操作系统答案

一、

DCDBCCCA

二、

1、减少存取时间　2、最优适应分配算法　最坏适应分配算法　3、中断系统

4、分区式存储管理　5、占用　空闲　6、互斥条件　请求和保持条件

7、封闭性　可再现性

三、

1、参考答案：

进程切换的步骤如下：

（l）保存当前进程上下文环境。

（2）对当前运行进程的 PCB 进行更新。并将其移入适当的队列。

（3）挑选其他进程执行。

（4）对挑选进程 PCB 进行更新，包括将其状态改为运行。

（5）对存储器管理数据结构进行更新。

（6）恢复被选择进程上次移出时的处理器状态。

2、参考答案：

（l）S:=S-1，（S 为信号量）。（2 分）

（2）若 S<0，阻塞当前进程，将其插入 S 的等待队列，调度另一进程运行。（2 分）

（3）若 S>=0，当前进程继续运行。（2 分）

3、参考答案：

为实现进程互斥，可利用软件方法，也可在系统中设置专门的同步机制来协调诸进程，但所有的同步机制都应遵循下述 4 条准则：（2 分）

（1）空闲让进（1 分）。无进程处于临界区时，相应的临界资源处于空闲状态，因而可允许下个请求进入临界区的进程立即进入自己的临界区，以有效地利用临界资源。

（2）忙则等待（1 分）。己有进程进入自己的临界区时，相应的临界资源正被访问，所有其他试图进入临界区的进程必须等待，以保证诸进程互斥地访问临界资源。

（3）有限等待（1 分）。对要求访问临界资源的进程，应保证该进程能在有效时间内进入自己的临界区，以免陷入“死等”状态。

（4）让权等待（1 分）。当进程不能进入自己的临界区时，应立即释放处理机，以免进程陷入“忙等”。

4、参考答案：

在操作系统中，引入缓冲的主要原因，可归结为以下几点：

（1）改善 CPU 与 I/O 设备间速度不匹配的矛盾（2 分）。例如一个程序，它时而进行长时间的计算而没有输出，时而又阵发性把愉出送到打印机。由于打印机的速度跟不上 CPU，而使得 CPU 长时间的等待。如果设置了缓冲区，程序输出的数据先送到缓冲区暂存，然后由打印机慢慢地输出。这时，CPU 不必等待，可以继续执行程序。实现了 CPU 与 I/O 设备之间的并行工作。事实上，凡在数据的到达速率与其离去速率不同的地方，都可设置缓冲，以缓和它们之间速度不匹配的矛盾。众所周知，通常的程序都是时而计算，时而输出的。

（2）可以减少对 CPU 的中断频率，放宽对中断响应时间的限制（1 分）。如果 I/O 操作每传送一个字节产生一次中断，那么设置了 n 个字节的缓冲区后，则可以等到缓冲区满才产生中断，这样中断次数就减少到 lln，而且中断响应的时间也相应地放宽。

（3）提高 CPU 和 I/O 设备之间的并行（1 分）性。缓冲的引入可显着提高 CPU 和设备的并行操作程度，提高系统的吞吐量和设备的利用率。

根据 I/O 控制方式，缓冲的实现方法有两种：

（1）采用专用硬件缓冲器。（1 分）

（2）在内存划出一个具有 n 个单元的专用缓冲区，以便存放输夕口输出的数据。内存缓冲区又称为软件缓冲（1 分）。

5、参考答案：

页式存储管理首先把主存储器分成大小相等的分块，作为主存分配的物理单位，同时要求程序逻辑地址也分成与块大小一致的页面，这样就可以把作业信息按页面存放在块中。进行存储分配时，根据作业大小，确定其页面数，在装入主存时给它分配相应数目的主存块。这些主存块可以不相邻，为了在作业执行过程中准确地查找逻辑地址与绝对地址的对应关系，系统为每个作业建立一张页表，指出逻辑地址中的页号与主存块中块号的对应关系。（2 分）

页表一般存放在主存储器中，当要按给定的逻辑地址进行读/写时，必须两次访问主存，延长了指令的执行周期，降低了执行速度，为了提高存取速度，系统设置一个小容量的高速缓冲存储器，利用高速缓冲存储器存放页表的一部分，这部分页表即“快表”，利用快表可以一次访问主存完成读/写，大大缩短地址转换时间，从而提高查找速度和执行指令速度。（4 分）

四、

参考答案：

（1）访问顺序，如下表所示：

10	11	104	170	73	309	185	245	246	434	458	364
0	0	1	1	0	3	1	2	2	4	4	3

（2）采用 FIFO 算法的情况如下所示。

	0	0	1	1	0	3	1	2	2	4	4	3
块号 0	0	0	1	1	1	3	3	2	2	4	4	3
块号 1			0	0	0	1	1	3	3	2	2	4
淘汰页号						0		1		3		2
缺页中断	√		√			√		√		√		√

采用 FIFO 算法产生的缺页中断为 6 次。

（3）采用 LRU 算法的情况如下表所示。

	0	0	1	1	0	3	1	2	2	4	4	3
块号 0	0	0	1	1	0	3	1	2	2	4	4	3
块号 1			0	0	1	0	3	1	1	2	2	4
淘汰页号						1	0	3		1		2
缺页中断	√		√			√	√	√		√		√

采用 LRU 算法产生的缺页中断为 7 次。

数据结构参考答案

一、选择题（每小题 1 分，共 8 分）

1. D　　2. A　　3. C　　4. A
5. B　　6. A　　7. B　　8. D

二、填空题（每小题 1 分，共 8 分）

1. O(nlogn)　2. n-i+1　3. abcd　4. 4
5. (n+1)/2　6. DEBFCA　7. n^2-2e　8. 69

三、简答题（每小题 6 分，共 36 分）

1.

(1) 10；　(3 分)
(2) 447　(3 分)

2.

路径	长度
(v1,v5)	10
(v1,v2)	20
(v1,v5.v6)	30
(v1,v5,v6,v3)	45
(v1,v5,v6,v3,v4)	85

3.

48 70 33 65 24 56 12 92
48 70 56 92 24 33 12 65
48 92 56 70 24 33 12 65
92 70 56 65 24 33 12 48

4.

v_1,v_4,v_3,v_2,v_5

5.

	第一个被访问的结点	最后一个被访问的结点
先序遍历二叉树	根结点	叶结点
中序遍历二叉树	叶结点或无左子树结点	叶结点或无右子树结点
后序遍历二叉树	叶结点	根结点

6.

哈希表

0	1	2	3	4	5	6	7	8	9	10
33	1	13	12	34	38	27	22			

ASL = 13/8

四、算法题（共 23 分）

1. （6 分）该函数的功能是：调整整数数组 a[]中的元素并返回分界值 i，使所有＜x 的元素均落在 a[1..i]上，使所有≥x 的元素均落在 a[i+1..h]上。

2. （6 分）ABBCCADD

3. （11 分）本题有多种实现方法，一种实现方法如下：

(1)定义所需数据结构（3 分）

```
#define MaxSize 100
typedef struct SeqList{
    ElemType data[MaxSize];
    int length;
    }SeqList;
typedef struct node{
    ElemType data;
    struct node *nxt;
    }ListNode;
typedef ListNode *LinkedList;
```

(2)算法（8 分）

```
    void   ans(SeqList L,LinkList * L2){
/* L 是顺序存储的线性表; L2  为新建链表的头指针*/
 *L2 = (LinkList)malloc(sizeof(ListNode));
 L2->next = L2;                                    （建空链表 2 分）
 for i=L.length;i>0;i--){                          （1 分）
    p = (ListNode *)malloc(sizeof(ListNode));
    p->data = L.elem[i-1];                         （生成新节点 2 分）
    p->next = L2->next;
    L2->next = p;                                  （正确链入 3 分）
  }
}
```