上 海 交 通 大 学 试 卷（ A 卷）

（ 2022 至 2023 学年 第 1 学期 ）

班级号_____ 学号_____ 姓名 _____

课程名称_____操作系统_____成绩 _____

一、汇编 （18 分）

以下为一个 AArch64 架构下简单的炸弹实验程序反汇编得到的结果，请根据这段汇编代码回答以下问题：

```
1  <+40>:   ldr   w0, [sp, #28]
2  <+44>:   cmp   w0, #0x0
3  <+48>:   b.lt  0x76c <func+64>
4  <+52>:   ldr   w0, [sp, #28]
5  <+56>:   cmp   w0, #0xe
6  <+60>:   b.le  0x770 <func+68>
7  <+64>:   bl    0x71c <explode>
8  <+68>:   ldr   w1, [sp, #28]
9  <+72>:   ldr   w0, [sp, #24]
10 <+76>:   sub   w0, w1, w0
11 <+80>:   str   w0, [sp, #44]
12 <+84>:   ldr   w0, [sp, #24]
13 <+88>:   cmp   w0, #0x0
14 <+92>:   b.ne  0x798 <func+108>
15 <+96>:   ldr   w0, [sp, #44]
16 <+100>:  cmp   w0, #0x0
17 <+104>:  b.eq  0x7a0 <func+116>
18 <+108>:  bl    0x71c <explode>
```

1. 请问第 1 行“`ldr w0, [sp, #28]`”指令的作用是什么？在 AArch64 汇编代码中，“w0”与“x0”寄存器的区别是什么？（4 分）

2. 考虑第 1 行到第 7 行代码，在内存地址[sp, #28]存储的变量满足什么条件的时候，不会触发第 7 行的 explode 函数？（4 分）

3. 第 7 行的 bl 指令执行后，哪些寄存器的值会发生变化？这些寄存器在该 bl 指令执行后的值是多少（假设第 1 行代码执行时的 PC 为 0x7e0）？（6 分）

我承诺，我将严格遵守考试纪律。

承诺人：_____

题号	1	2	3	4	5				
得分									
批阅人(流水阅卷教师签名处)									

4. 请描述 SP 寄存器在程序运行过程中的作用。（4 分）

二、内存 （29 分）

1. AArch64 的虚拟内存机制通常采用 4 级页表，每个页表页为 4KB，包含 512 个页表项，并支持 4KB、2MB、1GB 的页面大小，请根据 AArch64 下的虚拟内存机制回答以下问题：

   1) AArch64 中的 TLB 作用是什么？ASID 的作用是什么？(4 分)

   2) 使用大页的好处是什么？在怎样的场景下，大页能带来较大的性能提升？(4 分)

   3) 在仅考虑 4K 页的情况下，映射虚拟地址地址范围 0x00000000~0x00800000 每一级页表分别需要多少个页表页？（4 分）

2. 伙伴系统以及 SLAB 分配器通常被用来进行物理内存管理，请回答以下问题：

   1) 一个物理地址为 0x20000000 且大小为 64K 的物理块的伙伴块的物理地址是多少？(3 分)

   2) 伙伴系统和 SLAB 分配器都用于物理内存的管理，两者用途上的区别是什么？(3 分)

3. 基于硬件提供的地址翻译机制，操作系统为应用程序提供虚拟内存抽象。

   1) 请问延迟映射和立即映射的区别是什么？（2 分）

   2）应用程序访问非法虚拟地址时会触发缺页异常，应用程序访存时发生缺页或按需映射也会触发缺页异常，请问操作系统如何区别这三种情况？（6 分）

   3）请根据时钟算法策略填写以下表格，假设物理内存中可以放下三个页面，表格首行表示虚拟页面的访问顺序、最后一行表示是否缺页、用星号（*）表示是否访问位。（3 分）

页面访问	4	1	2	1	3	4	5	3
	4*	4*						
		1*						
是否缺页	是	是						

三、进程与调度 （20分）

1. 彩票调度和步幅调度是公平共享调度的两种实现方法。（6分）

   1) 请分别描述彩票调度和步幅调度两种策略。

   2) 请说出彩票调度和步幅调度各自的优势（各一点）。

2. 在以下程序的第7行处，父进程调用fork()产生了一个子进程。（5分）

```
1  #include <stdio.h>
2  #include <unistd.h>
3  int x = 123;
4  int main(void)
5  {
6      pid_t pid;
7      pid = fork();
8      printf("%d\n", x);
9      if (pid == 0) { // 子进程
10         x++;
11     } else { // 父进程
12         waitpid(pid, NULL, 0); // 等待子进程退出
13     }
14     printf("%d\n", x);
15     return 0;
16 }
```

   1) 第8行处，父子进程输出的x值是否相同？第14行处，父子进程输出的x值是否相同？

   2) 请解释上述现象背后的技术。

3. 进程间通信机制是微内核操作系统中的一项关键技术。（5分）

   1) 以课程实验为例，请描述ChCore中进程A与进程B的通信过程。

   2) 在ChCore中，一次完整的的进程间通信过程（进程A调用进程B，进程B返回进程A）涉及几次特权级切换，几次地址空间切换？

4. 假设系统中有100个进程，每个进程中有10个线程，在一对一线程模型下，操作系统会为这些线程一共分配多少个内核栈？（2分）

5. 当线程从用户态进入内核态（如发生系统调用）时，内核会将用户栈切换为内核栈，请问为什么需要区分用户栈和内核栈？（2分）

四、同步原语 （13 分）

1. 请简要描述信号量和条件变量各自的适用场景。（2 分）

2. 下面是一个简单的排号自旋锁（ticket lock）的实现，请回答以下问题：

```
1  struct lock {
2     volatile unsigned owner;
3     volatile unsigned next;
4  };
5
6  void lock_init(struct lock *lock)
7  {
8     lock->owner = 0;
9     lock->next = 0;
10 }
11
12 void lock(struct lock *lock)
13 {
14    volatile unsigned ticket = atomic_FAA(&lock->next, 1);
15    while (ticket != lock->owner);
16 }
17
18 void unlock(struct lock *lock)
19 {
20    lock->owner++;
21 }
```

请根据上述已有的代码，实现排号锁的 try_lock 函数，使调用者在调用 try_lock 后能快速得知拿锁是否成功而不陷入循环忙等状态。（3 分）需实现的函数原型为：

int try_lock(struct lock *lock)

返回 0 表示已经成功拿到锁，返回-1 表示当前无法立即拿到锁。你可以通过调用如下函数来使用 fetch-and-add 以及 campare-and-swap 原子指令：

int atomic_FAA(unsigned *addr, int add_val);

int atomic_CAS(unsigned *addr, unsigned compare_val, unsigned swap_val);

3. 某系统采用银行家算法来避免出现死锁。现在共有 4 个线程需求 3 种资源 A、B、C，需要的资源数如下所示，3 种资源总量分别是 10、15、8。请回答如下问题：

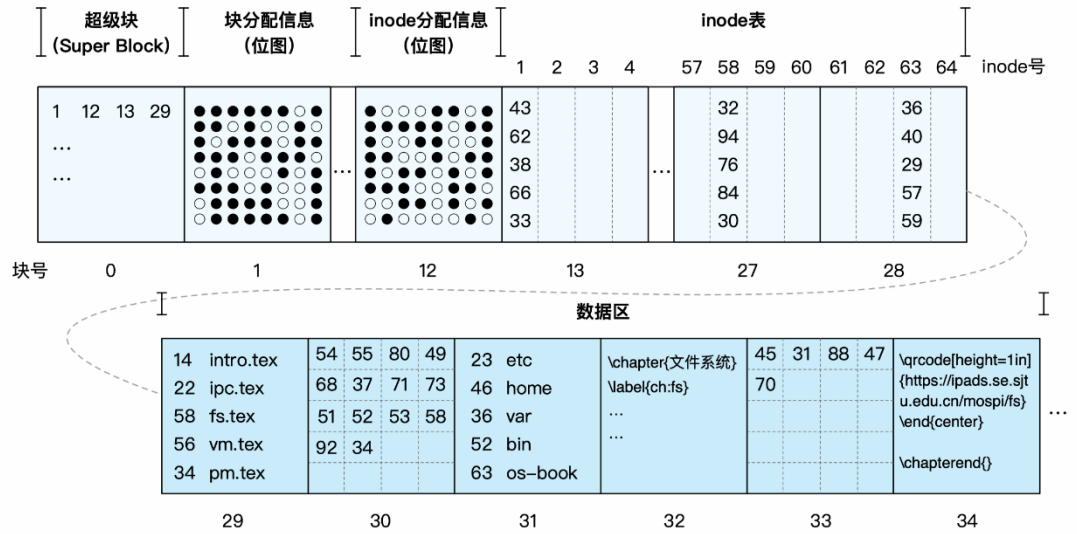
	已分配			最大需求数		
	A	B	C	A	B	C
T1	2	0	1	8	10	7
T2	0	3	1	6	12	3
T3	2	1	2	6	14	2
T4	0	1	1	8	12	8

1) 请问当前系统是否处于安全状态？如果是，请给出一个安全序列；如果不是，请简要说明原因。(3 分)

2) 此时 T4 线程请求一个 B 资源，该系统是否会立即满足其请求？请说明原因。（3 分）

4. 请给出设计程序时预防死锁出现的两种方法。（2 分）

五、文件系统 （20 分）



1. 小明正在修订本课程的教材。他认为文件系统这一章节需要增加一些内容，于是他编辑了/os-book/fs.tex，在**文件最末尾**新增了一些内容。现在假设他新增的字符刚好需要占用一个 block 大小的磁盘空间。文件系统的布局如上图所示。

   1) 请简单描述文件系统是如何找到该文件，以及如何将小明新增的内容存入磁盘的（从根目录开始，不考虑日志）（8 分）

   2) 在上图所示的文件系统中，块 30 和 33 的作用是什么？有什么好处？（3 分）

   3) 小明发现执行命令 mv /os-book /home/xiaoming 很快就完成了，而 mv /os-book /run/media/xiaoming/usb 则需要比较多的时间，试解释其原因。（3 分）（提示：/home/xiaoming 目录和 /os-book 目录位于同一个文件系统中，而 /run/media/xiaoming/usb 目录位于另外的 U 盘，/os-book 大小约为 1GB）

2. 许多文件系统都支持硬链接和软链接。小李同学执行了下列命令来研究软链接和硬链接的区别，请解释为什么命令①可以正常执行，而命令②则会出错？(2 分)（#符号后为提示）

```
$ echo os-book > a.txt
$ cat a.txt # 输出 a.txt 的内容
os-book
$ ln a.txt b.txt # 创建硬链接 b.txt 指向 a.txt
$ ln -s a.txt c.txt # 创建软链接 c.txt 指向 a.txt
$ rm a.txt
$ cat b.txt # 命令①
os-book
$ cat c.txt # 命令②
cat: c.txt: No such file or directory
```

3. 通过日志来保证文件系统的崩溃一致性会带来额外开销。Ext4 日志文件系统提供了多种日志模式，以满足用户对崩溃一致性和性能的不同需求。请简要阐述 ordered mode 和 data mode 的区别，以及它们各自的优劣。（4 分）