221900180 田永铭

Lesson 4 Practice：（多维数据、字符串、广义表、二叉树部分）

一、填空题

1. 空串与空格串的区别在于\_\_\_\_\_空串是不含任何字符的串，长度为0；空格串是由一个或多个空格字符组成的串，长度大于等于1\_\_\_\_\_。

2. 两个字符串相等的充要条件是“两个串长度相等且\_\_每个对应位置上的字符都要相等\_\_\_\_\_\_\_\_”。

3. 设S=“I\_am\_a\_student”，其长度为\_\_\_\_\_14\_\_\_\_\_。

4. 设有一个10阶的对称矩阵A，如果采用下三角压缩存储，A[0][0]为第一个元素，其存储地址为100，每个元素的长度是一个字节，则元素A[8][5]的存储地址为（ 141 ）。

5. 广义表LS=( (a) , (( (b),c )), (d) )，其长度是（ 3 ），深度是（ 4 ），表头是（ （a） ），表尾是（ ((( (b),c )), (d)) ）。(防止括号看不清，我的答案标红了)

6. 广义表LS=（a，（b，c，d），e），如果采用函数Head( )和Tail( )取出LS中的原子b，则运算是（ Head(Head(Tail(LS))) ）。

7. 对于一棵左子树为空的二叉树进行线索化后（不加入表头结点），其中序线索树有 2 个空指针，前序线索树有 2 个空指针，后序线索树有 1或者2（根节点有右子树的时候为1） 个空指针。

8. 高度为h（h>=0）的满二叉树上，共有 个结点，其中叶子结点为

(加了下取整函数使得答案可以包含h等于的的空树的特殊情况)（h = 0时为0，h不等于0时候为）

个。

9. 高度为h（h>=0）的完全二叉树至少有\_\_\_\_\_\_\_(若h==0 ,答案为0;若h==1，答案为1；若h>=2,答案为)\_\_\_\_\_\_\_\_\_\_\_个叶子结点。

二、选择题

1. 设有两个串p和q，求p在q中首次出现的位置的运算称作（ B ）。

（A）连接

（B）模式匹配

（C）求子串

（D）求串长

2. 串是一种特殊的线性表，其特殊性体现在（ B ）。

（A）可以顺序存储

（B）数据元素是一个字符

（C）可以链式存储

（D）数据元素可以是多个字符

3. 在顺序串中，根据空间分配方式的不同，可以分为（ B ）。

（A）直接分配和间接分配

（B）静态分配和动态分配

（C）顺序分配和链式分配

（D）随机分配和固定分配

4. 下列关于串的叙述中，正确的是（ A ）。

（A）一个串的字符个数即该串的长度

（B）一个串的长度至少是1

（C）空串是由一个或多个空格字符组成的串

（D）如果两个串S1和S2的长度相同，则这两个串相等

5. 串下面关于串的的叙述中，（B ）是不正确的？

A．串是字符的有限序列 B．空串是由空格构成的串

C．模式匹配是串的一种重要运算 D．串既可以采用顺序存储，也可以采用链式存储

6. 设串S1=“ABCDEFG“，S2=”PQRST“，

通过连接操作StrConcat(&T，S1，S2) 返回S1和S2连接而成的新串T，

通过求子串操作SubString(&Sub，S，i，j) 返回S从第i个(i>=1)字符起长度为j的子串Sub，

通过求长度操作StrLength(S) 返回S的长度；

那么执行操作StrConcat(T，SubString(Sub1，S1，2，StrLength(S2) )，SubString(Sub2，S1，StrLength(S2)，2) )后，T是（ D ）。

（A）BCDEF

（B）BCDEFG

（C）BCPQRST

（D）BCDEFEF

7. 将数组称为随机存储结构是因为（ B ）。

（A）数组元素是随机的

（B）对数组任意元素存取时间相等

（C）随时可以对数组进行访问

（D）数组的存储结构是不定的

8. 二维数组A的每个元素是8个字节组成的双精度实数，行下标的范围是[0,7]，列下标的范围是[0,9]，则存放A至少需要（ D ）个字节。

（A）80

（B）144

（C）504

（D）640

9. 对特殊矩阵采用压缩存储的目的主要是为了（ D ）。

（A）表达变得简单

（B）对矩阵元素的存取变得简单

（C）去掉矩阵中的多余元素

（D）减少不必要的存储空间

10. 如果广义表LS满足GetHead（LS）=GetTail（LS），则LS为（ B ）。

（A）（）

（B）（（））

（C）（（），（））

（D）（（），（），（））

【解答题】以下程序结果是什么:

#include <string.h>

int main(void){

char name[ ] = "ALEX";

char \*ptr;

ptr = name + strlen(name);

while(--ptr >= name)

puts(ptr);

return 0;

}

答案：

X

EX

LEX

ALEX

【画图】

画出下面广义表的链表存储的示意图：

1）F=（a，e，（（b, c, d）,f））

2）L= （（（b, c）,d）, (a), ((a), ((b, c), d)), e,（））

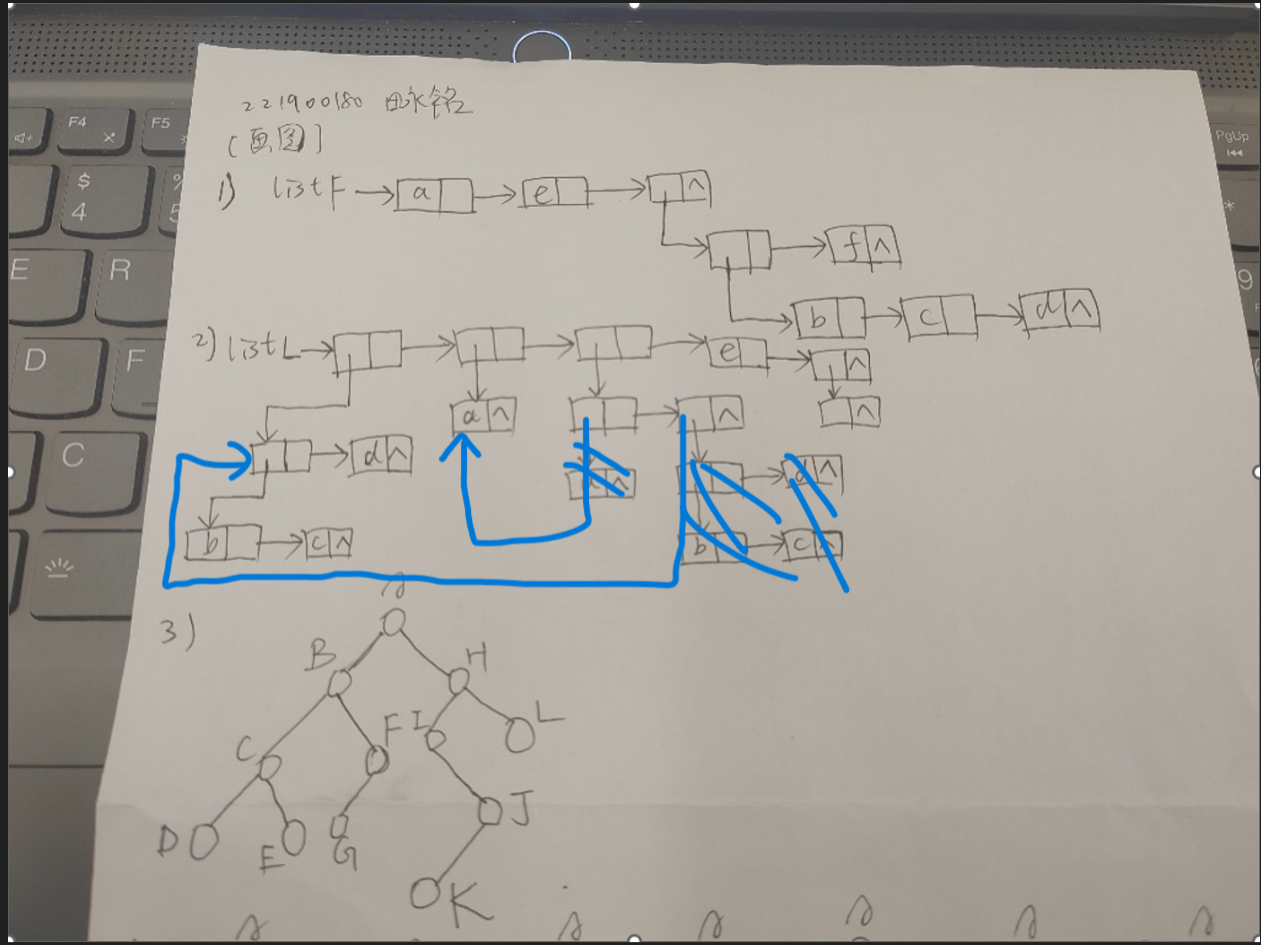
3) 已知一棵二叉树的先序、中序序列如下，画出该二叉树。

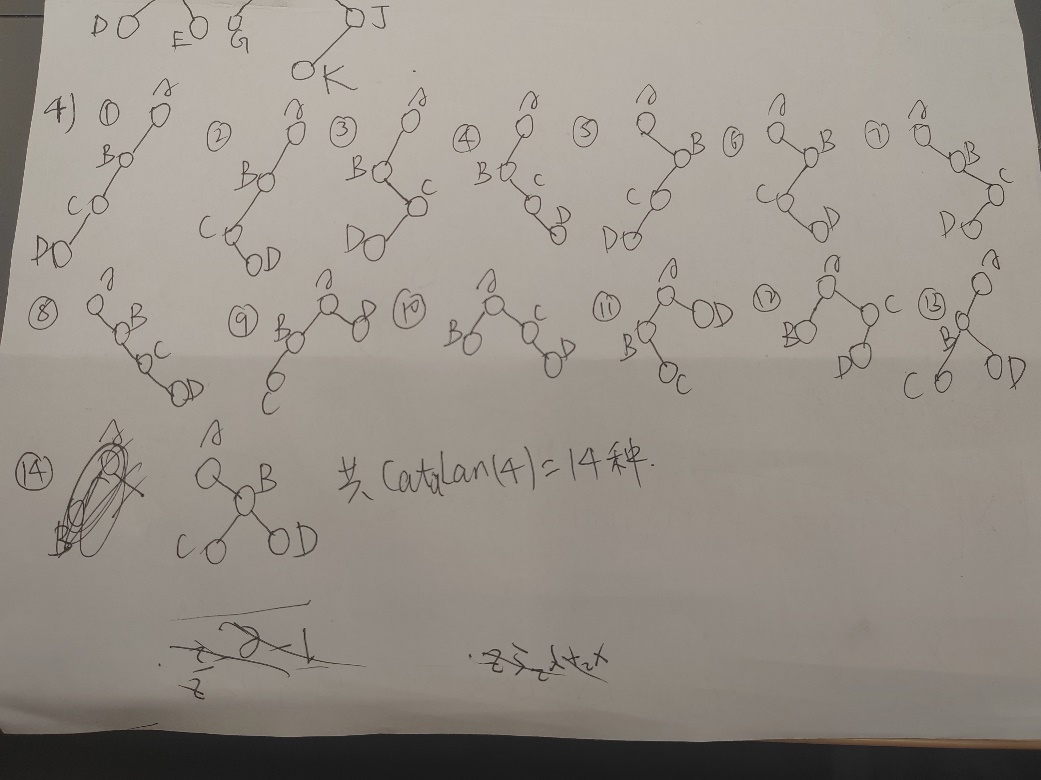
　　　　先序：ＡＢＣＤＥＦＧＨＩＪＫＬ

　　　　中序：ＤＣＥＢＧＦＡＩＫＪＨＬ

4）已知二叉树的前序遍历结果是ABCD，请画出所有可能的二叉树的形态。

答：





【写结果】

二叉树按完全二叉树的对应形式存储于数组A中，其中“@”表示空结点。写出该二叉树的中序遍历的结果。

0 1 2 3 4 5 6 7 8 9 10 11 12 13 14

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| A | B | C | D | E | F | G | @ | @ | H | @ | I | J | K | L |

答案：D B H E A I F J C K G L

【算法题】

写出二叉树左右子树的交换(每个分支结点下的子树都交换)的递归算法。

二叉树的结点定义如下：

struct BinTreeNode {

char data; // 树结点元素

BinTreeNode\* left; // 左子树指针

BinTreeNode\* right; // 右子树指针

}

答：

BinTreeNode\* BinTreeChange(BinTreeNode\* root){

if(root == nullptr) return root;

BinTreeNode \*tmp = root->left;

root->left = BinTreeChange(root->right);

root->right = BinTreeChange(tmp);

return root;

}

【算法填空】

以下非递归算法实现了二叉树左右子树的交换（栈实现），请补充完整：

栈的相关函数分别是：

template <class Type> class Stack {

void Push (Type x); //进栈

int Pop (Type& x); //出栈

int GetTop (Type& x); //取栈顶

bool IsEmpty( ); //判断栈是否为空

}

BinTreeNode\* BinTreeChangeStack(BinTreeNode\* root)

// 实现二叉树左右子树的交换（栈实现）

// 参数：二叉树根节点root

// 返回：二叉树

{

stack<BinTreeNode\*>s;

BinTreeNode \*temp=NULL , \*node=root;

int t=0;

if( 1) root==NULL )return NULL;

2) s.Push(root) ;

while(3) !s.IsEmpty() ){

temp=node->left;

4) node->left = node->right ;

node->right=temp;

if(node->right!=NULL){

5) s.Push(node->right) ;

}

if(6) node->left != NULL ){

7) node = node->left ;

}

else{

t= 8) s.Pop(node) ;

if (!t) break;

}

}

return node;

}