2023年 复旦大学姬算妓科学与技术学院《数据结构》期末试题（A卷）

整理：Gary Agasa

一、填空题

1. 规定数组下标起点为 0，一个 10 阶方阵 A 是对称方阵，按行优先方式仅保存其上三角到一维数组 B 中，则 B[50] 在 A 中的数组下标为 **[7, 8]**。
2. 用数组 q[m..n] 表示的顺序存储队列，用 front 表示队头指针，用 rear 表示队尾指针，在一个新元素要入队之前，如果表达式 `(rear==n)&&(front!=m-1)` 的值为 true，就可以确定队列将发生假溢出，这时队列中的元素个数为 **rear - front**。
3. 给定结点数为 `n` ,高度为 $h = \log_2(n+1) - 1$ 的满二叉树，如果从 0 开始按中序遍历顺序对结点编号，则树根的左孩子编号用 h 表示为 $2^{h-1} - 1$。
4. 如果二叉树中结点 n 的左右孩子都非空，则 n 的中序后继 p 的 **左** 孩子一定为空。
5. 规定根结点高度为 0，如果插入新结点后，一棵 AVL 树的高度为 2，并且由于右子树为空导致不平衡，需要进行左右双旋，旋转后该 AVL 树的根结点的平衡因子值为 **0**。
6. n 个结点 e 条边的无向图用邻接表表示时，空间复杂度是 **O(n+e)**。
7. Kruskal 算法为能快速确定是否加入一条边，一般会采用称为 **并查集** 的数据结构。
8. 在 n 个数中用堆排序选择最小的 k 个数 (k < n)，时间性能最好的算法时间复杂度可达到 $O(n \log_2 k)$。
9. 冒泡排序法、快速排序法、堆排序法和二路归并排序法四种排序法中，要求辅助空间最多的方法是 **二路归并排序法**。

二、选择题

1. n 个结点的链表，每个结点中保存一个整数值。如果设计一个算法 g 根据给定的一个整数 p 整理链表，使得所有保存的值小于 p 的结点在新链表的左边，等于 p 的结点在中间，大于 p 的结点在链表的右边。例如，初始链表为 [8 -> 2 -> 0 -> 2 -> 3 -> 8 -> 18]，给定整数 3，整理后的链表为 [2 -> 0 -> 2 -> 3 -> 8 -> 18 -> 8]。限制 g 的空间复杂度为 O(1)，则最优的时间复杂度可以达到：

- A. $O(n^2)$
- B. $O(n \log_2 n)$
- C. $O(n)$
- D. $O(1)$

- **答：C** [说明：根据 p 遍历一遍，根据与 p 的比较结果分别连入 3 个链表之中，最后把 3 个链表合并即可。]

2. 如果 T 是一棵有 n 个结点的 k 叉树，其中 $n \geq 1$，则树上共有 $n * (k-1) + 1$ 个空指针。如果用孩子兄弟链法把这棵树改为二叉树表示，则空指针个数为：

- A. n*(k-1)+1
- B. n*(k-1)
- C. n+1
- D. 2*n
- **答：C**

3. 根据输入的 n 个数据构造一棵 AVL 树的时间复杂度是：

- A. $O(n)$
- B. $O(n \log_2 n)$
- C. $O(n^2)$
- D. $O(n + \log_2 n)$
- **答：B**

4. 修改递归方式实现的图深度优先搜索算法，将输出（访问）顶点信息的语句移到退出递归前（即执行输出语句后立刻退出递归）。采用修改后的算法遍历有向无环图 G，若输出结果中包含的全部顶点，则输出的顶点序列是 G 的：

- A. 拓扑有序序列
- B. 逆拓扑有序序列
- C. 深度优先搜索序列
- D. 广度优先搜索序列
- **答：B**

5. 希尔排序利用了 _____ 在短序列和接近有序序列上性能较优的特点。

- A. 归并排序
- B. 快速排序
- C. 直接插入排序
- D. 冒泡排序
- **答：C**

三、问答题

1. 关于完全二叉树堆的调整（6分）

题目： 如果用三叉链表表示的完全二叉树表示堆，每个结点有 3 个指针 (pr, left, right)。给定 last 指针指向堆的最后一个结点。当向堆中添加一个新元素 v 时，需要找到 v 在堆中初始的双亲结点 p。请问怎样找到这个结点 p?

答：

- **情况 1：** last 所指结点为 NULL，则 v 的双亲为空，v 为堆顶。
- **情况 2：** last -> pr == NULL，则 v 是 last 的左孩子。否则，如果 last = last -> pr -> left，则 v 就是 last -> pr 的右孩子。
- **情况 3：** 从 last 开始逐层顺着 pr 指针向上，初始时 p = last，每次检查：
- `if (p->pr == NULL)`：新结点应作为整棵树最左结点的左孩子。
- `else if (p != p->pr->left) p = p->pr;`
- `else`：新结点应该作为以 `p->pr->right` 为根的子树的最左结点的左孩子（即从 `p->pr->right` 开始沿着 left 指针一直到遇到 left 指针为空的结点）。

2. 散列表设计（6分）

题目： 假设散列表中已装入 100 个表项，采用线性探测法。要求搜索表中已有表项时的平均搜索次数不超过 4，插入表中没有的表项时平均探测次数不超过 50.5。求散列表容量及除留余数法的散列函数。

公式： $S_n \approx \frac{1}{2}(1+\frac{1}{1-\alpha}), U_n \approx \frac{1}{2}(1+\frac{1}{(1-\alpha)^2})$

解题思路：

- 根据 $S_n \leq 4 \Rightarrow \frac{1}{2}(1+\frac{1}{1-\alpha}) \leq 4 \Rightarrow \alpha \leq 6/7 \approx 0.857$
- 根据 $U_n \leq 50.5 \Rightarrow \frac{1}{2}(1+\frac{1}{(1-\alpha)^2}) \leq 50.5 \Rightarrow (1-\alpha)^2 \geq 1/100 \Rightarrow \alpha \leq 0.9$
- 取 $\alpha = 0.857$，则 $m = n/\alpha = 100/0.857 \approx 116.6$。
- 选取不大于 m 的质数，取 p = 113。故 $Hash(key) = key\%113$。

3. 折半搜索改进（6分）

题目： 对有序表 data 进行折半搜索，返回 data 中值为 n 的 **第一个** 元素的下标, 如果没有找到就返回-1。要求对给定代码进行改进，并说明改进的理由。

> **Hint**:
>
> 1. `data` 中可能有值相同的元素，且 `data` 中数据分布不均匀
> 2. 要求改进时，任然保持折半搜索策略

```
int search(int data[], int left, int right, int n){
    int low = left, high = right, tmp;
    while(low <= high){
        tmp = (low + high) / 2;
        if(n == data[tmp]) return tmp;
        if(n > data[tmp]) low = tmp + 1;
    }
    return -1;
}
```

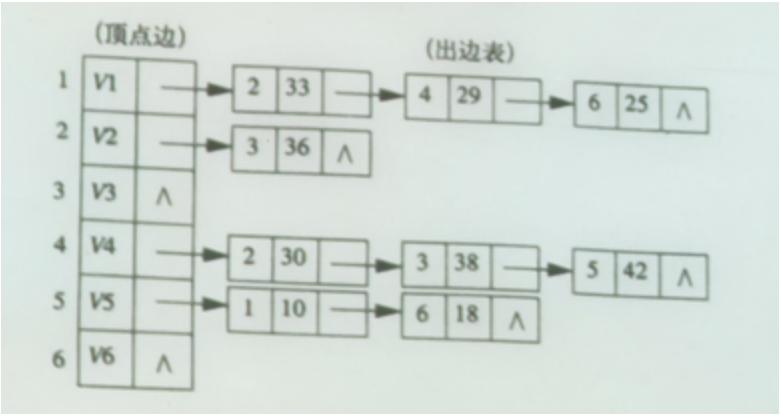
改进要点：

- 当 `data[tmp] == n` 时，不能直接返回，因为前面可能还有相同的 n。
- 应修改第 5-7 行行为：

```
if (data[tmp] >= key) {
    if (low < tmp) high = tmp;
    else return tmp;
} else low = tmp + 1;
```

4. 图的邻接表与搜索（6分）

题目： 给定带权有向图的邻接表：



(1) 画出该图形；
(2) 给出以 V1 为起点的广度优先遍历序列及生成树；
(3) 由 V1 到 V3 的最短路径。

四、算法设计题

1. 二叉搜索树后序遍历校验（15分）

题目：

输入一个整数数组，判断该数组是不是某棵二叉搜索树后序遍历的结果。如果是则返回true，否则返回false。例如，输入5、7、6、9、11、10、8，由于这一整数序列是如下图所示二叉搜索树的后序遍历结果，则返回true；如果输入7、4、6、5，没有哪棵二叉搜索树的后序遍历结果是该序列，则返回false。假设序列中没有重复元素。（15分）

核心思路： 后序遍历序列最后一位是根。序列可分为两部分：左子树部分全小于根，右子树部分全大于根。递归校验左右子部分。

评分标准：

总共 15 分。

思路 5 分——思路清晰正确为 3 分，方法不限于参考答案这一种，思路不完全正确最多为 2 分，没写为 0 分；

算法 10 分——算法结构基本正确为 2 分，部分正确为 4 分，全部正确为 7 分。

若没有判断空序列，扣 1 分。

2. 最小生成树（曼哈顿距离）（15分）

题目： 给定 2D 平面上的点 points，连接两点的代价为曼哈顿距离 $|x_i - x_j| + |y_i - y_j|$。求连接所有点的最小总代价（最小生成树）。

要求： (1) 描述设计思想（通常使用 Prim 或 Kruskal 算法）；(2) 给出代码；(3) 分析复杂度。

3. 排序

题目： 设计一个尽量快的算法统计整数序列$\{a_1, a_2, a_3, ..., a_n\}$中逆序对的个数。例如，整数数组$\{2, 0, 1, 5\}$中有2个逆序对$(2, 0)$
和$(2, 1)$

解答

答：设数组为A，用数组L 和 R 表示左右两个子数组，逆序对的个数 f(A)=f(L)+f(R)+s(L, R)；其中 f(L)和 f(R)分别表示 L 内部和 R 内部的逆序对个数，s(L, R)表示大数在 L，小数在 R 的逆序对。如果 L 和 R 是已经排好序的，就只需求 s(L, R)，这个可以在合并 L 和 R 依次进行比较的时候算出。可以改造归并算法来实现，时间复杂度 O(nlogn)。

算法实现如下：

```
#include "stdafx.h"
#include <iostream>
#include <limits>

using namespace std;
```