则在该散列表上进行等概率成功查找的平均查找长度为多少？请写出计算过程。
3. 若采用拉链法解决冲突，则在该散列表上进行等概率成功查找的平均查找长度为多少？请写出计算过程。

(1). 线性探测法：

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 位置 | 1 | 2 | 3 | 4 | 5 | 6 | 7 |
| Key | 48 |  | 38 | 25 | 74 | 52 | 63 |
| 查找长度 | 3次 |  | 1次 | 1次 | 2次 | 4次 | 1次 |

共查找了12次，平均查找长度为：12 / 6 = 2次

(2). 拉链法：

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 位置 | 1 | 2 | 3 | 4 | 5 | 6 | 7 |
| Key |  |  | 38 | 25 |  | 48 | 63 |
| Key |  |  | 52 | 74 |  |  |  |

平均查找长度为：(1 + 2 + 1 + 2 + 1 + 1) / 6 = 4 / 3次

注：解释各1分，答案各0.5分

1. 广义表A=（（（a）），（b），c，（a），（（（d，e））））

（1）画出其一种存储结构图；

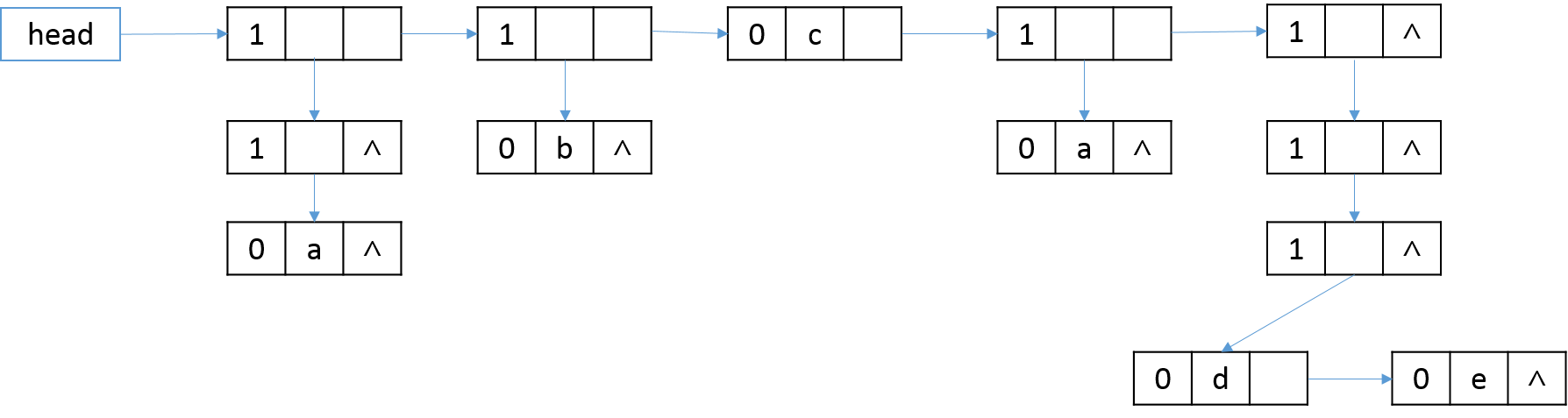
（2）写出表的长度与深度；

（3）用求头部，尾部的方式求出 e。

每个小题各0.5分

(1)不带头结点的情况

下面是 a 分别存储的情况。a 形成一个子表，统一存储，更有利于共享。这两种情况都不扣分。



带头结点的话，则在子表与子表中元素之间加入头结点。

(2)长度为5，深度为4

（3）head(tail(head(head(head(tail(tail(tail(tail(A)))))))))

1. 对于顺序输入的关键码 {1, 2, 3, 4, 5, 6, 7}
2. 严格遵循AVL树的操作规定，画出所得到的AVL树。
3. 严格遵循红黑树的操作规定，画出所得到的红黑树。

各1.5分

AVL: 4 ( 2 ( 1 3 ) 6 ( 5 7 ) )

RB: **2** ( **1** () 4 ( **3** () **6** (5 7) ) )

|  |
| --- |
| 得分 |
|  |

## 第三题 算法填空题（每空1分，共7分）

1. 下面的代码实现了一种计数排序：对每个待排序记录，扫描整个序列统计比它小的记录个数count，count即是该记录在序列中的正确位置。

template <class Record > void Sort(Record Array[], int n) {

int curIndex = 0; // 待排下标

while (curIndex < n) { // 需要一直进行，直到已经遍历了整个数组

int count = 0;

// 统计比Array[curIndex]小的值的个数

for ( ) // 空缺1

if (Array[j] < Array[curIndex])

\_\_\_; // 空缺2

swap(Array, curIndex, count);

/\* 如果curIndex不用移动，考虑下一个位置；

否则，继续考虑该位置（因为其他没排序的值交换过来了）\*/

if (curIndex == count)

\_\_ \_\_; // 空缺3

}

}

1. int j = 0; j<n; j++
2. count++;
3. curIndex++;
4. 考虑用双向链表来实现一个有序表，使得能在这个表中进行正向和反向检索。若指针p总是指向最后成功检索到的结点，检索可以从p所指结点出发沿任一方向进行。根据这种情况编写一个函数search(head, p, key)，检索具有关键码key的结点，并相应地修改p。

bool search( LinkNode \* head, LinkNode \* &p, int key ) {

if (head == NULL) return false;

if (p == NULL) p = head;

LinkNode \* temp = p;

while ( ) // 空缺4

// 空缺5

while ( ) // 空缺6

// 空缺7

if (temp->value == key) {

p = temp;

return true;

}

else return false;

}

4、temp->getKey() < key && temp->next() （或者temp->next() != NULL没有考虑的扣0.5分）

5、temp = temp->next();

6、temp->getKey() > key && temp->prev()（或者temp->prev() != NULL没有考虑的扣0.5分）

7、temp = temp->prev();

|  |
| --- |
| 得分 |
|  |

## 第四题 设计题（每小题5分，共10分）

1. 一个具有n个有序数据的线性表，后面又插入了f(n) 个无序数据。针对下列三种情况：
   1. f(n) = O(1)
   2. f(n ) = O(log n)
   3. f(n) =，

给出对这样 n + f(n) 个数据的排序算法，并分析时间复杂度。

注释：不需要写具体算法，简单说一下算法思路就可以。

参考答案：

方案一，5分

使用二分法，将f(n)个无序数据分别插入线性表中，复杂度为f(n) \* O(log n)

答案分别是O(log n), O(log n log n), O(log n)

方案二，4分

a) 基本有序，使用插入排序算法，复杂度O(n)。

b,c) 先快排f(n)个数，再进行归并，复杂度O(f(n) log(f(n)) + O(n) = O(n)。

1. 营业额统计 Turnover

Tiger 最近被公司升任为营业部经理，他上任后接受公司交给的第一项任务 便是统计并分析公司成立以来的营业情况。Tiger 拿出了公司的账本，账本上记录了公司成立以来每天的营业额。分析营业情况是一项相当复杂的工作。由于节假日，大减价或者是其他情况的时候，营业额会出现一定的波动，当然一定的波动是能够接受的，但是在某些时候营业额突变得很高或是很低，这就证明公司此时的经营状况出现了问题。经济管理学上定义了一种最小波动值来衡量这种情况：

该天的最小波动值= min { | 该天以前某一天的营业额 - 该天的营业额 | }

当最小波动值越大时，就说明营业情况越不稳定。而分析整个公司的从成立到现在营业情况是否稳定，只需要把每一天的最小波动值加起来就可以了。你的任务就是编写一个程序帮助 Tiger 来计算这一个值。

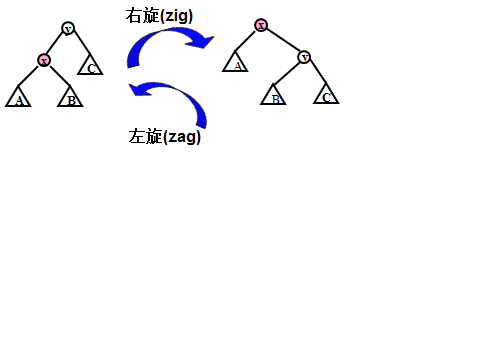
注释：第一天的最小波动值为第一天的营业额。数据范围：天数 n≤32767，每天的营业额 ai≤1,000,000。最后结果 T≤231。

提示：题目的意思非常明确，关键是要每次读入一个数，并且在前面输入的数中找到一个与该数相差最小的一个。我们很容易想到 O(n2)的算法：每次读入一个数，再将前面输入的数一次查 找一遍，求出与当前数的最小差值，记入总结果 T。但由于本题中 n 很大，这样 的算法是不可能在时限内出解的。而如果使用线段树记录已经读入的数，就需要记下一个 2M 的大数组，空间消耗太大。

红黑树与AVL树虽然在时间效率、空间复杂度上都比较优秀，但过高的编程复杂度却让人望而却步。建议采用伸展树。

答题要求：

1. 描述采用伸展树求解本问题的基本算法思路；
2. 利用一下的伸展树基本函数，写出主要算法框架。



伸展树是一种自平衡的BST，数据结构如下：

struct TreeNode {

int data;

TreeNode \* father,\* left,\* right;

};

可以用的函数为：

void zag(TreeNode\* y); // 对称的左旋(zag)和右旋(zig)，如右上图。

void zig(TreeNode\* x);

void Splay(x, f); // 将 x 旋转为 f 的子结点，f在 x 祖先路径

// 把 x 旋转到根结点即 Splay(x, NULL)

void Delete(TreeNode\* x); // 删除以x结点为根的子树

（1）算法思路（2分，利用Splay树的大致思路对也给分）

开始时，树S 为空，总和T 为零。每次读入一个数p，执行Insert(p,S)，将p插入伸展树S。（0.5分）这时，p 也被调整到伸展树的根节点。（0.5分）这时，求出p 点左子树中的最右点和右子树中的最左点，这两个点分别是有序集中p的前趋和后继。（0.5分）然后求得最小差值，加入最后结果T。（0.5分）

（2）算法框架（3分，0.5分\*6。给出伪代码即可）

初始化和整体情况

读入数据，执行Insert(p,S)，

调整p

求前驱

求后继

求最小差值，加入T

|  |
| --- |
| 得分 |
|  |

## 第五题 期中考试（共25分，同学们不必填写，由助教登记）

|  |
| --- |
| 得分 |
|  |

## 第六题 上机考试（共35分，同学们不必填写，由助教登记）