

# 考试科目名称 操作系统 (A)

考试方式： 闭卷      考试日期\_\_\_\_\_年\_\_\_\_\_月\_\_\_\_\_日      教师\_\_\_\_\_

系（专业）\_\_\_\_\_      年级\_\_\_\_\_      班级\_\_\_\_\_

学号\_\_\_\_\_      姓名\_\_\_\_\_      成绩\_\_\_\_\_

题号	一	二
分数		

得分	
----	--

      一、综合题（每题 7 分，共 42 分）

1. UNIX 系统中，运行如下代码：

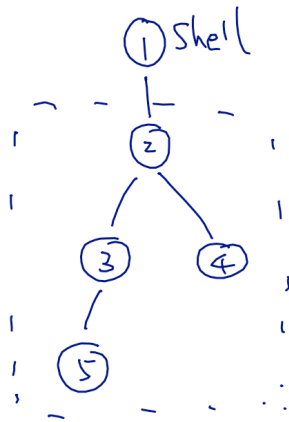
```
int main(void)
{
    int fd;
    int pid;
    char buffer[3];
    buffer[2] = 0;
    fd = open("data.txt", O_RDONLY);
    fork();
    fork();
    read(fd, &buffer, 2);
    pid = getpid(); // 获取进程ID
    printf("%d, %s\n", pid, buffer);
    close(fd);
    waitpid(-1); //等待子进程结束
}
```

假设 Shell 进程的 ID 为 1，后续创建进程所对应 ID 为依次加一，父进程总是比子进程优先调度执行，data.txt 是一个 ascii 编码的文本文件，内容为“abcdefghijklmnopqrst”，试回答如下问题：

- （一）此代码运行过程中，共产生多少进程？画出进程树（标上相应的进程 ID）。
- （二）写出代码的输出结果。
- （三）如果当前工作目录是/user/os，请简要叙述上述代码中 open 系统调用的工作过程。

答:

(一)



(二)

2:ab  
3:cd  
4:ef  
5:gh

(三)

- ① 读取根目录, 找user对应的文件数据块(inode);
- ② 读取user内容, 找zy os对应的inode;
- ③ 读取os内容, 找zy data.txt对应的inode;
- ④ 判断所读内容是否相, 创建新的inode,  
创建系统已打开文件表项, 创建用户打开文件表项。

2. 给定一组作业  $J_1, J_2, \dots, J_n$ , 它们的运行时间分别为  $T_1, T_2, \dots, T_n$ , 假定这些作业依次在时刻 0 按次序  $J_1, J_2, \dots, J_n$  进入单处理器系统。

(一) 请计算采用 FCFS 算法调度作业时的平均周转时间和平均带权周转时间。

(二) 试证明: 若按最短作业优先调度算法运行这些作业, 则平均周转时间最短。

(三) 采用最短作业优先调度算法会产生什么问题?

答:

(一)

$$\text{平均周转时间: } T_1 + \frac{n-1}{n}T_2 + \dots + \frac{1}{n}T_n, \text{ 或 } \frac{1}{n} \sum_{i=1}^n \sum_{k=1}^i T_k$$

$$\text{平均带权周转时间: } 1 + \frac{1}{n} \left( \frac{T_1}{T_2} + \frac{T_1+T_2}{T_3} + \dots + \frac{T_1+T_2+\dots+T_{n-1}}{T_n} \right), \text{ 或}$$

$$\frac{1}{n} \sum_{i=1}^n \left( \sum_{k=1}^i T_k \right) / T_i$$

(二)

首先, 对  $n$  个作业按执行时间从小到大重新进行排序, 则对  $n$  个作业:  $J_1', \dots, J_n'$ , 它们的运行时间满足:  $T_1' \leq T_2' \leq \dots \leq T_{(n-1)}' \leq T_n'$ 。那么有:

$$\begin{aligned} T &= [T_1' + (T_1' + T_2') + (T_1' + T_2' + T_3') + \dots + (T_1' + T_2' + T_3' + \dots + T_n')] / n \\ &= [n \times T_1' + (n-1) \times T_2' + (n-2) \times T_3' + \dots + T_n'] / n \\ &= (T_1' + T_2' + T_3' + \dots + T_n') - [0 \times T_1' + 1 \times T_2' + 2 \times T_3' + \dots + (n-1) \times T_n'] / n \end{aligned}$$

由于任何调度方式下,  $T_1' + T_2' + T_3' + \dots + T_n'$  为一个确定的数, 而当  $T_1' \leq T_2' \leq \dots \leq T_{(n-1)}' \leq T_n'$  时才有:  $0 \times T_1' + 1 \times T_2' + 2 \times T_3' + \dots + (n-1) \times T_n'$  的值最大, 也就是说, 此时  $T$  值最小。所以, 按最短作业优先调度算法调度时, 使得平均作业周转时间最短。

(三)

采用最短作业优先调度算法会产生长作业出现饥饿现象。

3. 设系统中有 A、B、C、D 四种资源，某时刻进程 P0、P1、P2、P3 和 P4 对资源的占用和申请情况如下表所示，系统各类资源的总数为(4, 8, 3, 4)。试回答一下问题：

	已分配资源				申请资源				最大需求资源			
	A	B	C	D	A	B	C	D	A	B	C	D
P0	1	2	1	1	0	0	1	0	1	2	2	1
P1	1	1	0	1	1	0	3	1	2	1	3	4
P2	0	2	0	1	1	1	1	2	2	4	1	3
P3	0	1	1	0	X	1	0	1	3	2	2	1
P4	1	1	0	0	0	1	3	1	2	3	3	1

- (一) 若 X=3，试用死锁检测算法判断当前系统是否存在死锁，若存在死锁，则涉及到了哪些进程？
- (二) 若 X=1，系统是否存在着死锁，若没有，则是否能够满足 P3 进程的资源申请？为什么？
- (三) 试讨论银行家算法在实际应用中存在哪些困难？

答：

- (一) 系统处于死锁状态，P1、P3、P4 进程死锁；
- (二) 不存在死锁，但不可满足 P3 进程的资源申请
- (三) 难以实现，一方面很难知道每个进程需要的资源最大数，另一方面系统中进程的数量是动态变化的；死锁检测算法可以实现。

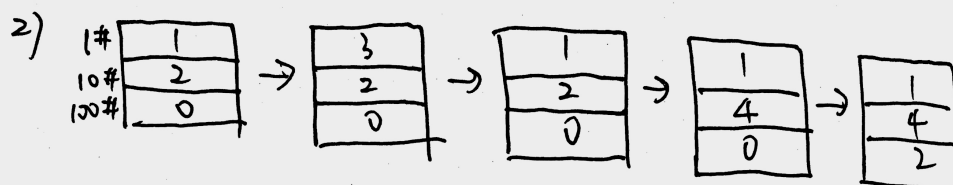
4. 一个 32 位系统的计算机，具有 1GB 物理内存，其上的操作系统采用请求式分页存储管理技术，页面大小为 2KB，页表项占 4B。试回答如下问题：
- (一) 如果采用一级页表，则页表最大包含多少个页表项？
  - (二) 如果采用二级页表，则 32 位逻辑地址该如何划分（页目录号、页号、页内偏移各占多少位）？简述二级页表相较一级页表有何优势？
  - (三) 如果采用反置页表，则反置页表包含多少个页表项？简要评述反置页表的优点和缺点。
  - (四) 如果一个进程的地址访问序列如下：2200, 4254, 1976, 6204, 4420, 502, 3110, 8240, 3510, 6034，分配给该进程 3 个固定页框，分别为 1, 10, 100（页框按编号从小到大依次分配），若采用 LRU 页面替换算法。则 1) 给出对应的页面访问序列；2) 画出页框中页面变化情况；3) 接下来要访问的逻辑地址为 3514，给出对应的物理地址。

答：

- (一)  $2^{21}$  个页表项
- (二) (22bits, 9bits, 11bits)，页表不需要连续存放，只需要部分留在内存，减少内存使用
- (三)  $2^{19}$  个页表项，优点：节约内存，缺点：地址转换效率低
- (四)

14) 1) 页面访问序列: ~~1→2→0→3→2→0→1→4→1→2~~

1→2→0→3→2→0→1→4→1→2



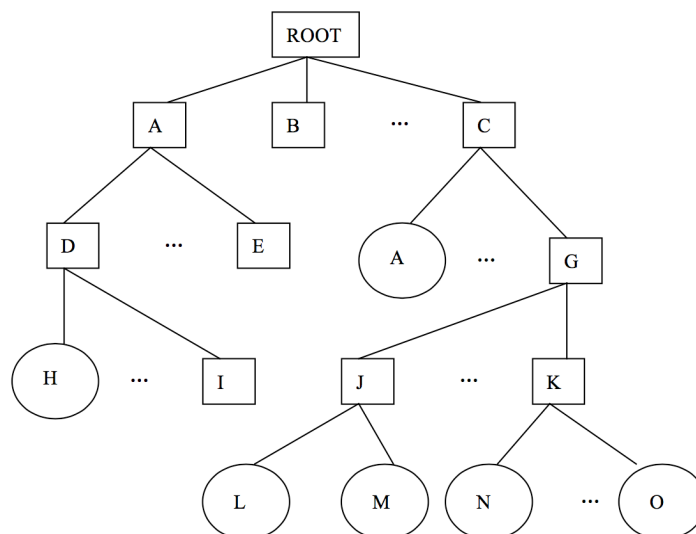
3)  $\lfloor 3514/2048 \rfloor = 1 = \text{页号}$

$3514 - 2048 = 1466 = \text{页内偏移}$

页框号 = 1

$\therefore \text{物理地址} = 1 \times 2048 + 1466 = 3514$

5. 有一个文件系统如下图所示：



图中，方形结点表示目录文件，圆形结点表示普通文件，**根目录常驻内存**。目录文件按链接结构组织，指示出下级文件名、文件类型及磁盘地址(共占 10 个字节)。若下级文件是目录，则指示其第一个硬盘块地址；若下级文件是普通文件，则指示其文件控制块的磁盘地址。每个目录文件磁盘块最后 12 个字节供链指针使用，并规定一个目录下最多存放 180 个下级文件。下级文件在上级目录文件中的次序在图中为自左向右。每个磁盘块为 512 字节。

- (一) 请说明目录文件和普通文件的主要差别。
- (二) 若普通文件按顺序结构组织，要读文件 O 的第 15 块，最少读取磁盘多少次?最多读取磁盘多少次?
- (三) 若普通文件按顺序结构组织，要读文件 L 的第 15 块，最少启动磁盘多少次?最多启动磁盘多少次?

答：

(一)

目录文件的内容为目录项，描述目录内容（如父子目录、读写属性等），不可为空；普通文件描述文件数据内容。

(二) 及 (三) 解题分析：

已知磁盘块长 512B，故供存放目录的空间为： $512-12=500\text{B}$ ，其中，12B 为链指针。由于每个目录项占 10B，且一个目录下最多存 180 个下级文件，即  $180 \times 10 = 1800\text{B}$ ，占用磁盘块数  $= 1800 \div 500 = 4$  个。由于目录文件采用链接结构，若访问的文件目录在第一磁盘块则需启动磁盘 1 次，若访问的文件目录在第四磁盘块则需启动磁盘 4 次。

(二) 若普通文件按顺序结构组织，要读文件 O 的第 15 块

最少启动磁盘次数：从内存根目录找到目录 C 的目录文件把第一块读入内存(第 1 次访盘)并查到 G，然后，再把目录 G 的目录文件第一块读入内存(第 2 次访盘)并查到 K。接着，再把目录 K 的目录文件第一块读入内存(第 3 次访盘)并查到 O。最后，把 O 的第 15 块读入内存(第 4 次访盘)，**故最少启动磁盘 4 次。**

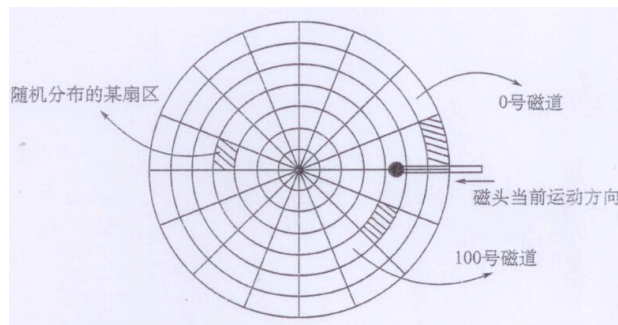
最多启动磁盘次数：读入文件 O 的第 15 块最多启动盘次数则是读入每一个目录时均在目录文件的第 4 块查到下一级目录或文件；这样，除把文件 O 的第 15 块读入内存需 1 次访盘外，由目录 C 查目录 G，由目录 G 查目录 K，由目录 K 查目录 O 都各需 4 次访盘、**最多启动磁盘为  $4 \times 3 + 1 = 13$  次。**

(三) 若普通文件按顺序结构组织, 要读文件 L 的第 15 块

最少启动磁盘次数: 同 (二), **最少启动磁盘 4 次。**

最多启动磁盘次数: 有 J 查找 L 只需一次, 故**最多启动磁盘为  $4 \times 2 + 1 + 1 = 10$  次。**

6. 设某单面磁盘旋转速度为每分钟 6000 转。共有 200 个磁道 (0-199), 每个磁道有 100 个扇区, 每个扇区 512 字节, 相邻磁道间的平均移动时间为 1ms, 若在某时刻, 磁头位于 100 号磁道处, 并沿着磁道号大的方向移动 (如下图所示), 采用 CSCAN (循环扫描) 磁盘调度策略。



试回答如下问题:

- (一) 系统使用 2KB 的内存空间记录 16384 个磁盘块的空闲状态, 请说明在该条件下如何进行磁盘块空闲状态管理。
- (二) 试计算磁盘容量。
- (三) 如果磁道号请求队列为: 50, 90, 30, 120。对请求队列中的每个磁道需读取 1 个随机分布的扇区, 则读完这 4 个扇区点共需要多少时间? (给出计算过程)

答:

(一)

用位图表示磁盘的空闲状态, 每一位表示一个磁盘块的空闲状态, 共需要  $16384/8=2048$  字节=2KB。系统提供的 2KB 内存正好表示这 16384 个磁盘块。

(二)

$$512 \times 100 \times 200 = 10240000$$

(三)

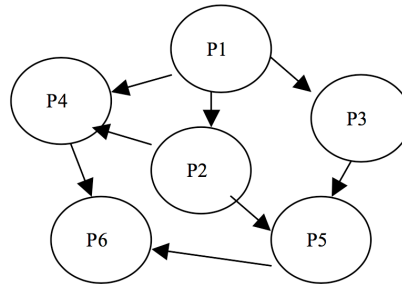
采用 CSCAN 调度算法, 访问磁道的顺序为 120, 30, 50, 90, 则移动磁道长度为  $20+79+199+30+20+40=388$ , 总的移动时间为  $388 \times 1\text{ms}=388$ 。

由于转速为 6000r/m, 则平均旋转延迟时间为  $60/(6000 \times 2) \times 1000\text{ms}=5\text{ms}$ , 总的旋转时间为  $5\text{ms} \times 4=20\text{ms}$ 。

由于转速为 6000r/m, 则读取一个磁道上的一个扇区的平均读取时间为  $10\text{ms}/100=0.1\text{ms}$ , 总的读取扇区的时间  $=0.1\text{ms} \times 4=0.4\text{ms}$ 。

读取上述磁道上的所有 4 个扇区所花费的总时间  $=388\text{ms}+20\text{ms}+0.4\text{ms}=408.4\text{ms}$ 。

1. 对于如下图所示的进程间优先图，用信号量和 P、V 写出满足此优先图的并发程序。



答：

```

var S1, S2, S3, S4, S5: semaphore;
    S1:=S2:=S3:=S4:=S5:=0;

```

```

cobegin
    P1:
    begin
        ...; V(S1); V(S1); V(S1);
    end;
    P2:
    begin
        P(S1); ...; V(S2); V(S2);
    end;
    P3:
    begin
        P(S1); ...; V(S3);
    end;
    P4:
    begin
        P(S1); P(S2); ...; V(S4);
    end;
    P5:
    begin
        P(S2); P(S3); ...; V(S5);
    end;
    P6:
    begin
        P(S4); P(S5); ...;
    end;
coend.

```