上 海 交 通 大 学 试 卷（B 卷）

（20_22_ 至 20_23_ 学年 第_1_ 学期）

班级号 _____ 学号 _____ 姓名 _____

课程名称 CS0501 数据结构 成绩 _____

一、选择题（每题 1 分，共 15 分）

1. 如果一棵二叉树有 85 个结点，则这棵二叉树的高度最少是?

A. 5 B. 6 C. 7 D. 8

答案 C

解析 一棵高度为 h 的二叉树，其节点数最多为 2^h-1。考虑 $2^6-1<85\le 2^7-1$，因此高度最少是 7。

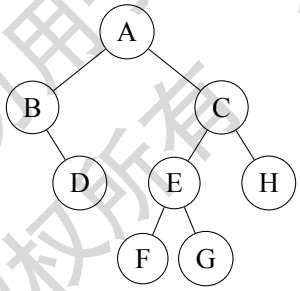
2. 如果一个二叉树的前序遍历为 ABDCEFGH，中序遍历为 BDAFEGCH，则其后序遍历为?

A. BDFGEHCA B. DBFGEHAC C. BDFGAEHC D. DBFGEHCA

答案 D

解析 根据前序（或后序）遍历找根所在位置，再由中序遍历确定左右子树的元素，可逐步构造出整棵树。

1. A 为根，则中序遍历分割为 (BD)A(FEGCH)，前序遍历 A(BD)(CEFGH)；
2. B 为左子树的根，则中序遍历 (B(D))A(FEGCH)，前序遍历 A(B(D))(CEFGH)；
3. C 为右子树的根，则中序遍历 (B(D))A((FEG)C(H))，前序遍历 A(B(D))(C(EFG)(H))；
4. E 是 C 的左子树的根，中序遍历 (B(D))A(((F)E(G))C(H))，前序遍历 A(B(D))(C(E(F)(G))(H))，整棵树构造完毕。



得到后序遍历 DBFGEHCA。

3. 通过调整数组而非逐一插入，创建一个堆的时间复杂度是?

A. $O(N)$ B. $O(\log N)$ C. $O(N\log N)$ D. $O(N^2)$

答案 A

解析 设二叉树的高度为 h，最多有 2^h-1 个元素，其中叶子节点 2^{h-1} 个。则有 2^{h-2} 个节点最多向下调整 1 次，2^{h-3} 个节点最多向下调整 2 次，……，2^0 个节点最多向下调整 $h-1$ 次。则总调整次数

$$\sum_{i=1}^{h-1} 2^{h-i-1} = 2^{h-1}-1$$

对于 N 个元素的二叉树，其高度 $h=\lfloor \log_2 N\rfloor+1$，代入得

$$T(N)=2^{\lfloor \log_2 N\rfloor+1-1}=O(N)$$

仅限上海交通大学数据结构学科营使用
请勿用于其他用途
翻录必究
版权所有

上海交通大学数据结构学科营 版权所有

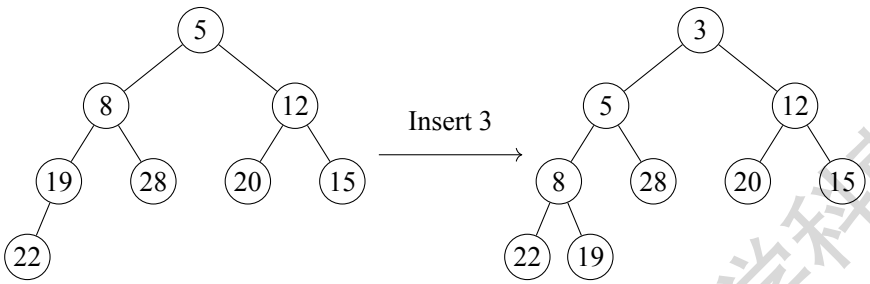
4. 已知关键字序列 {5，8，12，19，28，20，15，22} 是最小堆（小根堆、小顶堆），插入关键字 3，调整后得到的最小堆是?

A. 3, 5, 12, 8, 28, 20, 15, 22, 19　　B. 3, 5, 12, 19, 20, 15, 22, 8, 28

C. 3, 8, 12, 5, 20, 15, 22, 28, 19　　D. 3, 12, 5, 8, 28, 20, 15, 22, 19

答案 A

解析



5. 在下列方法中，平均查找时间与结点个数 n 无关的是?

A. 顺序查找　　B. 折半查找　　C. 分块查找　　D. 哈希查找

答案 D

解析 顺序查找平均查找时间 $\frac{n}{2}=O(n)$;

折半查找平均查找时间 $\lfloor\log_2 n\rfloor-1=O(\log n)$;

分块查找（假设分块数量为 m）平均查找时间 $\frac{1}{2}\left(m+\frac{n}{m}\right)+1$，当 $m=\sqrt{n}$ 时最优，复杂度为 $O(\sqrt{n})$;

哈希查找平均查找时间 $O(1)$，与节点个数无关。

6. 某线性表中最常用的操作是在最后一个元素之后插入一个元素和删除第一个元素，则采用哪种存储方式最节省运算时间和空间?

A. 单链表　　B. 仅有头指针的单循环链表

C. 双链表　　D. 仅有尾指针的单循环链表

答案 D

解析

	末尾插入元素	删除首个元素	空间占用
单链表	$O(n)$	$O(1)$	少
仅有头指针的单循环链表	$O(n)$	$O(1)$	少
仅有尾指针的单循环链表	$O(1)$	$O(1)$	少
双链表	$O(1)$	$O(1)$	多

7. 设用一维数组 `a[n]` 实现顺序栈，以 `a[n - 1]` 为栈底，用整型变量 `t` 指示当前栈顶位置，`a[t]` 为栈顶元素。当从栈中成功弹出一个元素时，变量 `t` 的变化是?

A. `t = t - 1`　　B. `t = t + 1`　　C. `t` 保持不变　　D. `t = n`

答案 B

解析 弹出操作后，栈顶元素出栈，栈顶位置向栈底方向移动，即 `t = t + 1`。

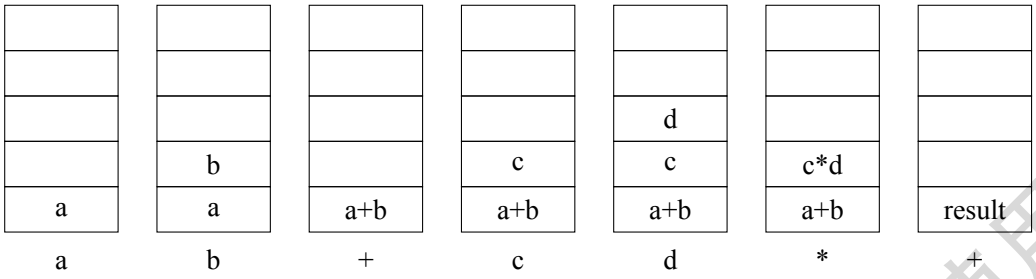
8. 利用栈，求后缀表达式 `ab+cd*+` 的值，问操作数栈至少有几个存储单元，以避免发生上溢?

A. 2　　B. 3　　C. 4　　D. 以上都不对

答案 B

仅限上海交通大学数据结构学科营使用
请勿用于其他用途
版权所有 翻录必究

解析



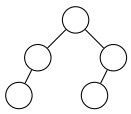
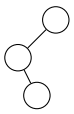
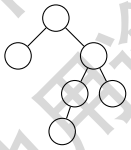
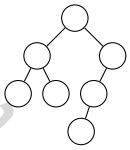
9. 若用数组 `A[0...5]` 来实现循环队列，且当前 `rear` 和 `front` 的值分别为 1 和 5，当从队列中出队一个元素，再进队两个元素后，`rear` 和 `front` 的值分别为?

A. 3, 4　　B. 5, 1　　C. 5, 0　　D. 3, 0

答案 D

解析 队列入队向队尾增加元素，出队从队首提取元素。初始时 `rear = 1; front = 5;` 出队一个元素后 `rear = 1; front = 0;` 入队一个元素后 `rear = 2; front = 0;` 入队第二个元素后 `rear = 3; front = 0`。

10. 对下列二叉查找树，AVL 树是?

A.  　　B.  　　C.  　　D. 

答案 A

解析 AVL 树要求每个节点左右子树的高度差不超过 1。

A 选项各节点均满足条件;

B 选项左子树高度为 2，右子树高度为 0，高度差为 2;

C 选项左子树高度为 1，右子树高度为 3，高度差为 2;

D 选项右子树的左子树高度为 2，其右子树高度为 0，高度差为 2。

11. 若外部存储上有 32400 个记录，做 3 路平衡归并排序，计算机内存工作区能容纳 400 个记录，则排序好所有记录，需要作多少趟归并排序?

A. 2　　B. 4　　C. 5　　D. 3

答案 B

解析 N 个有序片段做 k 路平衡归并排序，共需要排序 $\lceil \log_k N \rceil$ 趟。内存工作区能容纳 400 个记录，则共有 $32400/400 = 81$ 个有序片段，排序趟数为 $\lceil \log_3 81 \rceil = 4$。

12. 在有 8 个顶点的无向图中，至少需要多少条边它才可能是连通的?

A. 3　　B. 6　　C. 7　　D. 9

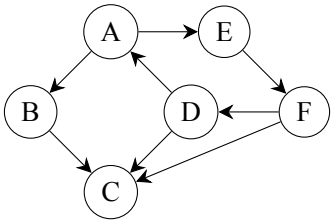
答案 C

解析 n 个顶点的无向图，至少需要 $n-1$ 条边才可能联通，此时构成一棵树。

13. 在如下所示的有向图中，下列哪项不是该图的广度优先搜索结果?

A. ABECFD　　B. BCAEFD　　C. CBAEFD　　D. AEFDCB

仅限上海交通大学数据结构学科营使用
版权所有 翻录必究 禁用于其他用途



答案 D

解析 D 选项先访问 A，此时 E、B 入队列，然后访问 E，F 入队列，根据 BFS 的搜索顺序，接下来应该访问 B 而不是 F，故错误。其余选项均符合 BFS 搜索顺序。

14. 下列内部排序算法中，最坏情况下时间复杂度为 $O(n\log n)$ 是?

A. 冒泡排序　　B. 插入排序　　C. 堆排序　　D. 快速排序

答案 C

解析

	最好时间复杂度	最差时间复杂度	平均时间复杂度	空间复杂度	稳定性
冒泡排序	$O(n)$	$O(n^2)$	$O(n^2)$	$O(1)$	稳定
直接插入排序	$O(n)$	$O(n^2)$	$O(n^2)$	$O(1)$	稳定
二分插入排序	$O(n\log n)$	$O(n^2)$	$O(n^2)$	$O(1)$	稳定
希尔排序			$\sim O(n^{1.25})$	$O(1)$	不稳定
选择排序	$O(n^2)$	$O(n^2)$	$O(n^2)$	$O(1)$	不稳定
堆排序	$O(n\log n)$	$O(n\log n)$	$O(n\log n)$	$O(1)$	不稳定
快速排序	$O(n\log n)$	$O(n^2)$	$O(n\log n)$	$O(\log n)$	不稳定
归并排序	$O(n\log n)$	$O(n\log n)$	$O(n\log n)$	$O(n)$	稳定
基数排序	$O(nk)$	$O(nk)$	$O(nk)$	$O(1)$	

15. 已知序列 {515，99，524，73，920，182，909，287，665，474}，采用 2 路归并排序法对该序列做升序排序时需要几趟排序?

A. 2　　B. 3　　C. 4　　D. 5

答案 C

解析 归并排序的趟数为 $\lceil\log_2 n\rceil = \lceil\log_2 10\rceil = 4$。

二、简答题（每题 5 分，共 40 分）

1. 假设需要编码的字符及其出现的频度为：

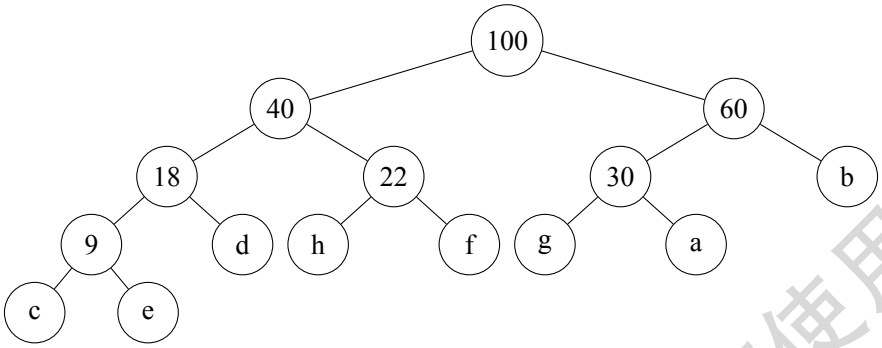
a：16，b：30，c：1，d：9，e：8，f：12，g：14，h：10

请问在按照哈夫曼算法构造的哈夫曼树中，f 在第几层（假设根为第一层）？如果哈夫曼树中左分支为 1，右分支为 0，那么 b 的编码长度是多少?

答案 建立哈夫曼树。

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
值									a	b	c	d	e	f	g	h
权值		100	60	40	30	22	18	9	16	30	1	9	8	12	14	10
父			1	1	2	3	3	6	4	2	7	6	7	5	4	5
左		3	4	6	14	15	7	10								
右		2	9	5	8	13	11	12								

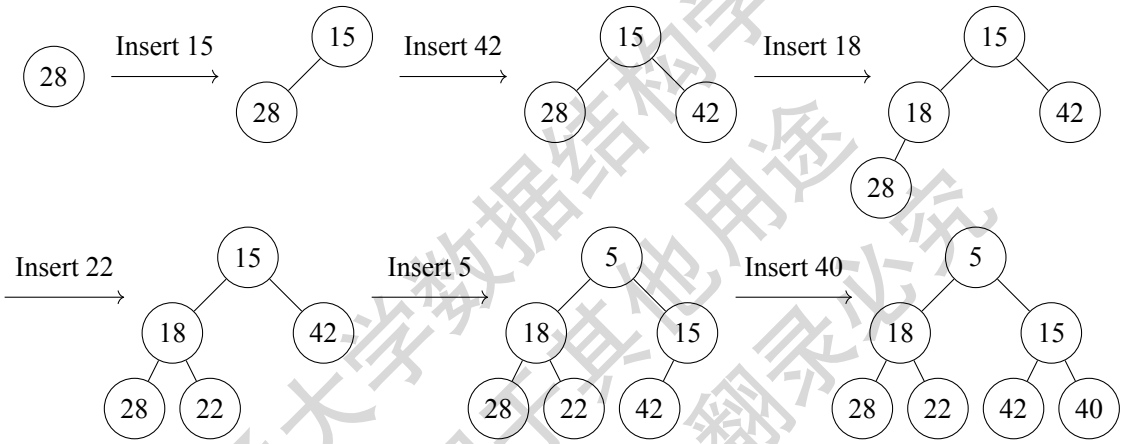
仅限上海交通大学数据结构学科营使用
请勿用于其他用途
版权所有 翻录必究



则 f 在第 4 层，b 的编码长度为 2 位。

2. {28, 15, 42, 18, 22, 5, 40} 逐个按顺序插入到初始为空的最小化堆中，则该二叉树的前序遍历结果为：

答案 按顺序插入小根堆。



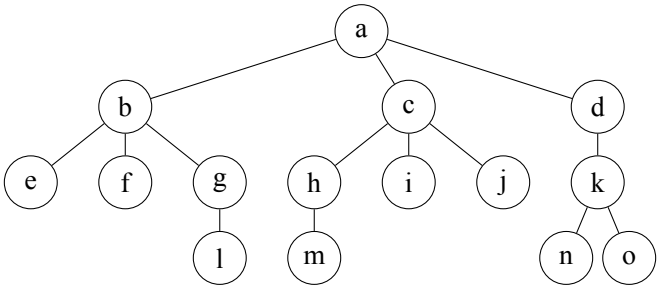
因此前序遍历为 {5, 18, 28, 22, 15, 42, 40}。

3. 设入栈序列为 A、B、C，且在入栈的过程中允许出栈，试写出所有可能得到的出栈序列，并写出每种序列对应的操作。例如：A 入 A 出，B 入 B 出，C 入 C 出，即 ABC。

解析 考虑每一步执行入栈或出栈，递归地搜索每种可能性。

答案 A 入，A 出，B 入，B 出，C 入，C 出，即 ABC；
A 入，A 出，B 入，C 入，C 出，B 出，即 ACB；
A 入，B 入，B 出，A 出，C 入，C 出，即 BAC；
A 入，B 入，B 出，C 入，C 出，A 出，即 BCA；
A 入，B 入，C 入，C 出，B 出，A 出，即 CBA。

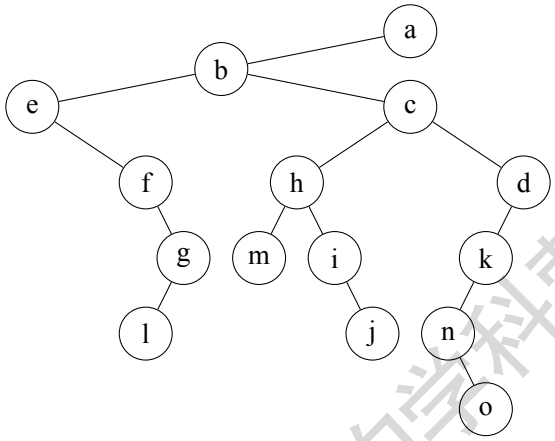
4. 利用孩子兄弟链的方法将下图中的树转换成二叉树的形式。



仅限上海交通大学数据结构学科营使用
请勿用于其他用途
翻录必究
版权所有

解析 孩子兄弟链法中，每个节点的左子节点为原树中的第一个孩子节点，右子节点为原树中的下一个兄弟节点。

答案 见下图。



5. 若哈希表的表长为 16，采用除留取余法确定哈希函数，并用线性探测法处理冲突。

对于关键字序列：(10，24，58，17，31，30，47，40，63，13，49，6)，请给出哈希函数和哈希表的存储示意图。

答案 采用除留余数法，取不大于表长的最大素数 $M = 13$，则哈希函数 $H(x) = x \bmod 13$。

哈希表插入过程如下所示（加粗表示哈希冲突）。

插入 x	H(x)	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
10	10											10					
24	11											10	24				
58	6							58				10	24				
17	4					17		58				10	24				
31	5					17	31	58				10	24				
30	4					**17**	31	58	30			10	24				
47	8					17	31	58	30	47		10	24				
40	1		40			17	31	58	30	47		10	24				
63	11		40			17	31	58	30	47		10	**24**	63			
13	0	13	40			17	31	58	30	47		10	24	63			
49	10	13	40			17	31	58	30	47		**10**	24	63	49		
6	6	13	40			17	31	**58**	30	47	6	10	24	63	49		

因此，哈希表的存储示意图如图所示。

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
13	40			17	31	58	30	47	6	10	24	63	49		

6. 试分析简单插入排序的最优情况与最差情况的时间复杂度，并判断该排序算法的稳定性。

答案 简单插入排序最优情况为输入序列为有序序列，此时不需要插入操作，只需要考虑比较操作。对于每趟排序，只需比较 1 次。因此时间复杂度 $T(n) = n - 1 = O(n)$。

最差情况时输入序列为逆序序列，对于每趟排序需要和该元素前的每一个元素都交换一次，因此时间复

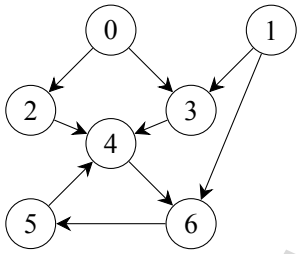
仅限上海交通大学数据结构学科营使用 请勿用于其他用途 版权所有 翻录必究

杂度为

$$T(n)=\sum_{i=1}^{n-1} i=\frac{n(n-1)}{2}=O(n^2)$$

插入排序是稳定的。

7. 试问，以下有向图是否存在拓扑序列？如果存在，给出一个拓扑序列；如果不存在，分析其原因。



答案 不存在。图中存在 $4\to6\to5\to4$ 的环路，只有有向无环图才存在拓扑序列。

8. 如果一个外存储器上的结构型数据文件，使用一棵高度为 3（即不算空叶子层的索引结点层数为 3）的 5 阶 B 树作为数据记录关键字索引，该数据文件中最少允许多少个记录？写出分析过程。

答案 m 阶 B 树的根节点至多有 m 个子节点，至少有 2 个子节点或为叶子节点。中间节点的子节点数 s 满足 $\lceil m/2\rceil\le s\le m$。当除根节点外所有的节点均有 $\lceil m/2\rceil$ 个子节点，且根节点有 2 个子节点时，数据记录数量最少。此时根节点有 1 个数据记录，其余节点有 $\lceil m/2\rceil-1$ 个数据，第 i 层（$i>1$，设根节点在第 1 层）有 $2\times\lceil m/2\rceil^{i-2}$ 个节点。因此总记录数为

$$N=1+\sum_{i=2}^{h}2\times\left\lceil\frac{m}{2}\right\rceil^{i-2}\times\left(\left\lceil\frac{m}{2}\right\rceil-1\right)=2\times\left\lceil\frac{m}{2}\right\rceil^{h-1}-1$$

代入 $m=5$，$h=3$ 得最多记录数 $N=17$。

三、程序填空（每空 2 分，共 20 分）

1. 下面的程序用于删除二叉树以 t 为根节点一个子树，请补充完整其中确实的代码。

```
template<class T>
void binaryTree<T>::clear(binaryTree<T>::Node ____(1)____)
{
    if (____(2)____) return;
    clear(t->left);
    clear(t->right);
    ____(3)____
    t = NULL;
}
```

解析 程序实现了删除二叉树以 t 为根节点的子树，t 应为 (1) 处传入的参数，根据 -> 运算符可知其为指针类型，由于最后要将 t 赋值为空并影响到调用方，t 还应为引用类型。(2) 处应判断是否为空，若不为空则递归删除左右子树。(3) 处应回收当前对象 t。

答案 (1) `*&t`

仅限上海交通大学数据结构学科营使用 请勿用于其他用途 版权所有 翻录必究

(2) t == NULL
(3) delete t;

2. 以下程序段为链式栈弹出栈顶元素的算法实现，请在空白处补充语句。

```
template <class elemType>
elemType linkStack<elemType>::pop()
{
    if (_____(1)_____) throw outOfBound();
    node *tmp = top_p;
    elemType x = tmp->data;
    _____(2)_____
    delete tmp;
    _____(3)_____
}
```

解析 程序为链式栈弹出元素的实现，其中 top_p 为栈顶元素指针，tmp 是栈顶指针的临时变量，在 (2) 处需要后移栈顶指针到下一个元素。x 为临时保存的元素值，应在 (3) 处返回。(1) 处需要判断栈是否为空。

答案 (1) top_p == NULL
(2) top_p = top_p->next;
(3) return x;

3. 以下为在采用邻接表存储结构的图中进行非递归深度优先搜索的算法实现，请在空白处补充语句。

```
template <class TypeOfVer, class TypeOfEdge>
void adjListGraph<TypeOfVer, TypeOfEdge>::DepthFirstSearch() const
{
    int i;
    linkStack<edgeNode*> stkNode;
    edgeNode* p;
    bool* visited = new bool[Vers];
    for (i = 0; i < Vers; i++) _____(1)_____
    for (i = 0; i < Vers; i++) {
        if (visited[i]) continue;
        std::cout << verList[i].ver << " ";
        visited[i] = true;
        stkNode.push(verList[i].head);
        do {
            _____(2)_____
            while (p) {
                if (!visited[p->end]) break;
                p = p->next;
            }
            if (p) {
                _____(3)_____
```

仅限上海交通大学数据结构学科营使用 请勿用于其他用途 版权所有 翻录必究

上海交通大学数据结构学科营 版权所有

```
                std::cout << verList[p->end].ver << " ";
                visited[p->end] = true;
                stkNode.push(verList[p->end].head);
            }
        } while (!stkNode.isEmpty());
    }
    ____________(4)_____________
    std::cout << endl;
}
```

解析 经典的深度优先搜索为递归算法，将递归算法转化为非递归的算法关键在于用栈模拟递归函数调用的过程，在栈中需要保存每一次调用的上下文数据，在调用结束后能根据栈中保存的数据继续运行原来的函数内容。

本题程序实现了非递归的深度优先搜索，图采用邻接表存储。`visited` 为标记顶点访问状态的数组，应在 (1) 处初始化为 `false`，并在 (4) 处回收。

对于每个顶点，若没有被访问过则以该顶点为起始深度优先遍历，将该顶点的第一条出边加入到栈中并设置访问状态。对于每轮深度优先遍历，从栈中取一条边 $p = \langle u, v \rangle$，从 p 开始继续遍历 u 的每一条边直到其指向的节点未被访问过或遍历结束。若未遍历结束，(3) 处将此时的 p 放回栈中用于在回溯后继续遍历 u，然后将 u 的第一条出边加入到栈中，模拟递归的过程。

答案 (1) `visited[i] = false;`
(2) `p = stkNode.pop();`
(3) `stkNode.push(p);`
(4) `delete[] visited;`

四、编程题（共 25 分）

1. 编写函数 `bool judge(node *t) const`，判断以 `t` 为根的二叉树是否是 AVL 树（假设结点中含有高度 `height` 字段）（5 分）。

```
int height(node *t) const
{ if (!t) return 0; return t->height; }
```

解析 AVL 树要求每个结点的左右子树的高度差均不超过 1，采用深度优先的遍历算法判断每个结点是否满足要求。

答案

```
bool judge(node *t) const {
    if (!t) return false;
    int h = height(t->left) - height(t->right);
    return (h >= -1 && h <= 1) && judge(t->left) && judge(t->right);
}
```

2. 请写出二叉树层次遍历的实现代码 `void binaryTree<T>::levelOrder() const`，要求遍历按照从上层到下层，每一层从右到左的方式访问。（10 分）

解析 要求按从上到下、从右到左的层次遍历，采用广度优先搜索，且右孩子先入队。

仅限上海交通大学数据结构学科营使用
请勿用于其他用途
翻录必究
版权所有

答案

```
template <class T>
void binaryTree<T>::levelOrder() const {
    Queue<typename binaryTree<T>::Node *> q;
    q.enQueue(this->root);
    while (!q.isEmpty()) {
        typename binaryTree<T>::Node *cur = q.deQueue();
        std::cout << cur->data << ' ';
        if (cur->right) q.enQueue(cur->right);
        if (cur->left) q.enQueue(cur->left);
    }
}
```

3. 一个无向图（不带权值）的邻接表类定义如下：

```
template <class TypeOfVer, class TypeOfEdge>
struct edgeNode {
    int end;
    edgeNode* next;
    edgeNode(int e, edgeNode* n = NULL) : end(e), next(n) {}
};

template <class TypeOfVer, class TypeOfEdge>
struct verNode {
    TypeOfVer ver;
    edgeNode* head;
    verNode(edgeNode *h = NULL) : head(h) {}
};

template <class TypeOfVer, class TypeOfEdge>
class adjListGraph
{
private:
    verNode* verList; //图中结点组成的动态数组
    int Vers, Edges; //图中结点数及边数
    int find(TypeOfVer v) const; //查找 v 在图中的内部序号
public:
    void allVetexes(TypeOfVer v) const;
};
```

请编码实现函数 `void allAdjVetexes(TypeOfVer v) const`，要求利用非递归算法，输出和顶点 v 没有路径相连的所有顶点的值及个数。(10 分)

解析 题目采用邻接表存储无向图，要求输出和顶点 v 没有路径相连的所有顶点的值和个数。要求使用非递归算法，考虑广度优先搜索，遍历以 v 为起点的广度优先生成树，则除这棵树以外的所有顶点均没有与 v

仅限上海交通大学数据结构学科营使用
请勿用于其他用途
版权所有，翻录必究

相连的路径。

答案

```
template <class TypeOfVer, class TypeOfEdge>
void adjListGraph<TypeOfVer, TypeOfEdge>::allAdjVetexes(TypeOfVer v) const {
    Queue<int> q;
    bool *visited = new bool[this->Vers];
    for (int i = 0; i < this->Vers; ++i) visited[i] = false;
    q.enQueue(this->find(v));
    while (!q.isEmpty()) {
        int cur = q.deQueue();
        visited[cur] = true;
        for (
            edgeNode<TypeOfVer, TypeOfEdge> *e = this->verList[cur].head;
            !e;
            e = e->next
        ) {
            if (!visited[e->end]) {
                q.enQueue(e->end);
            }
        }
    }

    int count = 0;
    for (int i = 0; i < this->Vers; ++i) {
        if (!visited[i]) {
            std::cout << verList[i].ver << ' ';
            ++count;
        }
    }
    std::cout << count << std::endl;

    delete[] visited;
}
```

仅限上海交通大学数据结构学科营使用
请勿用于其他用途
版权所有 翻录必究