清华大学本科生考试试题纸	
考试课程：操作系统（A卷）	时间：2021年10月25日下午03:20~05:20

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18个小题，共6页。
	5.考试完毕, 必须将试题纸和答卷本一起交回。

一、(10分)判断题

1. [ ] 采用微内核结构的操作系统会尽可能把内核功能移到用户空间，以实现保护与控制的分离。
2. [ ] 在采用自映射机制的riscv Sv32系统中，satp、第一级页表项和第二级页表项这三个地方的物理页号都是相同的，指向同一个物理页，但对页面内容的解释是不同的。
3. [ ] 用户线程需要维护自己的寄存器和栈。
4. [ ] 当时间片增大时，时间片轮转（RR）调度算法可退化成短作业优先（SJF）调度算法。
5. [ ] 优先级天花板协议是指占用资源进程的优先级和所有可能申请该资源的进程的最高优先级相同。

二、(22分)填空题

6. 假定在动态分区内存分配算法中采用紧凑的碎片整理策略。在某次碎片整理之后，紧接着进行了一次分配，这里不会产生新的__**(6)**__。

   填空内容建议：内碎片、外碎片

7. 在采用伙伴系统（Buddy System）的内存管理系统中，整体内存大小为2^y个内存单元。假定`y>x`，当一个大小为2^x个单元的内存块被释放时，最少会有__**(7.1)**__次合并，最多会有__**(7.2)**__次合并。

8. 快表（TLB）是为减少查页表带来的访问内存频率而启用的机制，在__**(8)**__时需清空快表。

   填空内容建议：进程切换、线程切换、协程切换

9. 局部性原理是指，程序执行过程中的一个较短时间内，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。数组的连续存放是__**(9.1)**__局部性，循环体内指令的多次重复是__**(9.2)**__局部性。

   填空内容建议：时间、空间、分支

10. 抖动指进程数量过多而各个进程的常驻集不足时，进程常驻页面频繁在各个进程工作集之间切换，进而导致__**(10)**__的现象。

11. CFS调度算法的全称是“__**(11)**__”。

12. BFS调度算法是__**(12)**__多处理机调度算法。

    填空内容建议：单队列、多队列

13. 优先级反置(Priority Inversion)是指，操作系统中出现__**(13.1)**__进程长时间等待__**(13.2)**__进程所占用资源的现象。

三、(68分)简答题

14. (20分)

1. ）为方便内核处理，uCore 和 rCore 处在内核态时都是关中断的。请问：1.1）是哪部分逻辑关闭了中断？1.2）在回到用户态后，又是哪部分逻辑重新打开了中断？

2. ）由于内核关中断，在内核态碰到时钟中断是不处理的。请问如下命题正确吗？简单解释原因。
   命题：对于 RR 调度算法，如果恰好有一个中断发生在内核态，该中断会被永远忽略，导致 RR 调度精度收到影响。

3. ）在用户态执行系统调用的时候，程序库（如 libc）会将系统调用参数按照规定置入指定寄存器。在RISC-V上存放参数的寄存器是 `a0-a7`。小贺发现这种 syscall 的调用方式最多只能传入系统调用的6 个参数，但是他想实现一种可以传入更多参数的系统调用机制。结合编译课所学，他打算把系统调用参数放在用户态的栈上，参数的压栈顺序从右到左。假定他想实现可以传入 64 个参数的系统调用，用户库系统调用执行流程伪代码如下：

```
const int ARG_NUM = 64;
long syscall(long id, long arg[ARG_NUM]) {
    for(int i = ARG_NUM - 1; i >= 0; i--) {
        // 伪指令 stack_push 类似 x86 的 push 指令，效果为：sp = sp -
sizeof(long), *sp = arg.
        stack_push(arg[i]);
    }
    register long a7 __asm__("a7") = id;  // a7 = id
    asm("ecall");
}
```

现在假定你是内核开发者，需要在 ch2 的基础上，通过修改代码来实现上述机制。部分代码修改已经给出，你的具体任务是，实现 `get_args` 函数，该函数需要利用 `trapframe` / `TrapContext` 参数填充 args 结构体。提示：实现使用伪代码即可。

C 版代码：

```
const int ARG_NUM = 64;
void usertrap(struct trapframe *trapframe)
{
  uint64 cause = r_scause();
  if (cause == UserEnvCall) {
    trapframe->epc += 4;
+       uint64 args[ARG_NUM];
+       int id = get_args(trapframe, args);
-       syscall();
+   syscall(id, args);
    return usertrapret(trapframe, (uint64)boot_stack_top);
  }
  // ...
}
int get_args(struct trapframe *trapframe, uint64* args) {
  int id = trapframe->a7;
  // YOUR WORK
  // 提示：使用任何风格伪代码皆可
  return id;
}
```

Rust 版代码：

```
const ARG_NUM: usize = 64;
#[no_mangle]
pub fn trap_handler(cx: &mut TrapContext) -> &mut TrapContext {
    let scause = scause::read();
    match scause.cause() {
        Trap::Exception(Exception::UserEnvCall) => {
            cx.sepc += 4;
+           let mut args: [usize; ARG_NUM] = [0; ARG_NUM];
+     get_args(cx, &mut args);
-           cx.x[10] = syscall(cx.x[17], [cx.x[10], cx.x[11],
cx.x[12]]) as usize;
+           cx.x[10] = syscall(cx.x[17], &args)
        }
    }
```

```
    // ...
    cx
}

pub fn get_args(cx: &mut TrapContext, args: &mut [usize]) {
  // YOUR WORK
  // 提示：使用任何风格伪代码皆可，不一定要按照 rust 语法约定
}
```

15. (10分)

1. ）操作系统内核的页面置换算法的功能是什么?
2. ）局部页面置换算法与全局页面置换算法的区别是什么?
3. ）简述Belady现象，并指出两种会有Belady现象的页面置换算法。
4. ）时钟页面置换算法的改进算法的改进目标是什么?
5. ）假定某进程的内存页访问序列如下：

```
a, a, b, a, d, a, c, d, b, d, a, c, b, b
```

请回答，当分配给该进程的页面数为 3 个时，对LRU、FIFO 两种内存置换算法而言，会发生多少次缺页？假定内存初始化为空。

16. (20分)

1. ）请结合图示简要描述多级页表的工作原理。
2. ）请结合图示简要描述反置页表的工作原理。
3. ）在一台假想的采用多级页表的计算机系统中，页大小（page size）为16 字节（Byte），支持256字节的虚拟地址空间（virtual address space），有256字节的物理内存空间（physical memory）。采用二级页表，页目录项（第一级页表项，page directory entry，PDE）大小为1 字节；页表项（第二级页表项，page-table entries PTEs）大小为1 字节，1个页目录表大小为4 字节，1个页表大小为4 字节。页目录基址寄存器（page directory base register，PDBR）保存了页目录表的物理地址（按页对齐）。

虚拟地址格式（8 bit）：

```
VPN12 VPN11 VPN22 VPN22 OFF3 OFF2 OFF1 OFF0
```

物理地址格式（8 bit）：

```
PPN3 PPN2 PPN1 PPN0 OFF3 OFF2 OFF1 OFF0
```

PDE格式（8 bit）：

```
  PT3 PT2 PT1 PT0 I R W X
```

PTE格式（8 bit）：

```
  PFN3 PFN2 PFN1 PFN0 I R W X
```

其中

```
PFN3 PFN2 PFN1 PFN0：物理页号（页帧号，Frame）
PT3 PT2 PT1 PT0:页表物理页号>>4
I(Inaccessible): I==1表示映射存在；I==0表示映射不存在。
R(Readable): R==1表示页面可读；R==0表示页面禁止读。
W(Writable): W==1表示页面可写；W==0表示页面禁止写。
X(Executable): X==1表示页面可执行；W==0表示页面禁止执行。
```

在如下的物理内存模拟数据文件中，给出了256字节的物理内存空间的值。

```
Physical Address 0x00: FC 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00
Physical Address 0x10: BC 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00
Physical Address 0x20: 00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00
Physical Address 0x30: 00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00
Physical Address 0x40: 00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00
Physical Address 0x50: 00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00
Physical Address 0x60: 00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00
Physical Address 0x70: 00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00
Physical Address 0x80: CA FE BA BE  00 00 00 00  00 00 00 00  00 00 00 00
Physical Address 0x90: AA AA AA AA  AA AA AA AA  AA AA AA AA  AA AA AA AA
Physical Address 0xa0: 00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00
Physical Address 0xb0: 8C 9A A9 00  00 00 00 00  00 00 00 00  00 00 00 00
Physical Address 0xc0: DE AD BF EF  00 00 00 00  00 00 00 00  00 00 00 00
Physical Address 0xd0: BB BB BB BB  BB BB BB BB  BB BB BB BB  BB BB BB BB
Physical Address 0xe0: 00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00
Physical Address 0xf0: CC DE E9 00  00 00 00 00  00 00 00 00  00 00 00 00
```

假定PDBR为`0x10`，请回答下列虚地址是否有合法对应的物理内存，同时给出对应的第一级页表项的物理地址和内容，第二级页表项的物理地址和内容，物理内存单元的物理地址和内容。

```
3.1) Virtual Address 0x03
3.2) Virtual Address 0x17
3.3) Virtual Address 0x29
3.4) Virtual Address 0x3c
```

注：3.1~3.4这四个小题中，可以有一个小题不做或做错，不计分。

17. (10分)

请阅读下列代码，然后回答问题：

```
#include <stdio.h>
#include <sys/types.h>
#include <unistd.h>
#include <stdlib.h>

int main() {
    for (int i = 0; i < 2; ++i)
        if (fork() || (fork() && fork()) || !fork())
            printf("A\n");
}
```

1. ）执行`fork()`系统调用后，父进程和子进程的返回值分别是什么?
2. ）假设程序正常运行，将输出多少个 "A"? 要求简要说明理由。
3. ）如果将 `printf("A\n")` 替换为则 `printf("A")`，则程序最终输出的 “A" 的数量明显增多，为什么？不要求给出此时程序的输出结果。

18. (8分)

针对实验中ch3的编程和课后习题之中涉及了Stride的一些内容。请回答如下关于Stride算法的问题。

1. ）假如在框架中设定`min priority = 1 , 2, 3`的时候，其分别对应的BigStride范围是多少?
2. ）下面是小明同学给出的判断每一个进程的`stride`大小的比较函数:
   `(int) ( proc_a.stride - proc_b.stride) > 0`来作为 `proc_a > proc_b`的比较。这个比较函数是正确的吗？如果是，请证明。如果不是，请指出在什么情况下会出错（假设`BigStride`在算法正确的范围之中）。