2014 年硕士学位研究生入学考试试题参考答案

考试科目：872 数据结构与操作系统

数据结构部分

1、判断

（1）错误， 无论是有向图还是无向图，所有的顶点的度数之和等于所有边的总数的 2 倍。

（2）错误，当排序数据已基本有序时，快速排序效率很低，因为此时快速排序在分区时产生的两个区域分别包含 n-1 个元素和 0 个元素；每一出现这种不对称划分时花在划分的时间代价为 O(n)，总的时间 复杂度是 O（n^2）。

（3）正确，给定一个子串，要求在某个字符串（主串）中找出与该子串相同的所有子串，这就是模式匹配。若存在相同的子串，则返回子串在主串中的位置。

2、

（1）迪杰特斯拉算法的用途是：在图数据结构应用中，用于计算一个点到其他所有点的最短路径。

（2）基本思想是：采用贪心的思想，首先设置一个集合 S，存放已求出其最短路径的顶点，则尚未确定最短路径的顶点集合是（V-S），其中 V 为图中所有顶点集合；然后按最短路径长度递增的顺序逐个以（V-S）中的顶点加到 S 中．直到 S 中包含全部顶点，而 V-S 为空为止。

（3）验证算法的正确性： 生成一个均匀分布的网络，然后对算法进行测试，将每对节点对之间的最短路径绘制成图，最后应该得到的是一幅完整的栅格图状图片，如果没有出现孤立节点，则说明该算法正确！

（4）采用邻接矩阵

```
typedef struct _graph
{
    char vexs[MAX];           // 顶点集合
    int vexnum;               // 顶点数
    int edgnum;               // 边数
    int matrix[MAX][MAX]; // 邻接矩阵
}Graph, *PGraph;
```

// 边的结构体

```
typedef struct _EdgeData
{
    char start; // 边的起点
    char end;   // 边的终点
    int weight; // 边的权重
}EData;
```

也可以采用邻接表，具体参考书上。

3、

算法思想：由于要按递减的顺序生成单链表，这里采用头插法。对 n 进行分解质因数，先找到一个最小的质数 k=2，然后按下述步骤完成：（1）如果这个质数恰等于 n，则说明分解质因数的过程已经结束，将其插入到链表中；（2）但 n 能被 k 整除，则应把 k 的值插入到链表中，并用 n 除以 k 的商作为新的正整数 n，重复执行此步。(3)如果 n 不能被 k 整除，则用 k+1 作为 k 的值,重复执行第一步。

采用 java 实现算法如下：

```
//节点类
public class Node {
    private   Node next; //指针域
    private int data;//数据域
    public Node( int data) {
        this. data = data;
    }
}
//单链表类
public class LinkList {
    public Node first; // 定义一个头结点
    private int pos = 0;// 节点的位置
    public LinkList() {
      this. first = null;
    }
    // 插入一个头节点，从头部插入节点
    public void addFirstNode( int data) {
        Node node = new Node(data);
```

```
            node. next = first;
             first = node;
        }
        //分解并添加到单链表中，采用头插法
        public static void resolveAndGenerateList(int n)
        {
              for (int i=2;i <=n;i++) {
                 if (i == n) {
                    addFirstNode(i);
                    return;
                  }
                if (n > i && (n % i == 0)) {
                     addFirstNode(i);
                     resolveAndGenerateList(n/i);
                     break;
              }
        }
}
```

4、

算法思想：　采用递归思想，递归的判断二叉树的左子树和右子树是否为严格二叉树。

```
typedef struct BiTNode{
  ElemType  data;
  struct BiTNode *lchild,*rchild;
}BiTNode,*BiTree
 //判断树是否为严格二叉树，函数值是 1 为严格树，0 为非严格树
   int    isstrictBinarryTree(BiTNode root) {
           if(root == null) {     //如果根节点是空则返回 false
                return 0;
```

```
        } else {
            int left = isstrictBinarryTree (root->lchild); //判断左边是否有子节点
            int right = isstrictBinarryTree (root->rchild);//判断右边是否有子节点
            //如果左边和右边都有子节点就返回 1，或者左边和右边都没有子
              节点也返回 1
            if((left + right == 2) || (left + right == 0)) {
                return 1;
            } else {
                return 0;//其他情况返回 0
            }
        }
    }
```

5.

算法思路：n 个整型数存放在顺序线性中，即可以定义数组 A 存放 n 个数，不妨设 k = n/4（n>4 时，k>0 且 k 的值是整数），要找出第 k 小的数，首先可以对所有的数进行排序，再找即可。采用快速排序思想：先从 n 个元素中找一个数 m 作为分界点,假设 m 在列表中的位置为 i，当 i = k 时，m 即是找的第 k 小的数；当 i > k 时，从第 1~ (i-1)中查找；当 i < k 时，从(i+1)~n 中查找。具体算法如下：

```
int find_k_quickSort (int A[], int left, int right,int n) {
    if(left < right   &&   n>=4){
        int pivotloc;
        int key = A[left];
        int low = left;
        int high = right;
        int k = n/4;
        while(low < high){
                while(low < high && A[high] > key){
                        high--;
                }
                A[low] = A[high];
                while(low < high && A[low] < key){
                        low++;
                }
                A[high] = A[low];
        }
```

```
            A[low] = key;
            pivotloc = low;
            if(pivotloc == (k-1)){
                return A[pivotloc];
            }
            else if(pivotloc < (k-1)){
                return find_k_quickSort (A,pivotloc+1,right,k);
            }
            else{
              return find_k_quickSort (A,left,pivotloc-1,k);
            }
        }else{
            return -1;
        }
    }
```

【举一反三】找出顺序表中最小的 k 个数。

操作系统部分

6、判断题

（1）错误，系统资源分配的最小单位是进程，线程是 cpu 最小的调度单位。

（2）错误，有可能没有一个资源，但至少有两个进程占有资源。

（3）错误，虚拟存储器的容量由计算机的地址结构决定。

（4）正确。

（5）正确。

7、

文件的逻辑结构：是从用户观点来看所观察到的文件的组织形式，是用户可以直接处理的数据及其结构。文件的物理结构是从计算机的角度出发，文件在外存上的存放组织形式。从逻辑结构上看，一种是有结构的记录式文件，另一种是无结构的流式文件。记录式的逻辑结构通常有顺序、索引和索引顺序。

顺序文件的优点：顺序存取时速度较快

索引文件的优点：可以进行随机访问，易于进行文件的增删

顺序索引文件的优点：极大提高了顺序存取的速度

文件目录的功能：实现按名存取，提高检索速度，允许文件同名，允许文件共享等。

再举个具体的例子结合上述说明下。

8、

（1）进程和它创建的子进程（即父进程和子进程）：父进程创建子进程，系统为子进程分配一块独立的地址空间，将可执行文件或其他任何必要的动态链接库文件的代码和数据装载到该地址空间中；子进程被创建之后，父进程的全局变量、静态变量等会复制到子进程的地址空间中。父进程与子进程各自拥有独立的地址空间，两个进程结束时，将互不相干。

（2）进程和它创建的线程：属于同一进程的不同线程会共享进程内存空间中的全局区和堆，而私有的线程空间主要拥有少量的资源，则主要包括计数器、栈和寄存器。因此，对于同一进程的不同线程来说，每个线程的局部变量都是私有的，而全局变量、局部静态变量、分配的堆变量都是共享的。当进程结束，其创建的线程也将结束，将释放进程中的所有线程的资源。这部分可以说明下进程和线程的异同点。

9、

（1）访问 2 次内存读取页表项，再访问内存 3*200 = 600ns

（2）访问 3 次内存读取页表项，再访问内存 4*200 = 800ns

（3）TLB 命中时间为：10+200=210ns

未命中时间为：10+2*200 + 200=610ns

平均时间：210*90%+610*（1-90%）=250ns

（4）TLB 命中时间为：80%*10 +200=208ns

未命中时间为：（1-80%）*（10+200+200）+50000+10 +200= 50292 ns

平均时间：208+50292 = 50500 ns

（5）TLB 命中时间为：80%*10 +200=208ns

未命中时间为：(1-80%)*(10+200+200)+10%*80000+(1-10%)*40000+10 +200= 44292 ns

平均时间：208+44292 = 44500 ns

10、

据题意，定义相应的信号量和各个进程描述如下：

```
Semaphore   buff = 1; //信号量 buff 用于实现对缓冲区的互斥操作，其初值为 1;
Semaphore   t1 = 0;   //信号量 t1 用来控制 R1 是否可以取出消息，其初值为 0;
Semaphore   t2 = 0;   //信号量 t2 用来控制 R2 是否可以取出消息，其初值为 0;
Semaphore   t3 = 0;   //信号量 t3 用来控制 R3 是否可以取出消息，其初值为 0;
Procedure S1 {
    while(true){
       P(buff);
        向缓冲区发送消息;
       V(t1);
 }
}

Procedure R1{
    while(true){
        P(t1);
         从缓冲区中取出消息;
       V(buff);
       V(t2);
    }
}

Procedure R2{
    while(true){
        P(t2);
        p(buff);
         从缓冲区中取出消息;
       V(buff);
       V(t3);
    }
}

Procedure R3{
    while(true){
       P(t3);
       P(buff);
         从缓冲区中取出消息;
       V(buff);
    }
}
```