武汉科技大学

2007 年硕士研究生入学考试试题

考试科目代码及名称：452 软件基础Ⅱ（含数据结构和离散数学）

说 明：1. 适用招生专业：计算机体系结构、计算机软件与理论和计算机应用技术（仅供数学类考生选考）

2. 答题内容写在答题纸上，写在试卷或草稿纸上的一律无效。

3. 考试时间 3 小时，总分值 150 分。

参考答案

第一部分 数据结构

1、

```
typedef  struct  node
  {int  freq; //频度域，记整数出现的次数。
   int  data; //输入的整数。
   struct  node  *next;
  }node, *LinkedList;
```

void CreatList（ ）//建立有整数的单向链表，重复整数只在链表中保留一个，并按出现次数降序排列。

```
{  LinkedList la;
  la= (LinkedList) malloc (sizeof (node)); //申请头结点。
  la->next=null;          //链表初始化。
  While (1)  //建立 n 个结点的链表
   {  scanf ("%d", a);    //接收整数 a。
      if (a==0)    break;
      p=la->next; pre=la; //p 是工作指针,pre 是前驱指针。
      while (p!=null&& p->data! =a)   {   pre=p; p=p->next;  }
      if (p==null)
          {    p= (LinkedList) malloc (sizeof (node));
               p->data= a;  p->freq=1; p->next=null;
               pre->next=p;
          }
      else
          {   p->freq++;       //整数重复出现,频度增 1。
              pre->next=p->next; //先将 p 结点从链表上摘下，再按频度域值插入到合适位置
              pre=la; q=la->next;
              while(q&&q->freq>p->freq)  {   pre=q; q=q->next; )
              pre->next=p; p->next=q; //将 p 结点插入到合适位置
          }
```

}//结束 **while** 循环建表。

2、

```
(1)void compress (int A[n][n], int B[ ], int n)
{//将三对角矩阵 A[0..n-1, 0..n-1]三条对角线上的元素逐行存放于数组 B 中
 K=0;
 for (i=1; i<=n; i++)
    for (j=1; j<=n; j++)
      if (A[i][j]!=0) B[k++]= A[i][j];
}
```

(2) 已知 k 求 i、j 时，则下标的对应关系是：

i = (k+1)/3+1

i＝1 时，j = n-i+k

i＝n 时，j＝ (k+2) %3

i 为其它值时，j＝n－i+ (k+1) %3

```
void uncompress (int B[ ], int A[n][n], int n)
{//由数组 B[0..3n-3]确定三对角矩阵 A[0..n-1, 0..n-1]
  for (i=0; i<n; i++)
       for (j=0; j<n; j++)  A[i][j] =0;
  for(k=0; k<=3*n-3; k++)
    { i=(k+1)/3+1;
      if (i＝＝1)  j＝n-i+k ;
      else if (i＝＝n)  j＝ (k+2) %3 ;
      else  j= n－i+ (k+1) %3;
      A[i][j]=B[k];
    }
}
```

3、

```
(1)int KMP(string s, string t)
{  i=1;   j=1;
   while( i<=s.curlen && j<=t.curlen)
       if( j==0 || s.ch[i]==t.ch[j]) {  i=i+1;  j=j+1;  }
       else  j=next[j];
   if(j>t.curlen) return i-t.curlen;
   else  return 0;
}
```

(2) KMP 算法的时间复杂性是 O（m+n)。

(3) p 的 next 值分别为 01112212321

4、

S1 和 S2 共享内存中一片连续空间，可以将 S1 和 S2 的栈底设在两端，两栈顶向共享空间的中心延伸，仅当两栈顶指针相邻（两栈顶指针值之差的绝对值等于 1）时，判断为栈满，当一个栈顶指针为 0，另一个栈顶指针 maxsize-1 时为两栈均空。

(1)入栈操作：

```
int push(int i,int x)     //入栈操作。i 为栈号，i=0 表示左边的栈 s1，i=1 表示右边的栈 s2，x 是入栈元素。入栈成功返回 1，否则返回 0。
{if(i<0||i>1){printf("栈号输入不对");exit(0);}
if(s.top[1]-s.top[0]==1) {printf("栈已满\n");return(0);}
switch(i)
 {case 0: s.stack[++s.top[0]]=x; return(1); break;
  case 1: s.stack[--s.top[1]]=x; return(1);
 }
}//push
```

（2） 退栈操作

```
elemtp pop(int i) //退栈算法。i 代表栈号，i=0 时为 s1 栈，i=1 时为 s2 栈。退栈成功返回退栈元素，否则返回-1。
{if(i<0 || i>1){printf("栈号输入错误\n"); exit(0);}
  switch(i)
   {case 0: if(s.top[0]==-1) {printf("栈空\n"); return (-1); }
            else return(s.stack[s.top[0]--]);
    case 1: if(s.top[1]==maxsize {printf("栈空\n"); return(-1);}
            else return(s.stack[s.top[1]++]);
   }
}//算法结束
```

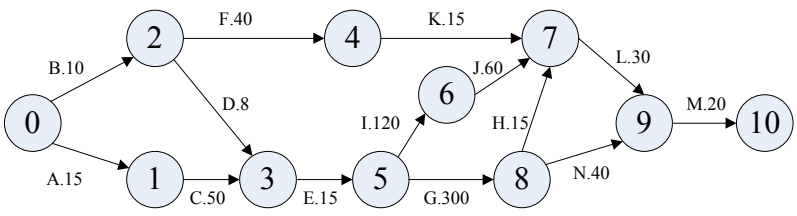
5、

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

void inorder(BiTree T)
{   BiTree s[MAXSIZE],p=T;
   Int top=0;
   Do {   While(p)   {    top++; s[top]=p;   p=p->lchild;   }
          If(top>0)   {   P=s[top];    top--;    Printf(p->data); P=p->rchild; }
          }while(p || top!=0);
}
```

6、

（1）AOE 网如右图



图中虚线表示在时间上前后工序之间仅是接续顺序关系不存在依赖关系。顶点代表事件，弧代表活动，弧上的权代表活动持续时间。题中顶点 0 代表工程开始事件，顶点 10 代表工程结束事件。

（2）各事件发生的最早和最晚时间如下表

事　　件	0	1	2	3	4	5	6	7	8	9	10
最早发生时间	0	15	10	65	50	80	200	395	380	425	445
最晚发生时间	0	15	57	65	380	80	335	395	380	425	445

（3）关键路径为顶点序列：0->1->3->5->8->7->9->10；

事件序列：A->C->E->G->H->L->M，完成工程所需的最短时间为 445。

7、

（1） 二叉排序树的建立与关键字的输入序列有关系。同样的关键字如果按照不同的输入序列会得到不同的二叉排序树。

（2） 采用二叉树的二叉链表作为数据结构

```
void bi_sort_tree (BiTree T)
 {
    if  (T)
      {
          bi_sort_tree (T->right) ;
          printf(T->data);
          bi_sort_tree (T->left) ;
       }
}
```

（3） 共分成四种情况：

第一种情况：p 为叶子节点

操作：f->left=NULL;　free (p);

第二种情况：p 只有左孩子节点

操作：f->left=p->left;　free (p);

第三种情况：p 只有右孩子节点

操作：f->left=p->right;　free (p);

第四种情况：p 左右孩子节点均存在

```
操作：q=p;  s=p->left;
      while( s->right )  {  q=s;   s=s->right;  }
      q->right=s->left;
      s->left=p->left;
      s->right=p->right;
      free(p);
```

8、

(1) 在最好情况下，假设每次划分能得到两个长度相等的子文件，文件的长度 $n=2^k-1$，那么第一遍划分得到两个长度均为⌊n/2⌋的子文件，第二遍划分得到 4 个长度均为⌊n/4⌋的子文件，以此类推，总共进行 $k=\log_2(n+1)$遍划分，各子文件的长度均为 1，排序完毕。当 n=7 时，k=3，在最好情况下，第一遍需比较 6 次，第二遍分别对两个子文件（长度均为 3，k=2）进行排序，各需 2 次，共 10 次即可。

(2) 在最好情况下快速排序的原始序列实例：4, 1, 3, 2, 6, 5, 7。

(3) 在最坏情况下，若每次用来划分的记录的关键字具有最大值(或最小值)，那么只能得到左(或右)子文件，其长度比原长度少 1。因此，若原文件中的记录按关键字递减次序排列，而要求排序后按递增次序排列时，快速排序的效率与冒泡排序相同，其时间复杂度为 $O(n^2)$。所以当 n=7 时，最坏情况下的比较次数为 21 次。

(4) 在最坏情况下快速排序的初始序列实例： 7, 6, 5, 4, 3, 2, 1，要求按递增排序。

第二部分 离散数学

1. 试证明：a) $f(A\cup B)=f(A)\cup f(B)$ b) $f(A\cap B)\subseteq f(A)\cap f(B)$.

证明：a) 设 $y\in f(A\cup B)$，则存在 $x\in A\cup B$，使 $f(x)=y$，即 $x\in A\vee x\in B$ 时，有 $y=f(x)$。故 $f(x)\in f(A)\vee f(x)\in f(B)$，因此，$y\in f(A)\cup f(B)$，于是 $f(A\cup B)\subseteq f(A)\cup f(B)$。

反之，设 $y\in f(A)\cup f(B)$，则有 $y\in f(A)\vee y\in f(B)$，由此 $\exists x\in A\cup B$,使得 $y=f(x)$，所以 $y\in f(A\cup B)$，故 $f(A\cup B)\supseteq f(A)\cup f(B)$。

综上所述，$f(A\cup B)=f(A)\cup f(B)$。

b) 设 $y\in f(A\cap B)$，则存在 $x\in A\cap B$，使 $f(x)=y$。即存在 $x\in A\wedge x\in B$，使 $f(x)=y$。故

$$y\in f(A)\wedge y\in f(B)\Rightarrow y\in (f(A\cap B))$$

所以：$f(A\cap B)\subseteq f(A)\cap f(B)$。

2．设 G 是无向连通图，证明：若 G 中有桥或割点，则 G 不是哈密顿图。

证明： (1)证明有割点的无向连通图 G 不是哈密顿图。

设 v 为 G 中的一个割点，则{v}为 G 中一个割集，则

p(G-{v})≥2>|{v}|=1

由定理可知，G 不是哈密顿图。

(2)再证有桥的图不是哈密顿图。

连通有桥图的最小阶数 n=2，即 K_2，而 K_2 不是哈密顿图。对于阶数 n≥3 的有桥图，任一桥的两个端点中，至少有一个是割点。由(1)知，这样的图也不是哈密顿图。

3．在一次象棋比赛中，n 名选手中的任意两名选手之间至多只下一盘，又每人至少下一盘，证明：总能找到两名选手，他们下棋的盘数相同。

证明：

做无向图 G=<V，E>，其中 V={v|v 为选手}

E={(u，v)|u，v∈V∧u 与 v 下过棋∧u≠v}

则 G 为无向简单图。

d(v)：v 下棋次数。由已知条件可知，1≤d(v)≤n-1。

于是 d(v1)，d(v2)，…，d(vn)只能取从 1 到 n-1 中取值。

由鸽巢原理，这 n 个顶点度数至少有 $\left\lceil \frac{n}{n-1} \right\rceil$=2 个相同，

即至少有 2 个人下棋次数相同。

4.答：(1) ③S∩R⊆T

(2) ④H=G∪T

(3) ⑤T∩G=∅

(4) ⑦G⊆F∪S

(5) ⑧S-(R∪M)⊆G

5. 在自然推理系统 **F** 中，证明推理：好人、坏人都是人，猴子不是人，因此猴子既不是好人、也不是坏人。

设：F(x):x 为好人，G(x):x 为坏人，R(x)为人，　H(x)为猴子

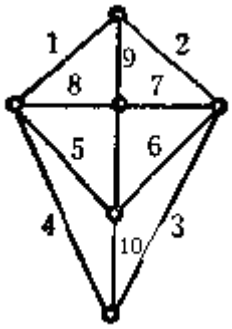
前提：　$\forall x((F(x)\vee G(x))\rightarrow R(x)), \forall x(H(x)\rightarrow \neg R(x))$

结论：　$\forall x(H(x)\rightarrow(\neg F(x)\wedge\neg G(x)))$

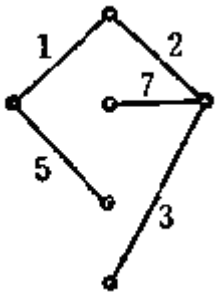
证明：

①	$\forall x((F(x)\vee G(x)\rightarrow R(x))$	前提引入
②	$F(y)\vee G(y))\rightarrow R(y)$	①UI
③	$\forall x(H(x)\rightarrow\neg R(x))$	前提引入
④	$H(y)\rightarrow\neg R(y)$	③UI
⑤	$\neg R(y)\rightarrow\neg (F(y)\vee G(y))$	②置换
⑥	$H(y)\rightarrow\neg (F(y)\vee G(y))$	④⑤假言三段论
⑦	$H(y)\rightarrow(\neg F(y)\wedge\neg G(y))$	⑥置换
⑧	$\forall x(H(x)\rightarrow(\neg F(x)\wedge\neg G(x)))$	⑦UG

6．利用 Kruskal 算法求下图一棵最小生成树。



解：最小生成树为：



7. 设群 G 为<P({a,b}), ⊕>， 其中 ⊕ 为集合的对称差运算，解下列方程：

{a} ⊕ x= ϕ 和 y ⊕ { a,b}={b}。

证明：由定理得知:

$x=\{a\}^{-1}\oplus \phi$

又已知群 G 的单位元为 ϕ,则$\{a\}^{-1}=\{a\}$

∴ $x=\{a\}\oplus \phi=\{a\}$

同理

$y=\{b\}\oplus\{a,b\}^{-1}=\{b\}\oplus\{a,b\}=\{a\}$