操作系统 A 卷参考答案及评分标准

一、课程目标 1：

1.（7 分）

(1) 实时系统是指在规定时间内对输入事件作出响应，并在规定时间内完成其处理任务的计算机系统。

(2) 关键特性包括：

- **确定性**：任务必须在预定的时间内完成。
- **响应时间可预测**：处理延迟可控。
- **高可靠性与稳定性**：系统需要长期运行且具备错误恢复能力。
- **并发处理能力**：通常涉及多个任务并发执行。

(3) 应用场景示例：

- 汽车防抱死系统（ABS）：必须在毫秒级内响应车轮滑动状态，防止制动失控。

2.（7 分）

屏幕输出：12345

out.txt 内容：6789a

3.（6 分）

(1) /D/DC/DDC/Zhao

(2) 根目录（获取条目 D），目录 D（获取条目 DC）、目录 DC（获取条目 DDC），目录 DDC(获取条目 Zhao)

(3) 每个目录项占 256B，每块 512B ⇒ 每块存 2 个项。**需要读 8 次才能找到 Zhao**

4.（10 分）

(1) 0x5200 0000 是**逻辑地址**（即虚拟地址）。

(2) 地址转换流程：

- CPU 发出逻辑地址。
- MMU（内存管理单元）将逻辑地址分为页号与页内偏移。
- 查页表得到对应物理页框号。
- 与偏移合并，形成物理地址。
- 访问主存得到数据。

(3) 越界识别：

- CPU 提供的页号超出当前进程的页表范围，MMU 会抛出“地址越界”异常。

(4) 缺页识别与处理：

- 页表中该页无效（Valid 位为 0）时即为缺页。
- 系统中断处理：触发缺页异常 → 保存当前状态 → 查找页在磁盘中的位置 → 将页加载到空闲物理页框 → 更新页表 → 恢复指令执行。

5.（5 分）

(1) 工作原理：
DMA（Direct Memory Access）由专门的控制器直接在外设和内存之间传输数据，无需 CPU 干预传输过程。
(2) 优点：

- 减少 CPU 负载，提高系统效率。
- 传输速度快，适合大块数据。
- CPU 可以并行处理其他任务。

(3) 应用场景：

- 音视频播放中，音频解码器通过 DMA 将数据传输到声卡缓冲区。

二、课程目标 2：

1.（10 分）
(1) 2 个
(2) A: y=10
(3) 在子进程中执行
(4) B: y=8
(5) 不会出现僵尸进程

2.（15 分）
(1)不一定为 0。由于线程间的竞争条件，可能出现多个线程同时读取到 balance > 0，并各自执行取款操作，导致余额变为负数。
(2)

```
if (balance >= 200) { ... }  // 检查
 balance -= amount;        // 修改
```

这两个操作不是原子的，可能被其他线程打断，导致多线程重复取款。
(3)（6 分）
使用互斥锁保护共享资源 balance:

```
pthread_mutex_t lock;  // 全局锁
// 初始化锁（在 main 函数开始处）
pthread_mutex_init(&lock, NULL);

// 修改 withdraw 函数
void* withdraw(void* arg) {
    while (1) {
        pthread_mutex_lock(&lock);  // 加锁
        if (balance <= 0) {
            pthread_mutex_unlock(&lock);  // 解锁后退出
            return NULL;
        }

        int amount = (balance >= 200) ? 200 : balance;
        balance -= amount;
        printf("用户%d 取款%d 元，新余额：%d\n", id, amount, balance);
```

```
            pthread_mutex_unlock(&lock);  // 解锁

            usleep(DELAY);  // 释放 CPU，避免忙等待
        }
    }
```

(4)是的。互斥锁保证了对 balance 的检查和修改是原子操作，任一时刻只有一个线程能执行取款逻辑。当余额变为 0 时，其他线程将无法进入取款流程，从而避免余额为负。（3 分）

---

3. (10 分)

```
    // 全局信号量
    semaphore water_boiled = 0  // 水烧开的信号量（初始 0：未烧开）
    semaphore tea_made = 1      // 茶泡好的信号量（初始 1：允许烧水）

    // 烧水线程
    procedure BoilWater() {
        while (true) {
            wait(tea_made)       // 等待允许烧水（茶已泡好）
            boil_water()         // 烧水操作
            print("水已烧开！")
            signal(water_boiled) // 通知可以泡茶
        }
    }

    // 泡茶线程
    procedure MakeTea() {
        while (true) {
            wait(water_boiled)  // 等待水烧开
            make_tea()          // 泡茶操作
            print("茶已泡好！")
            signal(tea_made)    // 通知可以再次烧水
        }
}
```

三、课程目标 3

1. (5 分)

原因：

- RR（时间片轮转）适用于交互式系统，是因为它公平地为每个进程分配 CPU 时间。
- 每个进程最多运行一个时间片，从而保证了系统响应及时，适合用户频繁输入输出的场景。

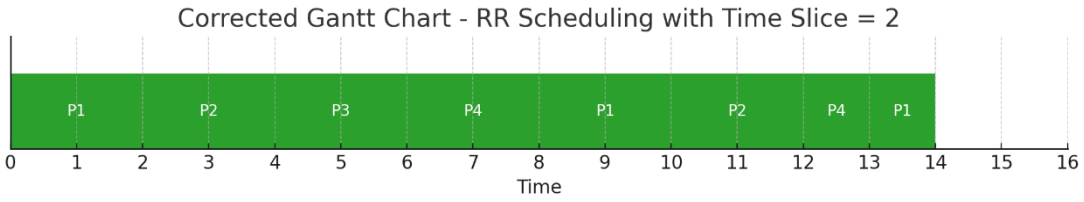
时间片影响分析：

- 时间片太小：频繁上下文切换，导致系统开销大，性能下降。
- 时间片太大：退化为 FCFS，响应延迟增加，失去交互系统优势。
- 合适的时间片：在切换开销与响应及时性之间取得平衡。

---

2. (6 分)

(1)



(2)

作业	到达	完成	周转时间（完成 - 到达）
P1	0	14	14
P2	1	12	11
P3	2	6	4
P4	3	14	11

平均周转时间 = (14 + 11 + 4 + 10) / 4 =**9.75**

---

3. (5 分)

目的:

- 系统在资源分配前检查是否处于“安全状态”，确保存在某种分配顺序使得所有进程都可顺利完成，避免进入死锁状态。

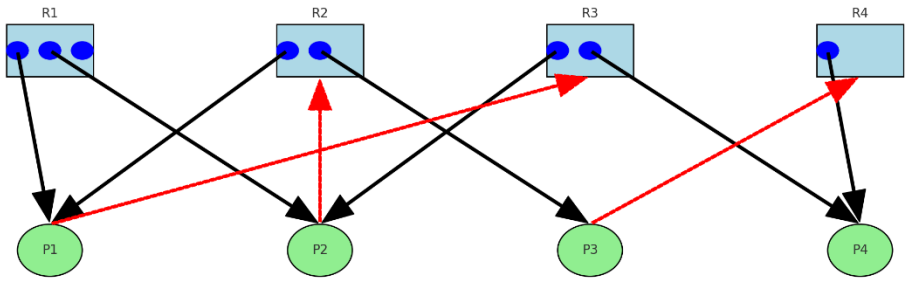
不检查可能后果:

- 若分配前未进行安全性检查，可能导致系统进入不安全状态，一旦资源无法满足某些进程的最小需求，会产生死锁，导致部分或全部进程无法继续执行。

---

4. (6 分)

(1) 构建资源分配图:



(2) 死锁判断:

- P_4已完成，无请求，释放 R_3和 R_4。
- P_3请求 R_4，释放后即可完成。
- P_1请求 R_3，也可获 R_3完成。
- P_2请求 R_2，P_1释放后获得，亦可完成。

结论：系统**无死锁**，通过资源释放后，所有进程都可完成。

5. (8 分)

(1) 逻辑地址位数:

- 页号位数：$\log_2(2^{20}) = 20$ 位
- 页内偏移：$\log_2(4KB) = 12$ 位
- 所以逻辑地址需 **32 位**

(2) 物理地址位数:

- 主存为 512MB = 2^{29} 字节
- 所以物理地址需 **29 位**

(3) 第三次访问地址 3F55H：

- 十六进制 3F55H = 二进制 0011 1111 0101 0101
- 页面大小为 4KB（12 位偏移），页号 = 3F55H >> 12 = 0x3（页号为 **3**），偏移 = 0xF55
- 页号 3 不在当前页表中 → 缺页 → 触发置换（LRU）

当前帧情况（页号 → 帧）:

- 初始：5 → 20，8 → 22（21 空着）
- 第三次访问缺页，分配帧 21 给页 3

物理地址 = 页框号 21 * 4KB + 0xF55= 21 × 4096 + 3925 = 86016 + 3925 = **89941 (十进制)** = **15F55H (十六进制)**

当前驻留集页面号：**[5, 8, 3]**