E09：UNIX V6++中进程的调度状态

参考答案与说明

1. 【参考答案】

答：存在核心态进程从运行状态转变为就绪状态的情况。UNIX 中的进程状态转换都是发生在核心态下的。一个用户态下执行的进程，因为响应中断请求或系统调用，而由用户态进入核心态执行，由于中断或系统调用返回用户态前的例行调度，可能会下台，这时候，进程的状态由运行状态转变为就绪状态。

需要注意的是，这时，进程在核心态下的工作都已经完成，是一个马上要回到用户态的进程，因为我们把这种就绪状态称为用户态就绪。用户态就绪的进程优先数是由计算获得的，它和核心态就绪是不同的。核心态就绪指的是进程由于系统调用入睡，醒来之后进入就绪状态，这时进程的优先数是设置获得的，优先级较高，因而可以尽快上台完成后续的核心态下需要完成的工作。

不存在用户态进程从运行状态转变为就绪状态的情况。因为 UNIX 中的进程状态转换都是发生在核心态下的。

2. 【参考答案】

```
static const unsigned long USER_ADDRESS = 0x400000 - 0x1000 + 0xc0000000;
/* 0xC03FF000 */
User& Kernel::GetUser()
{
        return *(User*)USER_ADDRESS;
}
```

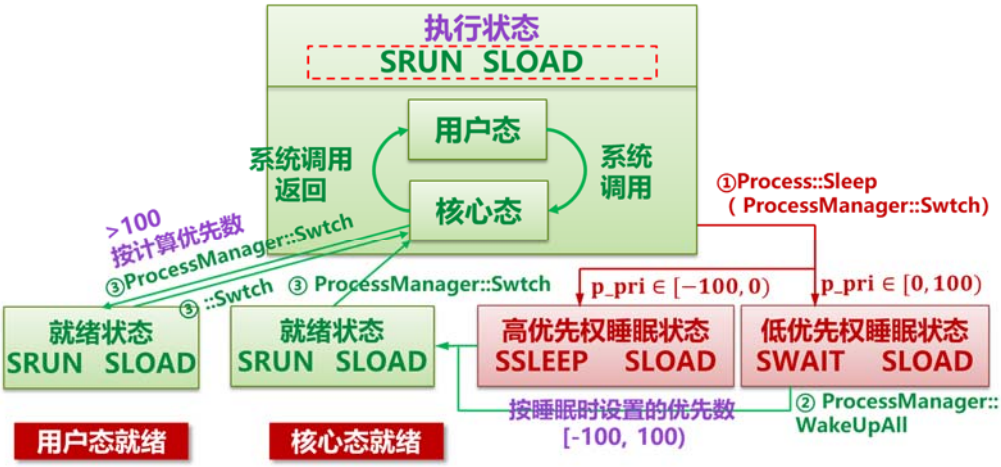
先通过调用::GetUser 函数获取现运行进程 ppda 区的虚拟地址，0xC03FF000。

该地址经过现运行进程核心态页表的地址变换后，获得现运行进程 ppda 区的起始物理地址；

利用 User 结构在的 u_procp 找到现运行进程的 proc 结构。

3. 【参考答案】反映了 UNIX 核心态不调度的重要思想。

4. 【参考答案】



（1）优先数不同：核心态就绪的进程是用过设置获得的优先数；而用户态就绪的进程是通过计算获得的优先数；

（2）上台后执行的代码不同：核心态就绪的进程上台后继续执行核心态下没有完成的代码；而用户态就绪的进程上台后，马上中断返回用户态，继续执行用户态代码。

需要注意的是，尽管二者有很大的区别，但是它们当前都处在核心态。