广东工业大学

2020 年硕士学位研究生招生考试试题

考试科目（代码）名称：(829)数据结构　　满分 150 分

（考生注意：答卷封面需填写自己的准考证编号，答完后连同本试题一并交回！）

一. 解答题(共 70 分，7 小题，每题 10 分)

1.（10 分）假设栈 S 为空栈，且进栈顺序为 1、2、3、4，试写出所有可能以 2 打头的出栈序列。

2.（10 分）已知哈希表的地址区间长度为 13，散列函数为：H(key) = key % 13，采用双散列探测法解决第 i 次冲突：$H_i(key) = (H(key) + i*H'(key)) \% 13$，$i \geq 1$，其中 H'(key) = (key % 11) + 1。试画出依次插入关键字（92, 81, 58, 21, 57, 45, 161）后的哈希表。

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

3.（10 分）已知一棵二叉查找树 T 如图所示，依次执行下列操作，试画出每一步操作的结果：

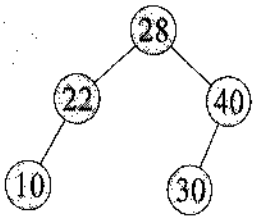
（1）（2 分）删除 30；

（2）（2 分）插入 15；

（3）（2 分）删除 22；

（4）（2 分）插入 45；

（5）（2 分）删除 28。



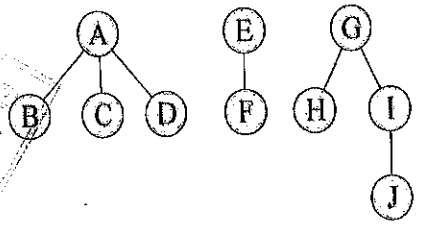
4.（10 分）已知关键字序列（12, 2, 16, 30, 28, 10, 18, 35, 8, 54），若对其执行升序的希尔排序，且增量序列为 d = {5, 3, 1}，试写出每一趟排序的结果。

第一趟：

第二趟：

第三趟：

5.(10 分)请画出与图中所示的森林 F 等价的二叉树 T。



6.（10 分）已知无向带权图 G 的顶点集 V = {a, b, c, d, e}，由于其邻接矩阵为对称矩阵，因此，可以只将邻接矩阵的上三角元素（包含主对角线元素）按行序为主序的顺序依次保存在如下的连续存储区域中，试回答下列问题：

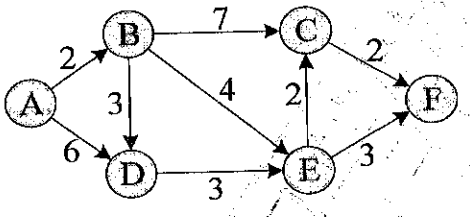
∞	5	∞	∞	3	∞	∞	∞	∞	∞	4	3	∞	2	∞

（1）（3 分）写出图 G 的邻接矩阵；

（2）（3 分）画出无向带权图 G；

（3）（4 分）基于该邻接矩阵，写出从顶点 a 出发的深度优先遍历序列。

7.（10 分）已知有向图 G 如图所示，试按照迪杰斯特拉算法，依次写出从源点 A 出发到其它顶点的最短路径和最短路径长度（顺序不能颠倒）。



二. 算法题（共 80 分，8 小题，每题 10 分）

1.（10 分）已知顺序表 L，阅读算法 f1，回答下列问题：

（1）（5 分）若 L=（5, 1, 9, 4, 3, 15），请写出执行算法 f1(L, 3, e)后的 L 和 e；

（2）（5 分）简述算法 f1 的功能。

```
Status f1(SqList &L, int i, ElemType &e) {
   int j;
   if(i<=0 || i>L.length) return ERROR;
   e = L.elem[i-1];
   for(j = i; j<L.length; j++) L.elem[j-1] = L.elem[j];
   L.length--;
   return OK;
}
```

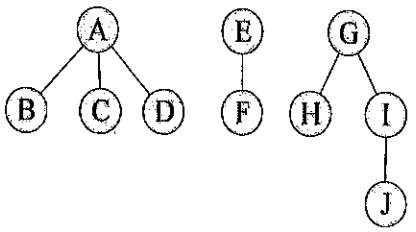
2.（10 分）已知森林 F 采用孩子兄弟存储结构，其类型定义如下。阅读算法 f2，回答下列问题：

```
typedef struct CSTNode {
  TElemType  data;
  struct CSTNode *firstChild, *nextSibling;
} CSTNode, *CSTree;
```

（1）（5 分）若 F 如图所示，请写出执行算法 f2(F)后的返回值；

（2）（5 分）简述算法 f2 的功能。

```
int f2(CSTree F) {
  int d1, d2, d;
  if(NULL==F) return 0;
  d1 = f2(F->firstChild);
  d2 = f2(F->nextSibling);
  d = d1+1 > d2 ? d1+1 : d2;
  return d;
}
```



3.（10 分）阅读算法 f3，回答下列问题：

（1）（5 分）若顺序表 L =（[illegible], -1, 8, 6, -4, 7, -2, -10），其中 0 号位置为空，请写出执行算法 f3(L)后的 L；

（2）（5 分）简述算法 f3 的功能。

```
void f3(SqList &L) {
  int i, j;
  i = 1; j = L.length;
  L.elem[0] = L.elem[1];
  while(i<j) {
    while(i<j && L.elem[j]<=L.elem[0]) j--;
    L.elem[i] = L.elem[j];
    while(i<j && L.elem[i]>=L.elem[0]) i++;
    L.elem[j] = L.elem[i];
  }
  L.elem[i] = L.elem[0];
}
```

4．（10 分）已知有向图 G 采用邻接数组存储结构，其类型定义如下。阅读算法 f4，回答下列问题：

```
#define MaxNum  5
typedef struct {
    char vexs[MaxNum];
    int  arcs[MaxNum][MaxNum];
    int  n, e;
} MGraph;
```

（1）（5分）若G的邻接矩阵如图所示，请写出执行算法f4(G, 2)的返回值；

（2）（5分）简述算法f4的功能。

```
int f4(MGraph G, int i) {
   for(int j = 0; j<G.n; j++) {
     if(G.arcs[i][j]!=0) return j;
   }
   return 0;
}
```

$$G.arcs = \begin{pmatrix} 0 & 0 & 1 & 1 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 1 \\ 1 & 1 & 0 & 0 & 0 \\ 1 & 1 & 1 & 1 & 0 \end{pmatrix}$$

5．（10分）设带头结点双向链表的类型定义如下：

```
typedef struct DuLNode {
   ElemType  data;
   struct DuLNode  *prior, *next;
} DuLNode, *DuLinkList;
```

算法f5删除双向链表L中p指针所指的结点，若删除成功，返回OK，否则，返回ERROR。请在空缺处填入合适内容，使其成为完整的算法。

```
Status f5(DuLinkList &L, DuLinkList p) {
   DuLinkList q;
   if(NULL==p || _____①_____) return ERROR;
   q = p->prior;
   _____②_____ = _____③_____;
   if(_____④_____) _____⑤_____ = q;
   free(p);
   return OK;
}
```

6.（10分）设记录类型定义如下：

```
typedef struct {
   KeyType key;
   …
} RcdType;
```

算法f6在有序序列rcd[low..high]折半查找目标关键字key，若存在，返回其位序，否则返回-1。请在空缺处填入合适内容，使其成为完整的算法。

```
int f6(RcdType rcd[], KeyType key, int low, int high){
   int mid = ____①____;
   if(____②____) return -1;
   if(____③____) return mid;
   else if(rcd[mid].key>key) return ____④____;
   else return ____⑤____;
}
```

7.（10分）设二叉树T采用二叉链表存储结构，其类型定义如下：

```
typedef struct BiTNode{
   int  data;
   struct BiTNode *lchild, *rchild;
}BiTNode, *BiTree;
```

算法f7在二叉树T中查找各结点中的最大元素值，并返回其结点所在位置。请在空缺处填入合适内容，使其成为完整的算法。

```
BiTree f7(BiTree T) {
   BiTree lp, rp, p;
   if(____①____) return NULL;
   lp = f7(T->lchild);
   rp = ____②____;
   ____③____;
   if(____④____) p = lp;
   if(rp!=NULL && rp->data>p->data) p = rp;
   ____⑤____;
}
```

8．（10分）设自然数集$S = \{a_1, a_2, a_3, \ldots\ldots, a_n\}$，$S_1$，$S_2$，……，$S_r$为S的真子集，其中$S = S_1 \cup S_2 \cup \ldots\ldots \cup S_r$，并且 $S_i \cap S_j = \emptyset$，且$i \neq j$，$1 \leq i, j \leq r$，若每个子集S_i的元素已按从大到小顺序排列有序，并且$|S_i| \geq k$，$1 \leq i \leq r$，试设计一个最坏情况下时间复杂度为$O(n)$的算法，选出集合S中前k个最大元素。

（1）（6分）用自然语言陈述算法的设计思想及执行步骤；

（2）（4分）分析在最坏情况下算法的时间复杂度。