# 清华大学本科生考试试题纸

考试课程：操作系统（A 卷）　时间：2022 年 11 月 09 日上午 09:50~11:50

系别：_____　班级：_____　学号：_____　姓名：_____

**答卷注意事项:**

1. 答题前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. 在答卷本上答题时，要写明题号，不必抄题。
3. 答题时，要书写清楚和整洁。
4. 请注意回答所有试题。本试卷有 12 个小题，共 5 页。
5. 考试完毕，必须将试题纸和答卷本一起交回。

## 一、对错题（6 分）

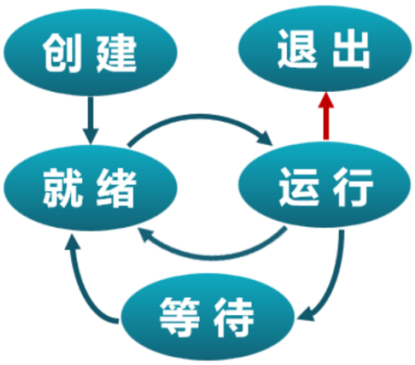
1. [ ] 利用硬件中断机制，操作系统可以在内核态及时处理用户进程的异常行为。
2. [ ] 在覆盖技术中，不存在调用关系的模块可共用同一块内存区域。
3. [ ] RISC-V 中使用 satp 寄存器记录当前地址空间的映射方式、标识符和页表的起始地址。进程切换时需要先将保存的值写入 satp 寄存器，执行 sfence.vma 刷新页表，再切换控制流。

## 二、填空题（18 分）

4. 操作系统是管理__（4.1）__ 、控制程序运行、改善人机界面和 __（4.2）__ 的一种系统软件。
5. 操作系统内核的四个主要特征是__（5.1）__ 、__（5.2）__ 、__（5.3）__ 和__（5.4）__ 。
6. 进程(Process) 是一个具有一定独立功能的__（6.1）__ 在一个__（6.2）__ 上的一次动态执行过程。
7. 局部性（locality）原理是指，程序在执行过程中的一个较短时期，所执行的指令地址和指令的操作数地址，分别局限于一定区域。“一个数据的一次访问和下次访问都集中在一个较短时期内”是__（7.1）__ 局部性。“当前访问的数据和邻近时间访问的几个数据都集中在一个较小区域内”是__（7.2）__ 局部性。

   可能的选项：时间、空间、分支等

8. 在三状态进程模型中，内核维护的进程状态会由于执行过程中的事件或操作出现而发生状态变迁。




- 创建 -> 就绪：__（8.1）__
- 就绪 -> 运行：__（8.2）__
- 运行 -> 就绪：__（8.3）__
- 运行 -> 等待：__（8.4）__
- 运行 -> 退出：__（8.5）__
- 等待 -> 就绪：__（8.6）__

  可能的选项：程序启动、进程进入就绪队列、被调度、时间片用完、等待事件、进程结束、等待事件发生等

## 三、简答题

### 9. （18 分）

请简要回答关于系统调用处理过程的如下问题。

9.1）简述在用户态程序执行系统调用的过程中，从用户态到内核执行具体系统调用服务函数前，以及从内核态执行具体系统调用服务函数返回值到返回用户态得到系统调用结果的过程中，用户态程序、硬件和内核分别需要做哪些事情。

9.2）在 RISC-V 的页表设计中，除了控制读写的 RW 权限位外，还有一个标识该页是否是所有地址空间可见的 G 标志位。当该标志位为 1 时，表示所有地址空间都可以访问该页，且刷新页表时硬件可以不刷新该页的缓存以提升效率。试举一例说明什么样的页表可以设置 G 标志位。

9.3）请依据自己的实验选择，分析函数 syscall()中汇编代码部分每一行的功能。

uCore 中用户态程序访问内核提供系统调用服务的 syscall()代码：

```
#define MAX_ARGS            5

static inline int
syscall(int num, ...) {
    va_list ap;
    va_start(ap, num);
    uint32_t a[MAX_ARGS];
    int i, ret;
    for (i = 0; i < MAX_ARGS; i ++) {
        a[i] = va_arg(ap, uint32_t);
    }
    va_end(ap);

    asm volatile (
        "int %1;"
        : "=a" (ret)
        : "i" (T_SYSCALL),
          "a" (num),
          "d" (a[0]),
          "c" (a[1]),
          "b" (a[2]),
          "D" (a[3]),
          "S" (a[4])
        : "cc", "memory");
    return ret;
}

int
sys_exit(int error_code) {
    return syscall(SYS_exit, error_code);
}

```

rCore 中用户态程序访问内核提供系统调用服务的 syscall()代码。

```
 pub fn syscall6(id: usize, args: [usize; 6]) -> isize {
    let mut ret: isize;
    unsafe {
        core::arch::asm!("ecall",
            inlateout("x10") args[0] => ret,
            in("x11") args[1],
            in("x12") args[2],
            in("x13") args[3],
            in("x14") args[4],
            in("x15") args[5],
            in("x17") id
        );
    }
    ret
}

pub fn sys_openat(dirfd: usize, path: &str, flags: u32, mode: u32) -> isize {
    syscall6(
        SYSCALL_OPENAT,
        [
            dirfd,
            path.as_ptr() as usize,
            flags as usize,
            mode as usize,
            0,
            0,
        ],
    )
}

```

## 10. （10 分）

请简要回答关于 stride 调度算法的如下问题。

10.1）描述 stride 调度算法的工作原理。

10.2）stride 调度算法中每个进程设置一个表示该进程当前已经运行的“时间长度”。假定用无符号整型变量 pass 来表示这个不断累加的“时间长度”，简要回答如何解决 pass 的整数溢出问题。

## 11. （20 分）

请简要回答关于虚拟存储管理的如下问题。

11.1）已知 RISC-V 的 Sv39 页表使用 39 位的虚拟地址对应 56 位的物理地址，请结合图示简述从虚拟地址找到对应的物理地址的过程。

11.2）SV39 的页表项（Page Table Entry）在什么情况下会成为页目录项（Page Directory Entry）？（回答标志位组合）

11.3）请简要描述采用 SV39 时系统调用 mmap()在陷入内核后的执行过程（注：不用给出边界检查的细节）。

11.4）在内核地址空间的创建过程中，为下列地址区间建立恒等映射并立刻分配物理页帧，请问共需要多少个物理页帧？（提示：假设此时页表为空，页表也需要分配物理页帧）

```
.text   [0x8020_0000, 0x8020_9000)
.rodata [0x8020_9000, 0x8020_c000)
.data   [0x8020_c000, 0x8020_d000)
.bss    [0x8020_d000, 0x80a1_e000)
```

## 12.（30 分）

设想有一种四叉伙伴系统，用于**连续内存分配**。它把伙伴系统的二叉树改成了四叉树，即:

设当前可分配的块大小为 $4^U$

- 如果需分配的块大小在 $(3\times4^{(U-1)},4^U]$ 之间，则把整块分配给应用；
- 如果需分配的块大小在 $(2\times4^{(U-1)}, 3\times4^{(U-1)}]$之间，则均分四块，把前三块作为整体分配给应用，产生一个大小为 $4^{(U-1)}$的空闲块；
- 如果需分配的块大小在 $(4^{(U-1)}, 2\times4^{(U-1)}]$之间，则均分四块，把前两块作为整体分配给应用，产生两个大小为 $4^{(U-1)}$的空闲块；
- 否则，需分配的块大小小于等于 $4^{(U-1)}$，将当前区间划分成四个 $4^{(U-1)}$ 大小的空闲块，然后在第一块中递归上面的划分过程

它的释放和合并也类似伙伴系统。当四块空闲块满足以下条件时可以合并：

- 大小相同，都是 $4^{(U-1)}$

- 地址相邻
- 低地址空闲块起始地址为 $4^U$ 的倍数

12.1）请简要描述伙伴系统（Buddy-system）的工作原理。

12.2）四叉伙伴系统的策略比较类似以下的 _____

A.最先匹配
B.最佳匹配
C.最差匹配

12.3）假定初始状态时，给定相同的分区大小，分区初始为空，每层大小分别是 ...16MB, 4MB, 1MB, 256KB... 现在有一个用户程序，请求分配的**连续**内存块 99% 都是 1.5MB, 2.5MB, 800KB 三种大小之一，那么：如果其中 2.5MB 和 800KB 大小的分配请求占绝大多数，使用 __（12.3.1）__ 分配器的效果更好；如果其中 1.5MB 大小的分配请求占绝大多数，使用 __（12.3.2）__ 分配器的效果更好。为什么？

可能选项：伙伴系统、四叉伙伴系统等

12.4）判断下面的说法是否正确，并给出例子或原因。

假定初始状态时，给定同样的分区大小，分区初始为空；执行同样的内存分配/内存释放请求序列。每次分配都能成功。一次分配的“内碎片”定义为每次用户请求的分配大小和实际分配大小的差值；分配器产生的“内碎片”定义为每次分配产生的“内碎片”长度和。

12.4.1. 相比原来的伙伴系统，这个分配器将产生更少内碎片

12.4.2. 相比伙伴系统，四叉伙伴系统将产生更多外碎片

12.4.3. 存在一种请求序列，使得未分配的内存中明明有对应长度的空区间，伙伴系统却无法分配

12.4.4. 存在一种**只有内存分配没有内存回收**的请求序列，使得未分配的内存中明明有对应长度的空区间，新分配器却无法分配