

# 华中科技大学

## 二〇〇四 年招收硕士研究生入学考试试题

考试科目: 数据结构及程序设计技术

适用专业: 生物信息技术

(除画图题外, 所有答案都必须写在答题纸上, 写在试题上及草稿纸上无效, 考完后试题随答题纸交回)

### 数据结构部分 (共 80 分)

#### 一 选择填空 (每空 1 分, 共 15 分)

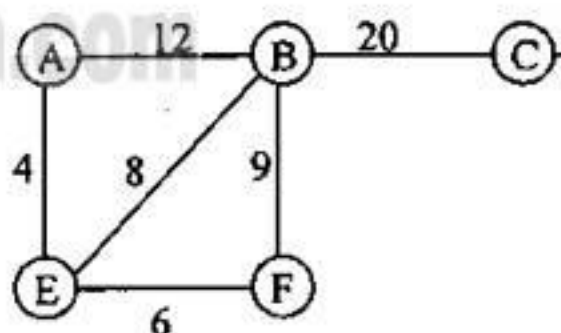
- 1 设有 100 个结点, 用二分方法查找时, 最大的比较次数是\_\_\_\_\_。  
(A) 25 (B) 50 (C) 10 (D) 7
- 2 若线性表最常用的操作是存取第  $i$  个元素及其前趋的值, 则采用\_\_\_\_\_存储方式节省时间。  
(A) 单链表 (B) 双向链表 (C) 单循环链表 (D) 顺序表
- 3 已知数据表 A 中每个元素距其最终位置不远, 则采用\_\_\_\_\_排序算法最省时间。  
(A) 堆排序 (B) 插入排序 (C) 直接选择排序 (D) 快速排序
- 4  $n$  个顶点的无向图的邻接表中结点总数最多有\_\_\_\_\_个。  
(A)  $2n$  (B)  $n$  (C)  $n/2$  (D)  $n(n-1)$
- 5 设连通图  $G$  的顶点数为  $n$ , 则  $G$  的生成树的边数为\_\_\_\_\_。  
(A)  $n-1$  (B)  $n$  (C)  $2n$  (D)  $2n-1$
- 6 在建立邻接表或逆邻接表时, 若输入的顶点信息即为顶点的编号, 则建立图的算法的时间复杂度为\_\_\_\_\_。  
(A)  $O(n+e)$  (B)  $O(n^2)$  (C)  $O(n \times e)$  (D)  $O(n^3)$
- 7 快速排序算法在最好情况下的时间复杂度为\_\_\_\_\_。  
(A)  $O(n)$  (B)  $O(n^2)$  (C)  $O(n \log_2 n)$  (D)  $O(\log_2 n)$
- 8 以知一棵二叉树的前序和中序序列分别为 ABDEGCFH 和 DBGEACHF 则该二叉树的后序序列为 \_\_\_\_\_。  
(A) GEDHFBCA (B) DGE BH FCA (C) ABCDEFGH (D) ACBFEDHG

- 9 线索二叉树中指向某结点的指针为 D, 该结点没有左孩子的条件是\_\_\_\_\_。  
 (A)  $D \rightarrow Lchild = NULL$  (B)  $D \rightarrow Rchild = NULL$   
 (C)  $D \rightarrow Ltag = 0$  (D)  $D \rightarrow Ltag = 1$
- 10 下列算法中, 某一轮结束后未必能选出一个元素放在其最终位置上的是\_\_\_\_\_。  
 (A) 堆排序 (B) 冒泡排序 (C) 直接插入排序 (D) 快速排序
- 11 将新元素插入到链式队列中时, 新元素只能插入到\_\_\_\_\_。  
 (A) 链头 (B) 链尾 (C) 链中 (D) 第  $i$  个位置 ( $i \geq 1$  且  $i \leq \text{表长} + 1$ )
- 12 假如只想得到 1000 个元素组成的序列中第 4 个最小元素之前的部分排序的序列, 用\_\_\_\_\_方法最好。  
 (A) 冒泡排序 (B) 快速排序 (C) 堆排序 (D) 选择排序
- 13 \_\_\_\_\_是不稳定的排序方法。  
 (A) 冒泡排序 (B) 归并排序 (C) 插入排序 (D) 选择排序
- 14 线性表是具有  $n$  ( $n \geq 0$ ) 个\_\_\_\_\_的有限序列。  
 (A) 字符 (B) 表元素 (C) 数据元素 (D) 数据项
- 15 从逻辑上, 可以将数据结构分为\_\_\_\_\_两类。  
 (A) 动态表和静态表 (B) 顺序结构和链式结构  
 (C) 线性结构和非线性结构 (D) 动态结构和静态结构

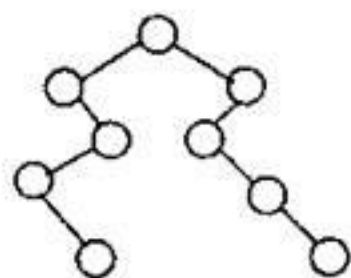
## 二 问题求解 (每小题 6 分, 共 18 分)

1 给定下列网络 G:

- 1) 求 G 的最小生成树, 画出逻辑图;
- 2) 画出 G 的邻接矩阵和邻接表。



2 已知下图为一棵二叉排序数, 各结点的值依次为 1—9, 请标出各结点的值。



3 找出所有二叉树, 其结点在下述两种次序之下, 恰好都以同样的顺序出现:  
 a) 前序和中序; b) 前序和后序

三 简述下述算法的功能（每小题 6 分，共 18 分）

```
(1) void a1 ( stack &s) {  
    int i, n, a[255];  
    n = 0;  
    while ( ! stackempty(s) )  
        { n++;  
          pop ( s, a[n]);  
        }  
    for ( i = 1; i <= n; i++) push(s, a[i]);  
}
```

```
(2) void a2 ( stack &s, int e ) {  
    stack t; int d;  
    initstack (t);  
    while (! stackempty (s))  
        { pop (s, d);  
          if ( d != e ) push ( t, d ); }  
    while (! stackempty (t))  
        { pop (t, d);  
          push ( s, d ); }  
}
```

```
(3) void change (linklist &head) //head 是无表头结点的单链表  
{ linklist q, p;  
  if (head && head->next)  
  { q=head;  
    head=head->next;  
    p=head;  
    while (p->next) p=p->next;  
    p->next=q;  
    q->next=NULL;  
  }  
}
```

四、算法设计（共 29 分）

1 若输入 12000 个不同的整数, 其值介于 0 和 19999 之间, 采用哈希方式存储这些数, 哈希函数为  $h(k)=k/2$ , 用链地址法克服冲突, 请你设计一个实现算法。提示: 利用两个数组来进行。一维数组  $r[6000]$  存放发生冲突时的关键字, 结构数组 HT 有二个成员, 一个存放哈希函数的关键字, 另一个指示发生冲突时存于  $r$  中的关键字的地址。（15 分）

- 2 试设计一个非递归算法把二叉树的叶结点按从左到右的顺序连成一个单链表，二叉树用二叉链表存储，链接时叶子结点的 rchild 域用做单链表的指针。（14 分）

算法所涉及的结构如下：

```
typedef struct bnode{
    elementype info;
    struct bnode *lchild, *rchild;
} bnode, *bitree; // 结点类型和指向二叉树的指针的类型
```

```
typedef struct seqstack {
    bitree data [maxsize]; // 栈的最大容量为 maxsize
    int top;
} seqstack; // 定义顺序栈类型
```

## C 语言程序设计部分 (70 分)

一、写出下列程序的输出结果：（每小题 3 分，共 45 分）

```
1. fun (int *p1, int *p2)
{ if (*p1 > *p2) printf ("%d\n", *p1);
  else printf ("%d\n", *p2);
}
main ( )
{ int a=3, b=7;
  fun (&a, &b);
}

2. #define FMT " %d, "
   main ( )
   { int b [ ][4]={1, 3, 5, 7, 9, 11, 13, 15, 17, 19, 21, 23};
     printf (FMT, *(b+1)+1 );
     printf (FMT, b[2][2]);
   }
```



```

3. fun(int *p)
{ int a=10;
  p=&a;
  ++a; }
main ( )
{ int a=5;
  fun (&a);
  printf(" %d\n" ,a);
}

```

```

4. #include <stdio.h>
main ( )
{ int a [2][3]={1, 2, 3, 4, 5, 6};
  int m,*p;
  p=&a[0][0];
  m=(*p)*(*(p+2))*(*(p+4) );
  printf (" %d\n" ,m);
}

```

```

5. #include<stdio.h>
void main (void)
{ double a, b, x;
  scanf ("%f %f" ,& a, & b);
  if (! a)
    printf (" error in input data \n" );
  else { x=-b/a;
        printf (" x=%.4f\n" ,x);
      }
}

```

输入: 100 50 输出: \_\_\_\_\_

输入: 0 20 输出: \_\_\_\_\_

```

6. #include <stdio.h>
void f (int k)
{ static double s=0.0;
  s-=1.0/k;
  printf (" k=%d, s=%6.2f\n" ,k, s);
}
void main (void)
{ int i;
  for (i=1; i<=3; ++i) f(i);
}

```

```

7. main( )
{ int a=2, x=5, b=7;
  fun (b, 6);
  printf(" a=%d, b=%d, x=%d\n" , a, b, x);
}
fun (int a int b)
{ int x=7;
  printf (" a=%d, b=%d, x=%d\n" , a, b, x);
}

8. main( )
{ int x=2, y=3, z=0;
  printf (" x=%d Y=%d z=%d\n" , x, y, z);
  f(x, y, z);
  printf (" @x=%d, Y=%d z=%d\n" , x, y, z);
}
f(int x, int y, int z)
{ z=x+y; x=x*x; y=y*y;
  printf(" x=%d Y=%d z=%d\n) " , x, y, z);
}

9. try( )
{ static int x=3;
  x + +; return(x);
}
main( )
{ int i, x;
  for (i=0; i<=2; i++) x=try( );
  printf(" %d\n" , x);
}

10. int abc(int u, int v);
main( )
{ int a=24, b=16, c;
  c=abc(a, b); printf(" %d\n" , c);
}
int abc(int u, int v)
{ int w;
  while (v) { w=u%v; u=v; v=w;}
  return u;
}

```

```

11. main( )
    { int x=10, y;
      y= func(x);
      printf ( " %d\n" , x+y);
    }
    func(int m)
    { int x=20;
      m=m+x;
      return m;
    }

```

```

12. main( )
    { incl( );  incl( );  incl( );
      inc2( );  inc2( );  inc2( );
    }
    incl( )
    { int x=0;
      x++;
      printf(" in incl x=%d\n" , x);
    }
    inc2( )
    { static int x=0;
      x++;
      printf(" in inc2 x=%d\n" , x);
    }

```

```

13. main ( )
    { int a[ ]={1, 2, 3, 4, 5, 6}
      int *p;
      p=a;
      printf(" %d, " , *p);
      printf(" %d, " , *(++p));
      printf(" %d, " , *++p);
      printf(" %d, " , *( p--));
      p+=3;
      printf ( " %d,%d\n" , *p, *(a+3));
    }

```

14. #include <stdio.h>

main ( )

{ char a[ ]= " language" ,b[ ]= " program" ;

char \*p=a, \*q=b;

int k;

for (k=0; k<7; k++)

if (\*(p+k) == \*(q+k)) printf ("%c" ,\*(p+k));

}

15. #include <stdio.h>

char b[ ]= " program" ;

char \*a=" PROGRAM" ;

main ( )

{ int i=0;

printf(" %c%s\n" ,\*a, b+1);

while (putchar(\*(a+i))) i++;

printf (" \ni=%d\n" ,i);

while (--i) putchar (\*(b+i));

printf(" \n%s\n" ,&b[3]);

}

## 二、用中文表示下列 C 说明 (每小题 2 分, 共 10 分)

例如 C 说明为 char \* PC; 则中文表示 PC 是字符型指针变量

1. short int a;

2. float b[3];

3. long (\* C)[3];

4. unsigned \* d[3];

5. void (\* f)(int);

## 三、程序设计 (15 分)

相传韩信点兵, 从不直接数自己军队的人数, 只是让士兵先后以 3 人一排、5 人一排、7 人一排的变换队形, 而他每次只掠一眼队伍的排尾就知道总人数了 (不超过 100 人)。实际上这是一个余数问题, 即已知一个自然数被 3、5、7 除后的余数, 求此自然数。编程输入此三余数, 输出满足条件的最小自然数。