清华大学本科生考试试题专页纸	
考试课程：操作系统 B（卷）	时间：2022年06月14日下午 14:30~16:30

系别：_____	班级：_____	学号：_____	姓名：_____

答卷注意事项	1. 答题前，请先在试题纸和答卷本上写明A卷或B卷、系别、班级、学号和姓名。
	2. 要求在答卷本上答题，要写明题号，不必抄题。
	3. 答题时，要书写清楚和整洁，让判卷人清楚回答内容。
	4.请回答所有题目，本试卷有1个判断题，4个简答题，共9页。
	5. 考试完毕，必须将试题纸和答卷本一起交回。

一、(30分)判断题

请写出“√”和“×”来判断下列题目的对错。

1. [ ] 一个进程正确执行fork系统调用后，将创建(复制)一个子进程并返回子进程的PID。
2. [ ] shell进程通过执行fork+exec系统调用组合，可以创建一个子进程来执行与shell不一样的新程序。
3. [ ] 进程是一个具有一定独立功能的程序在某数据集合上的一次执行和资源使用的动态过程。
4. [ ] 操作系统采用短作业优先算法可能会出现长作业无法获得CPU资源的饥饿现象。
5. [ ] 操作系统实现FCFS（First Come, First Served）调度算法不需要硬件时钟中断的支持。
6. [ ] 在实时系统中，如果实时进程之间没有需要互斥访问的共享资源，则不会存在优先级反置现象。
7. [ ] 在多核计算机系统中，采用单队列多处理器调度的操作系统在对单就绪队列进行读写操作时，需要采用原子操作或互斥机制确保对就绪队列这个临界共享资源的互斥访问。
8. [ ] 多个进程通过页表机制可共享同一块物理内存。对这块共享内存的读写访问不需要经过系统调用。
9. [ ] 管道（pipe）是一种进程间基于字节流的单向通信机制，支持非阻塞的发送和接收数据操作。
10. [ ] 信号机制是一种进程间异步通知机制，支持一次传输4KB的字节流数据。
11. [ ] 在单核处理器计算机中，操作系统可通过禁止/使能中断来实现同步互斥机制。
12. [ ] 相对于进程，线程能减少创建/并发执行的时间和空间开销，但由于缺少地址空间隔离，所以多线程在安全性上不如多进程。
13. [ ] 在对共享资源进行分配中，如果系统处于安全状态，则一定没有死锁。

14. [ ] 管程中条件变量释放处理方式包括MESA方式、Hoare方式和Hansen方式。
15. [ ] 为支持多进程互斥访问文件而设计实现的文件锁，必须保存在持久化存储介质中。（类似为了实现硬链接引用计数而在inode中添加的nlinks字段，这些信息需保存在持久化存储介质中）

二、(70分)简答题

1. 文件系统(实践)(16分)

以下是 rCore 和 uCore 中的文件系统相关数据结构定义（有删减）

rCore：

```
pub struct OSInode {
    // ... 此处略去一些字段
    inner: UPSafeCell<OSInodeInner>,
}

pub struct OSInodeInner {
    offset: usize,
    inode: Arc<Inode>,
}

impl File for OSInode {
    // ... 此处略去
}

pub struct TaskControlBlock {
    // ... 此处略去一些字段
    inner: UPSafeCell<TaskControlBlockInner>,
}

pub struct TaskControlBlockInner {
    // ... 此处略去一些字段
    pub fd_table: Vec<Option<Arc<dyn File + Send + Sync>>>,
}
```

uCore：

```
struct file {
    // ... 此处略去一些字段
 enum { FD_NONE = 0, FD_PIPE, FD_INODE, FD_STDIO } type;
```

```
 int ref; // reference count
 struct inode *ip; // FD_INODE
 uint off;
};

struct proc {
    // ... 此处略去一些字段
 struct file *files[FD_BUFFER_SIZE];
};
```

请参考以上代码，回答问题：

1. 通常我们认为一个进程是资源的容器，这里的资源包括进程打开的文件；而文件系统负责管理文件的具体读写操作等,进程与文件系统之间通过一些数据结构建立了二者连接的桥梁。
   1.1 rCore 在 fd_table 里使用 Arc 结构和 uCore 中 struct file 里的 ref 字段都实现了引用计数功能，请说明这个引用计数功能的具体作用是什么？（Hint：考虑何时需要设置、增减引用计数，以及在引用计数为0时要做的事情）
   1.2 在支持文件系统的操作系统（ch6）中，进程执行某系统调用后，操作系统的内核数据结构中，会出现多于一处引用同一个打开的文件（**即多于一处引用同一个 dyn File 或 struct files[i]，使得**引用计数值大于1。hint:**这里的引用计数不是硬链接计数**）的现象。请举出一个会产生此现象的系统调用。（Hint：进程管理或文件相关的系统调用）

2. rCore 和 uCore 中的文件系统都支持对文件的读写操作。小饶同学写的代表两个不同应用程序的两个进程（无父子或兄弟关系）执行了如下操作（C伪代码）

P1/P2进程中定义的全局变量

```
1. char *test_str = "HelloWorld";
2. char * fname = "fname";
3. int fd=0;
4. char buffer[15]={'/0',…,'/0'};
5. int read_len=0;
```

P1（进程1）：

```
1. int fd = open(fname, O_CREATE | O_WRONLY);//创建可写文件
2. write(fd, test_str, strlen(test_str));
3. close(fd);
4. fd = open(fname, O_RDONLY); //打开只读文件
5. memset(buffer, 0, sizeof(buffer));//buffer清零
6. int read_len = read(fd, &buffer,5);
7. close(fd);
```

P2（进程2）：

```
1. fd = open(fname, O_RDONLY);
2. memset(buffer, 0, sizeof(buffer));
3. int read_len = read(fd, &buffer, sizeof(buffer));
4. close(fd);
```

2.1 如果两个进程的执行顺序是先执行P1，P1结束后，再执行P2，直到P2结束。请问当开始执行P2的第4行时，P2进程中的buffer[0..9]的内容是什么？请分别简要说明P1和P2进程中的buffer[0..9]在上述执行过程中的内容变化情况。

2.2 如果两个进程的执行顺序是先执行P1，当执行完P1的第1行时，操作系统切换到P2开始执行；当执行完P2的第3行时，操作系统切换到P1的第2行继续执行；等P1执行完毕后，再继续执行P2直到结束。请问当执行到P2的第4行时，P2进程中的buffer[0..9]的内容是什么？请分别简要说明P1和P2进程中的buffer[0..9]在上述执行过程中的内容变化情况。

3. rCore 和 uCore 中的文件系统都支持对文件的读写操作。小尤同学写的一个多线程应用进程P(进程P中有线程T1和T2)，执行了如下操作（C伪代码）：

进程P中的全局变量：

```
1. char *test_str = "HelloWorld";
2. char * fname = "fname";
3. int fd1=0, fd2=0;
4. char buffer[15]={'/0',…,'/0'};
5. int read_len1=0, read_len2=0;
```

T1（线程1）：

```
1. fd1 = open(fname, O_CREATE | O_WRONLY); //创建可写文件
2. write(fd1, test_str, strlen(test_str));
3. close(fd1);
4. fd1 = open(fname, O_RDONLY); //打开只读文件
5. memset(buffer, 0, sizeof(buffer)); //buffer清零
6. int read_len1 = read(fd1, &buffer,5);
7. close(fd1);
```

T2（线程2）：

```
1. fd2 = open(fname, O_RDONLY);
2. memset(buffer, 0, sizeof(buffer));
3. int read_len2 = read(fd2, &buffer[5], 5);
4. close(fd2);
```

3.1 如果两个线程的执行顺序是先执行T1，T1结束后，再执行T2，直到T2结束。请问当执行到T2的第4行时，buffer[0..9]的内容是什么？请简要说明buffer[0..9]在上述执行过程中的内容变化情况。

3.2 如果进程P中的两个线程的执行&切换顺序如下：

T1:1-4 //T1执行完第1-4行代码，切换到T2；

T2:1-2 //T2执行完第1-2行代码，切换到T1；

T1:5-6 //T1执行完第5-6行代码，切换到T2；

T2:3 //T2执行完第3行代码，切换到T1；

T1:7 //T1执行完第7行代码，并结束；

T2:4 //T2执行完第4行代码，并结束；

整个进程P结束。

请问当线程T1执行完毕T1的第6行时，buffer[0..9]的内容是什么？请简要说明buffer[0..9]在上述执行过程中的内容变化情况。

2. 文件系统（原理）(16分)

由于rCore和uCore的文件系统块布局不太一样，本题的描述中我们统一采用简化的unify文件系统规范。下面给出了磁盘上unify文件系统中各区域总体分布，其中各区域占用空间和具体内容后面会详细说明，各区域的最小单位与 Block 大小一致。

Super Block	Block Bitmap	Inode Bitmap	Inode Table	Data Block

Super Block: 超级块用于记录整个文件系统的信息，本题目中默认整个文件系统只有一个超级块，占用了1个Block的空间。其记录的信息包括：

- Block 与 Inode 的总数量。
- 未使用的 Block 与 Inode 的总数量。
- Block 与 Inode 的大小：本题目中 Block 大小默认为 1024 B， Inode 大小默

认为 128 B。

- 文件系统布局（不同区域起始块位置）
- 其他文件系统的元信息，无特殊说明情况下，本题目中相关计算可不考虑这些信息。

Block Bitmap：顾名思义，用来对空闲 Block 进行管理，需要根据 Block 的分配和释放进行相应修改。占用了1个Block的空间。注意：本题目中 Block Bitmap 管理的是 Data Block 的分配情况，不包括 Super Block等区域。

Inode Bitmap：与 Block Bitmap 同理，用来对空闲 Inode 进行管理，需要根据 Inode 的分配和释放进行相应修改。占用了1个Block的空间。

Inode Table：存放 Inode，在本题目中，Inode Table的空间在文件系统创建时就已分配完毕，新创建的 Inode 需要对 Inode Table 进行写入。

Data Block：存放数据，这些数据可能包括：

- 文件数据：每个 Block 内最多只能存放和一个文件有关的数据，一个文件中的数据可能占用多个 Block。
- 目录数据：对于类型为 目录 的 Inode，需要使用数据块来存放目录项（Directory Entry，简称Dirent）信息。 每个 Block 内最多只能存放和一个目录有关的数据，一个目录中的目录项可能占用多个 Block。为简化题目，Dirent 采用顺序分配的方式，释放 Dirent 后其状态被置为无效，但占用的空间不会被释放。

Inode：为简化题目，默认其包含如下信息：

- Inode 类型：文件、目录、软链接等。
- 文件大小
- 分配的 Block 数量
- 直接索引和间接索引：Inode 中有 60 B 用来记录索引，每个索引占用 4 B。前 12 个为直接索引，第 13 个为一级间接索引，第 14 个为二级间接索引，第 15 个为三级间接索引。一级索引指向的数据块为索引块，可支持256个直接索引，二、三级索引以此类推。
- 硬链接数
- 时间戳：访问时间、创建时间、修改时间
- 权限控制

1. 请计算这个文件系统最多支持多少个 Inode ？最多支持多少个 Data Block？Inode Table 需要占用多少个 Block？ 请写下计算过程。
2. 假设根目录下没有文件，不存在块缓存（Block Cache），且对磁盘的读写单位为 1024 B。在根目录下创建一个新文件（文件长度为0）最少需要多少次磁盘访问？请简要说明各次磁盘访问的目标区域。（Hint：读一块Block算一次磁盘访问，写一块

Block算另外一次磁盘访问。可以考虑文件系统只创建一个文件的情况，注意创建新文件要在根目录下创建 Dentry）

3. 该文件系统支持的最大文件大小是多少？请写下计算过程（Hint：多级索引要分配索引块，需要考虑 block bitmap的约束，结果可以是数学表达式）

4. 假设该文件系统支持软链接和硬链接。设应用程序A的执行过程中，先在当前目录D上创建了文件F1（F1的当前引用计数值为1，长度为0），然后该应用程序A在当前目录D上建立F1的符号链接（软链接）文件F2，再建立F1的硬链接文件F3，然后删除F1。此时，在目录D的目录项列表中是否记录了F1、F2和F3这三个文件？如果有被记录的文件，那么这些文件的引用计数值分别是多少？

3. 同步互斥(21分)

读者-写者问题描述如下：一个共享数据区，有若干个线程对其进行读入工作，若干个线程对其进行写入工作。有三种策略：读者优先、写者优先和公平竞争。读者优先策略是：一旦有读者线程正在读数据，允许多个读者线程同时进入读数据，只有当全部读者线程退出，才允许写者线程进入写数据。写者优先策略是：当有写者/读者线程都在等待时，只有所有写者线程写完后，读者才能获得读权限。

1. 同步互斥问题如果处理不当，很容易引入死锁问题或多个线程进入临界区的问题。所以我们需要很清楚临界区的访问规则，以及死锁的必要条件。请问：临界区的定义是什么？临界区的4个访问规则是哪些，其中那个规则是可选的？死锁的4个必要条件是什么？

2. 请用基于MESA方式的管程机制（互斥锁+条件变量）编程实现读者优先策略的读者-写者问题。请简要清晰地描述设计思路。

3. 请用信号量机制编程实现写者优先策略的读者-写者问题。请简要清晰地描述设计思路。

注意：

2，3小问请用类C伪代码并基于请基于下面给出的代码框架和参考下述变量定义与函数声明编写。请说明信号量的初始值，并给出信号量声明的含义，对代码有清晰简明的注释。请实现*start_read(), end_read(), start_write(), done_write()* 函数。

（Hint：MESA方式是指当线程 T2进入管程，使线程 T1 wait的条件满足，并发出signal操作通知 T1 后，T2 会在管程中继续执行。T1 并不会立即执行，而是在T2退出管程后，T1 与其他等待进入管程的线程一起，重新竞争管程访问权限。）

信号量相关的定义与函数说明：

semaphore s =1; //定义信号量s,其初始计数为1 （要根据需求设定具体的初始值）

P(s); //信号量s的P操作（按先进先出顺序插入等待线程）

V(s); //信号量s的V操作（按先进先出顺序弹出等待线程）

互斥锁相关的的定义与函数说明：

lock m; //定义锁，初始为锁空闲状态

acquire(m)；尝试获得锁，如得到将继续执行，否则等待（不会按时间排序）

release(m)；释放锁（不会按时间序唤醒）

条件变量相关的的定义与函数说明：

condition c; //定义条件变量

wait(c, m); //条件变量c的等待操作，会释放互斥锁m（按先进先出顺序插入等待线程）

signal(c); //条件变量c的唤醒操作（按先进先出顺序弹出等待线程）

读写数据相关的举例：

read(); //读共享数据区

write(); //写共享数据区

```
//变量定义区域
//请给出可能的信号量/互斥锁/条件变量/全局变量的定义、初值和描述
……
//读者线程的代码
read(){
  start_read();
  read(shared_data);
  done_read();
}
//写者线程的代码
write(){
  start_write ();
  write(shared_data);
  done_write();
}
//请实现start_read(), end_read(), start_write(), done_write() 函数
…..
```

代码框架

4. 银行家算法(17分)

假如操作系统种有四种资源A,B,C,D，它们一共有(7,8,7,8)个。现在有4个进程，它们的需要的总资源数目如下：

最大需求矩阵

	资源 A	资源 B	资源 C	资源 D
进程 K	6	3	2	3
进程 L	1	4	1	2
进程 M	3	4	3	1
进程 N	2	2	3	4

它们目前还需要的资源如下：

当前资源请求矩阵

	资源 A	资源 B	资源 C	资源 D
进程 K	3	2	1	1
进程 L	1	3	1	0
进程 M	0	3	2	1
进程 N	2	0	2	3

1. 请简要说明解决死锁的 3 种可能方法。
2. 请计算已分配资源矩阵。
3. 请计算当前可用资源向量。
4. 假设只有这 4 个进程以及这些资源。请分别分析下述几种请求之后系统是否安全，请说明理由或计算过程（各个请求独立，没有先后顺序或相关性）。

   - 1.K 请求(0,2,0,0)

   - 2.L 请求(1,1,0,0)

   - 3.M 请求(0,0,2,1)

   - 4.N 请求(1,0,2,0)