江苏大学
硕士研究生入学考试样题

科目代码： 851 **A卷**

科目名称 数据结构 满分： 150分

注意：①认真阅读答题纸上的注意事项；②所有答案必须写在答题纸上，写在本试题纸或草稿纸上均无效；③本试题纸须随答题纸一起装入试题袋中交回！

一、单项选择题(每小题 1 分，共 10 分)

1. 以下说法正确的是(_____)。
   ① 数据的逻辑结构是指数据的各数据项之间存在的逻辑关系
   ② 顺序存储结构的优点是存储密度大
   ③ 数据元素是表示数据的最小单位
   (A) 只有①正确 (B) 只有②正确 (C) 只有③正确 (D) ①②③都正确
2. 如果用带头结点的头指针为 head 的单链表存储长度为 n 的线性表（$a_1,a_2,\ldots,a_n$），那么访问线性表中第 i 个元素的时间复杂度为(_____)。
   (A) O（1） (B) O（i） (C) O（i+1） (D) O（n）
3. 在单链表的指针为 p 的结点之后插入指针为 s 的结点,正确的操作是(_____)。
   (A) p->next=s;p->next=s->next; (B) s->next=p->next;p->next=s;
   (C) p->next=s->next;p->next=s; (D) p->next=s;s->next=p->next;
4. 若一个栈以下标范围是 1 到 n 的数组 elems 存储元素,初始空栈的栈顶指针 top 为 n+1,则下面 x 进栈的正确操作是(_____)。
   (A) top--; elems[top] =x; (B) elems[top]=x; top--;
   (C) top++; elems[top] =x; (D) elems[top]=x; top++;
5. 设有两个串 s 和 t，求 t 在 s 中首次出现的位置的运算称作(_____)。
   (A) 求子串 (B) 判断是否相等 (C) 模式匹配 (D) 连接
6. 程序设计语言对数组的操作主要有(_____)。
   (A) 建立和删除 (B) 索引和修改 (C) 存储和提取 (D) 查找和索引
7. 一棵完全二叉树上有 234 个结点，其中叶子结点的个数是(_____)。
   (A) 117 (B) 233 (C) 235 (D) 115
8. 在用邻接表表示一个有 n 个顶点和 e 条弧的有向无环图时，拓扑排序算法时间复杂度为(_____)。
   (A) O(n) (B) O(n*n) (C) O(n*e) (D) O(n+e)
9. 既希望较快的查找又便于线性表动态变化的查找方法是(_____)。
   (A) 顺序查找 (B) 折半查找 (C) 哈希法查找 (D) 索引顺序查找
10. 下列排序算法中，其中(_____)是稳定的。
    (A) 直接插入排序和快速排序 (B) 折半插入排序和冒泡排序
    (C) 简单选择排序和二路归并排序 (D) 锦标赛排序和希尔排序

二、填空题(每小题 2 分，共 10 分)

1. 设 n 为正整数，则下列程序段中加下划线的语句 x=x+y 的执行次数是_____。

```
int x=0,y=1;
for (int i=1; i<=n; i++)
    for (int j=1; j<=i; j++)
        for (int k=1; k<=j; k++)
            x=x+y;
```

2. 设双向循环链表的结点包括 data（数据域）、prior（指向前驱结点的指针域）、next（指向后继结点的指针域），head 是带头结点的双向循环链表的头指针，则双向循环链表为空的条件是_____。
3. 设 Head 表示取表头操作，Tail 表示取表尾操作，如果广义表 L=(((a,b),(c),(d,e)))，则操作 Head(Tail(Tail(Head(L))))的结果是_____。
4. 平衡因子的定义是_____。
5. 在快速排序、直接插入排序、二路归并排序、简单选择排序、堆排序这些排序算法中，在初始序列已基本有序的情况下，排序效率最高的算法是_____。

三、应用题(共 80 分)

1. (6 分)已知中缀表达式为#A*(B+C)/D#，其中#表示表达式开始符和结束符。现利用栈将该中缀表达式转换成后缀表达式。初始时将开始符#预先放到栈中，然后对 A*(B+C)/D#进行处理，按表 1 所示的格式写出各步骤的结果（在答题纸上画出该表填写）。

表 1　中缀表达式转换成后缀表达式

步骤	符号序列	栈	输出结果
0	无	#	无
1			
2			
…			

2. (10 分)已知树的形态如图 1 所示，要求:
   (1) 画出该树的双亲存储表示。
   (2) 画出该树所对应的二叉树。

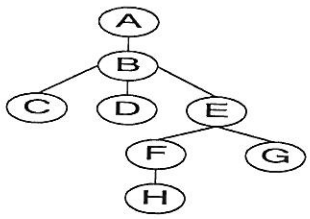

图 1 树

3. (10 分)一棵二叉树的先序、中序、后序序列如下，序列中部分字母未显示出来：

先序序列：_ _ C D E _ G H I _ K
中序序列：C B _ _ F A _ J I H K
后序序列：_ E F D B _ I K H _ A

要求：

(1) 写出完整的先序、中序、后序序列，并标出各序列中原未显示出来的字母。

(2) 画出该二叉树。

(3) 画出该二叉树的中序全线索二叉树。

4. （5 分）已知带权无向图如图 2 所示，请用普里姆(prim)法求该带权无向图的任意一棵最小生成树，设初始时 U={u_0}中的 u_0 为 V1，请给出构造最小生成树的过程。

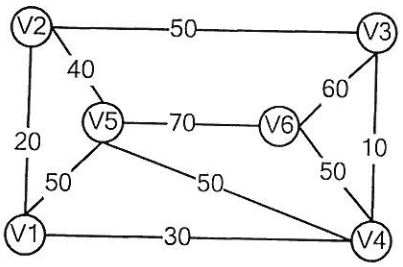

图 2 带权无向图

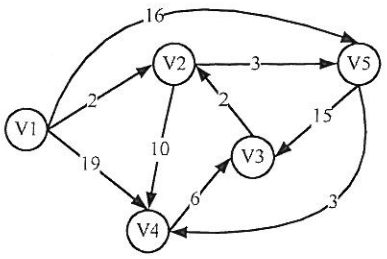

图 3 带权有向图

5. （15 分）已知带权有向图如图 3 所示，要求：

（1） 画出该带权有向图的邻接表。

（2） 给出基于你所画出的邻接表的从顶点 V1 出发的广度优先搜索 BFS 序列以及 BFS 生成树。

（3） 用 dijkstra 算法，求从源点 V1 出发到其它各终点的最短路径以及长度，请给出执行算法过程中各步的状态。

6. (7 分)按折半查找法对有序表（11，22，33，44，55，66，77，88，95，99）进行查找，要求：

(1) 画出描述折半查找过程的二叉查找树（也称判定树）的形态。

(2) 计算出在等概率情况下对该有序表进行折半查找时查找成功的平均查找长度 ASL。

(3) 如果对有 n 个记录的有序表进行折半查找，最大比较次数是多少？

7. （11 分）已知关键字序列 F={12，17，3，10，8，31，11，33}，利用除留余数法构造哈希函数，哈希函数为 H(K)= K % 7，其中 K 为关键字，哈希地址空间长度为 10，要求：

（1）若以线性探测(再散列)方法解决冲突，请画出相应的哈希表。

（2）指出该哈希表的装填因子α是多少？

（3）对用线性探测(再散列)方法构造的哈希表如何处理被删除的结点？请说明理由。

8. （8 分）已知关键字序列 F= {17，7，21，35，13，33，9，15，25，11，23}，要求最后排序结果是按关键字从小到大的次序排列，请给出增量序列 D={5，3，2，1}的希尔排序前 2 趟排序结果。并指出第 1 趟排序过程关键字总的比较次数是多少？

9. （8 分）已知关键字序列 F={21，27，53，44，8，19，39，14，34}，要求最后排序结果是按关键字从小到大的次序排列，请写出快速排序的每一趟的排序结果。

四、简答题(共 30 分)

1. （6 分）请解释什么是算法？简要说出算法有哪些重要特性？

2. （6 分）下面是一算法的核心部分，试说明该算法的功能。

```
    p=head->next;
    //head 是带头结点的非空单链表的头指针，结点有数据域 data 和指针域 next
    while (p->next!=NULL)
    {   q= p->next;
        if (q->data>=p->data) p=q;   else   return(false);
    }
   return(true);
```

3. （6 分）如果用单循环链表来实现队列，那么对于队列指针我们可以有两种方案。方案一：只设一个头指针，不设尾指针；方案二：只设一个尾指针，不设头指针。你认为对于单循环链表实现的队列，采用哪种方案更合适，请给出理由。
4. （6 分）请给出堆排序算法的基本思想。并从时间和空间两方面对简单选择排序、树形选择排序（也称锦标赛排序）和堆排序作一比较。
5. （6 分）若森林共有 m 个结点和 t 条边（t<m），则该森林中共有多少棵树，要求给出分析过程。

五、算法设计题(共 20 分)

注：(1)可采用类 C 语言或 C 语言或 C++语言描述你的算法，关键之处请给出简要注释。
(2)算法中可使用试卷最后给出的存储结构。
(3)若算法中使用了其他的存储结构和运算，请给出其定义和实现。

1. (8 分)设单链表的头指针为 head，数据域为 data，指向下一结点的指针域为 next。现有一个带头结点的单链表 L 中包含有 3 类字符（英文字母字符、数字字符和其他字符），每个结点数据域 data 存放一个字符，试编写一个算法将单链表 L 拆分为三个带头结点的单链表，使每个单链表中只包含同一类字符，原单链表 L 变为空表。要求用原表的结点空间，头结点可以另辟空间。同时要求用最少的时间。
2. (7 分)设二叉排序树用二叉链表存储，数据域为 data，其类型为 int，leftChild 是指向左孩子结点的指针，rightChild 是指向右孩子结点的指针。请写出一个非递归的算法（不使用栈结构），求出以 t 为根的二叉排序树中数据域的值为整数值 key 的结点的双亲结点的地址（假设二叉排序树 t 中结点的数据域的值各不相同）。
3. (5 分)假设带权有向图 G 以邻接表存储，请编写删除从顶点为 v1 到 v2 的弧的函数（v1,v2 是顶点表中相应顶点的下标，类型是 int）。若删除成功，则返回 1，否则返回 0（假设给定的 v1、v2 均是合法的，且 v1 不等于 v2）。

//单链表的存储结构描述如下：

```
// 结点类
template <class ElemType>
struct Node {
    ElemType   data;                    // 数据域
    Node<ElemType>   *next;             // 指针域
    Node(){next = NULL;}        // 无参数的构造函数，可用于构造头结点
    };
```

```
// 单链表类
template <class ElemType> class LinkList
{   protected:
      Node<ElemType> *head;     // 头结点指针
   public:
      LinkList(){head = new Node<ElemType>;} //构造函数，构造一个只有头结点的空链表
      virtual ~LinkList(){Clear(); delete head;}      // 析构函数
      void Clear();//删除单链表中所有元素结点，使单链表成为只有头结点的空表
};
//二叉排序树的存储结构描述如下：
// 二叉树结点类
template <class ElemType>
struct BinTreeNode
{     ElemType data;                                  // 数据域
      BinTreeNode<ElemType>   *leftChild;  // 左孩子指针域
      BinTreeNode<ElemType>   *rightChild; // 右孩子指针域
      BinTreeNode(){leftChild = rightChild = NULL;} // 无参数的构造函数
};
// 二叉排序树类
template <class ElemType>
class BinarySortTree
{
protected:
      BinTreeNode<ElemType> *root;
      void Destroy(BinTreeNode<ElemType> * &r);   // 销毁以 r 为根的二叉排序树
public:
      BinarySortTree(){ root = NULL;}// 无参数的构造函数，构造一棵空二叉排序树
      virtual ~BinarySortTree(){Destroy(root);}// 析构函数，销毁二叉排序树
};

//带权有向图的邻接表存储结构描述如下：
#define DEFAULT_SIZE 1000
// 带权有向图弧结点类
template <class WeightType>
struct AdjListNetworkArc
{
      int adjVex;                                     // 弧头顶点（邻接点）序号
      WeightType weight;                              // 弧的权值
      AdjListNetworkArc<WeightType> *nextarc;      // 指向下一个弧结点的指针
      AdjListNetworkArc(int v, WeightType w, AdjListNetworkArc<WeightType> * next =
```

```
        NULL) {adjVex = v;   weight = w;     nextarc = next;   } //构造邻接点序号为 v，弧的
        权为 w，下一个弧结点的指针为 next 的邻接弧
};
//带权有向图顶点结点类
template <class ElemType, class WeightType>
struct AdjListNetWorkVex
{
    ElemType data;                                  // 数据元素值
    AdjListNetworkArc<WeightType> *firstarc;   // 指向邻接表第一个弧结点的指针
    AdjListNetWorkVex(){firstarc = NULL;}//无参数的构造函数，构造一个空顶点结点
};
//带权有向图邻接表类
template <class ElemType, class WeightType>
class AdjListDirNetwork
{
protected:
    int vexNum, arcNum, vexMaxNum;    // 当前顶点数目、弧数和允许的顶点最大数目
    AdjListNetWorkVex<ElemType, WeightType> *vexTable; // 顶点表
public:
    AdjListDirNetwork(int vertexMaxNum = DEFAULT_SIZE)
    {//构造允许的顶点最大数目为 vertexMaxNum，弧数为 0 的空有向图
        if (vertexMaxNum < 0)
            throw Error("允许的顶点最大数目不能为负!");// 抛出异常
        vexNum = 0;
        vexMaxNum = vertexMaxNum;
        arcNum = 0;
        vexTable = new AdjListNetWorkVex<ElemType, WeightType>[vexMaxNum];
    }
    ~AdjListDirNetwork(){Clear(); delete []vexTable;}    // 析构函数
    void Clear();  //释放所有的弧结点，并把有向图的顶点数和弧数设置为 0
}
```