

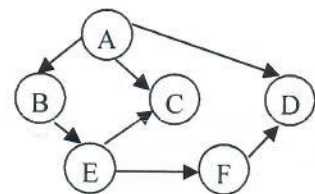
浙江工业大学
2019 年硕士研究生招生考试试题

考试科目： _____ (843) 数据结构 _____ 共 7 页

★★★★ 答题一律做在答题纸上，做在试卷上无效。★★★★

一、简答题 (30 分，每题 6 分)

- 1、什么是非线性结构？非线性结构只能依赖链式存储实现吗？请举例简述你的观点。
- 2、一棵拥有 1000 个结点的二叉树，最高几层？最矮几层？请简述你结论的计算依据。(假设根结点为 0 层)
- 3、栈 (Stack) 和队列 (Queue) 各有什么特点？
- 4、图的基本遍历策略有哪两种？
给出右图对应策略的遍历结果 (A 起点)。



5、已知有如下两种搜索策略:

① bool Search1(int a[], int n, int target, int& pos)

```
{ int i;
  for(i=0; i<n; i++)
    if(a[i]==target) { pos=i; return true; }
  pos=-1;
  return false;
}
```

② bool Search2(int a[], int n, int target, int &pos)

```
{ int f=0, e=n-1, m;
  while(f<e)
  { m=(f+e)/2;
    if(target>a[m]) f=m+1;
    else if(target<a[m]) e=m-1;
    else{ pos=m; return true; }
  }
  pos=-1;
  return false;
}
```

请使用大O标记法给出它们各自的时间复杂度, 哪种策略更优?
策略②是否适用于有序的单向链? 简述你的理由。

二、程序阅读题 (30 分, 每题 10 分)

1、阅读以下程序并回答随后的问题:

```
#include <iostream>
#include <vector>
using namespace std;
void f(const vector<int> & a, const vector<int> & b)
{
    vector<int> result;
    int i, j, t=0;
    for(i=0; i<a.size(); i++)
    { bool flag=true;
      for(j=0; j<b.size(); j++)
        if(a[i]==b[j]) { flag=false; break; }
      if(flag) result.push_back(a[i]);
    }
    for(i=0; i<result.size(); i++) cout<<result[i]<< ' ';
    cout<<endl;
}
int main()
{ int ax[]={19, 28, 37, 46, 55, 64, 73, 12, 9, 1}, bx[]={46, 3, 7, 19, 89, 72, 12};
  vector<int> a(ax, ax+10), b(bx, bx+7);
  f(a, b);
  return 0;
}
```

- 请回答 ①函数 f 完成什么功能?
②分析函数 f 的算法复杂度是什么?
③给出程序的输出结果。

2、阅读以下程序并回答随后的问题:

```
#include <iostream>
using namespace std;
class TNode
{
public:
    int w;
    TNode *lchild;
    TNode *rchild;
    TNode(int x, TNode *lptr = NULL, TNode *rptr = NULL)
    { w=x;
      lchild=lptr;
      rchild=rptr;
    }
};
```

```

void f (TNode *head)
{
    TNode *p;
    p=head;
    if(p!=NULL)
    {
        f (p->rchild);
        f (p->lchild);
        cout<<p->w<<' -' ;
    }
}

TNode *buildTree()
{
    TNode *root;
    TNode *p1,*p2,*p3,*p4,*p5,*p6,*p7,*p8;
    p1 = new TNode(10, NULL, NULL);
    p2 = new TNode(4, NULL, NULL);
    p3 = new TNode(7, p1, p2);
    p4 = new TNode(6, NULL, NULL);
    p5 = new TNode (12, p4, p3);
    p6 = new TNode(8, NULL, NULL);
    p7 = new TNode(11, NULL, NULL);
    p8 = new TNode(5, p6, p7);
    root = new TNode (3, p5, p8);
    return root;
}

int main()
{
    TNode *root;
    root = buildTree();
    f (root);
    return 0;
}

```

请回答①请描述该函数 f 的功能;

②画出 buildTree() 建立的二叉树;

③给出程序的输出结果。

3、请阅读以下两段代码：

代码一：

```
void f1(unsigned num)
{   int a[100],i=0;
    char ch;
    while(num) {
        a[i++]=num%16;
        num/=16;
    }
    for(i--;i>=0;i--) {
        if(a[i]>9) ch='A'+a[i]-10;
        else ch=a[i]+'0';
        cout<<ch;
    }
    cout<<endl;
}
```

代码二：

```
void f2(unsigned num)
{   if(num) {
        int k=num%8;
        f2(num/8);
        cout<<num%8;
    }
}
```

请回答：①代码一和代码二分别完成什么功能？

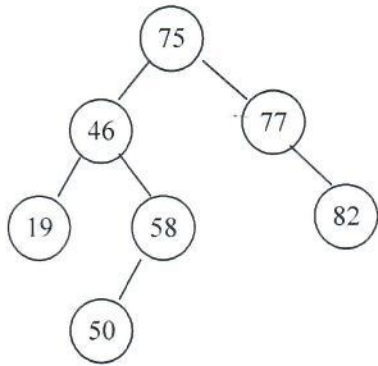
②若有调用语句 $f1(100)$ ；和 $f2(200)$ ；则分别输出什么？

三、综合题（55分）

1、（10分）有数据集合 $a=\{56, 23, 88, 45, 99, 7, 74, 36, 43\}$ ，使用散列函数 $F(X)=X\%11$ ；数据空间为 11 个整型单元；请①计算集合内各数据的散列值；②使用线性开地址散列的存储模型；③计算元素 99 在存储模型中的查找长度是多少？

2、（10分）有字符集合 $a=\{ 'A': 12, 'Q': 7, 'K': 21, 'B': 20, 'T': 12, 'J': 18, 'U': 10 \}$ ，冒号后表示各字符的使用频率。①给出由该字符集合构造 huffman 树的过程；②给出各字符对应的 huffman 编码。③计算该 huffman 树的期望长度。

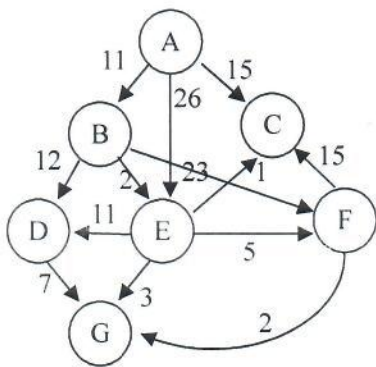
3、(10分) 已知有二查找树如下：



- ① 在该二叉查找树上添加结点 53。请画出新的查找树。
- ② 请计算新的查找树所有非叶节点的平衡因子，并在二叉树上注明。
- ③ 对新的查找树进行 AVL 平衡调整。要求画出平衡调整的每一步过程。

4、(10分) 有中缀表达式 $a*(b-c)/(d+e*f)*h$ ，请画出其在转换为后缀表达式过程中每一步符号栈和后缀表达式的变化过程，并给出转换后的后缀表达式。

5、(15分) 有如下带权有向图，请给出①该图的邻接表表示；②给出用 Dijkstra 算法计算从 A 点出发到其他各点的最短路径的过程。



四、程序设计题 (3 题, 总计 35 分)

1、(10 分) 已知链表结点定义如下:

```
class node
{
public:
    int nodeValue;
    node *next;
    node() : next(NULL) {}
    node(const int& item, node *nextNode = NULL): nodeValue(item), next(nextNode) {}
};
```

单向链表表头为 first, 待删除的值为整型, 假设有函数名叫 void DeleteKeys(node* first, int key), 可以让完成链表内重复值的删除。如原始集合 {45, 6, 45, 17, 23, 38, 45, 7}, 则调用 DeleteKeys(first, 45) 后集合变为 {6, 17, 23, 38, 7}。请编程实现这个函数。

2、(10 分) 冒泡排序和插入排序为常见的排序策略, 请编写程序实现下列要求:

- ①编写函数实现改进的冒泡排序策略: 在冒泡过程中引入对数据交换的检查来判断集合是否已经有序, 从而提高冒泡的效率。函数原型为: void BobSort(vector<int>& v); //v 为待排序集合, 要求按升序排列;
- ②编写函数实现插入排序策略。函数原型为: void InsertSort(vector<int>& v); //v 为待排序集合, 要求按升序排列。
- ③ 说明两种排序策略的稳定性。

3、(15 分) 二叉查找树 (Binary Search Tree) 是一种支持快速查找的结构, 现在请你给出二叉查找树的①二叉查找树结点 BinNode 的描述; ②若已知查找树的根结点指针为 root, 那么请实现在二叉查找树中插入值为 item 的算法。BinNode* BST_Insert(BinNode* root, int item) ③BST_Insert 的算法复杂度是多少?