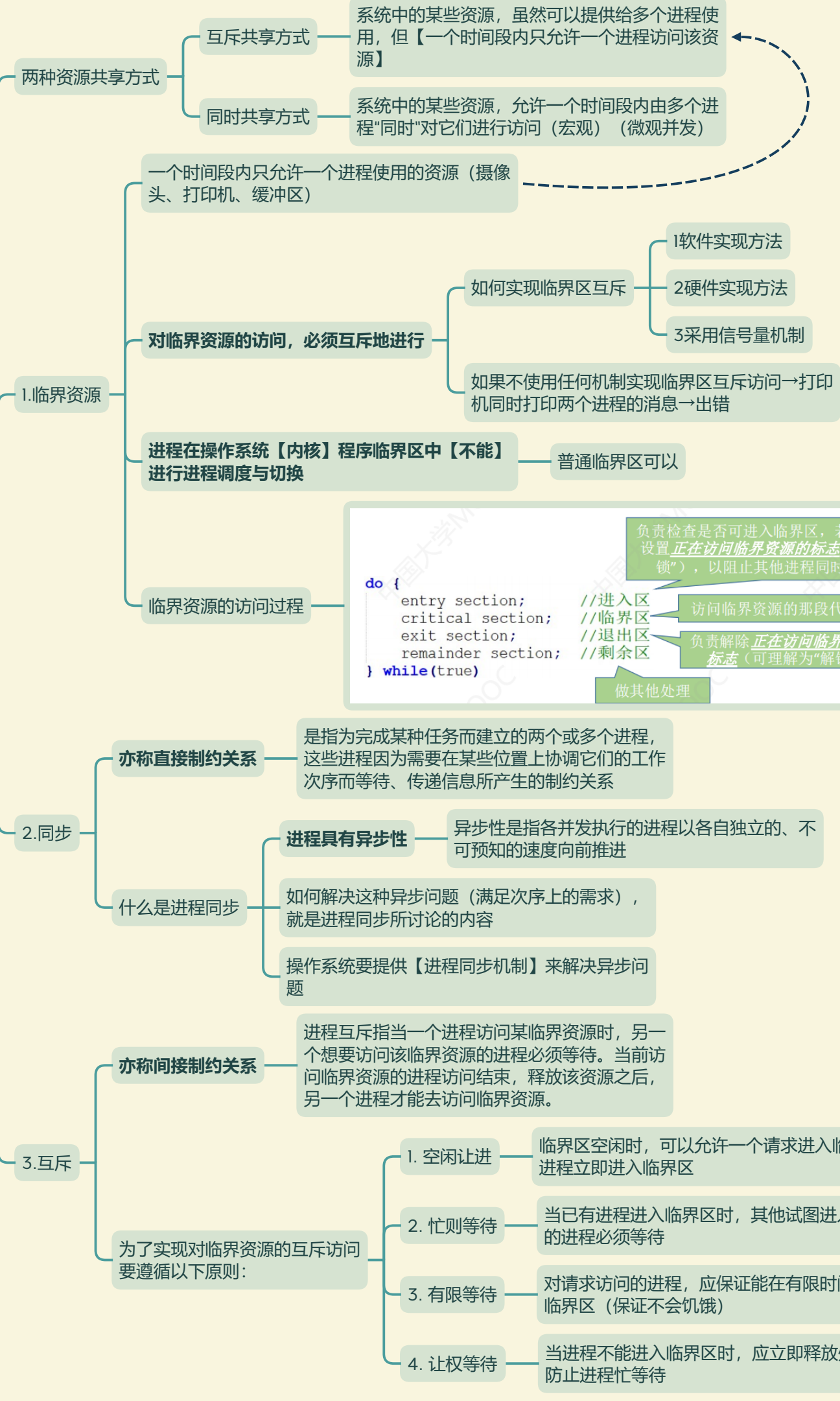


基本概念

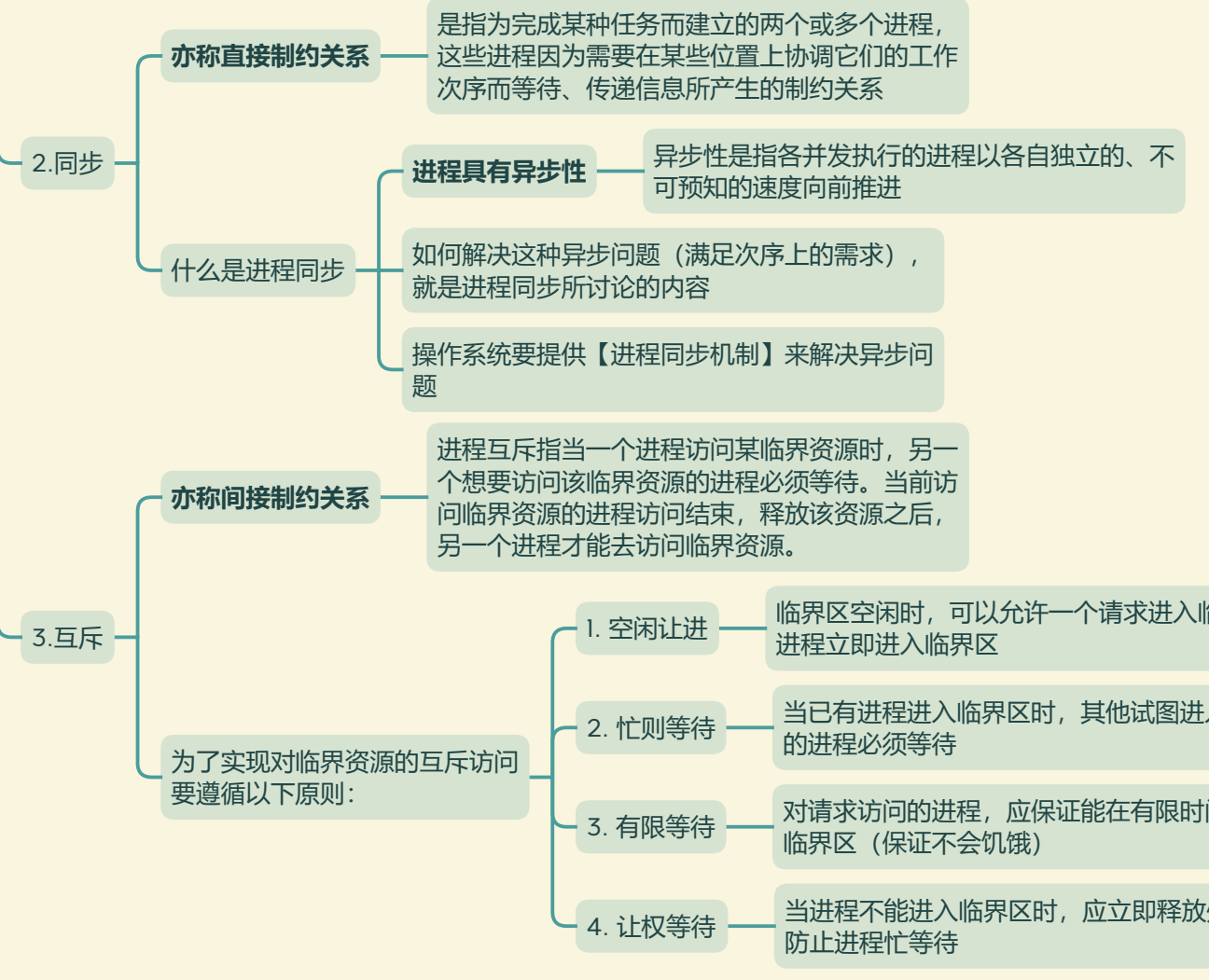


临界区——是进程中访问临界资源的代码段

临界段

进入区和退出区——是负责实现互斥的代码段

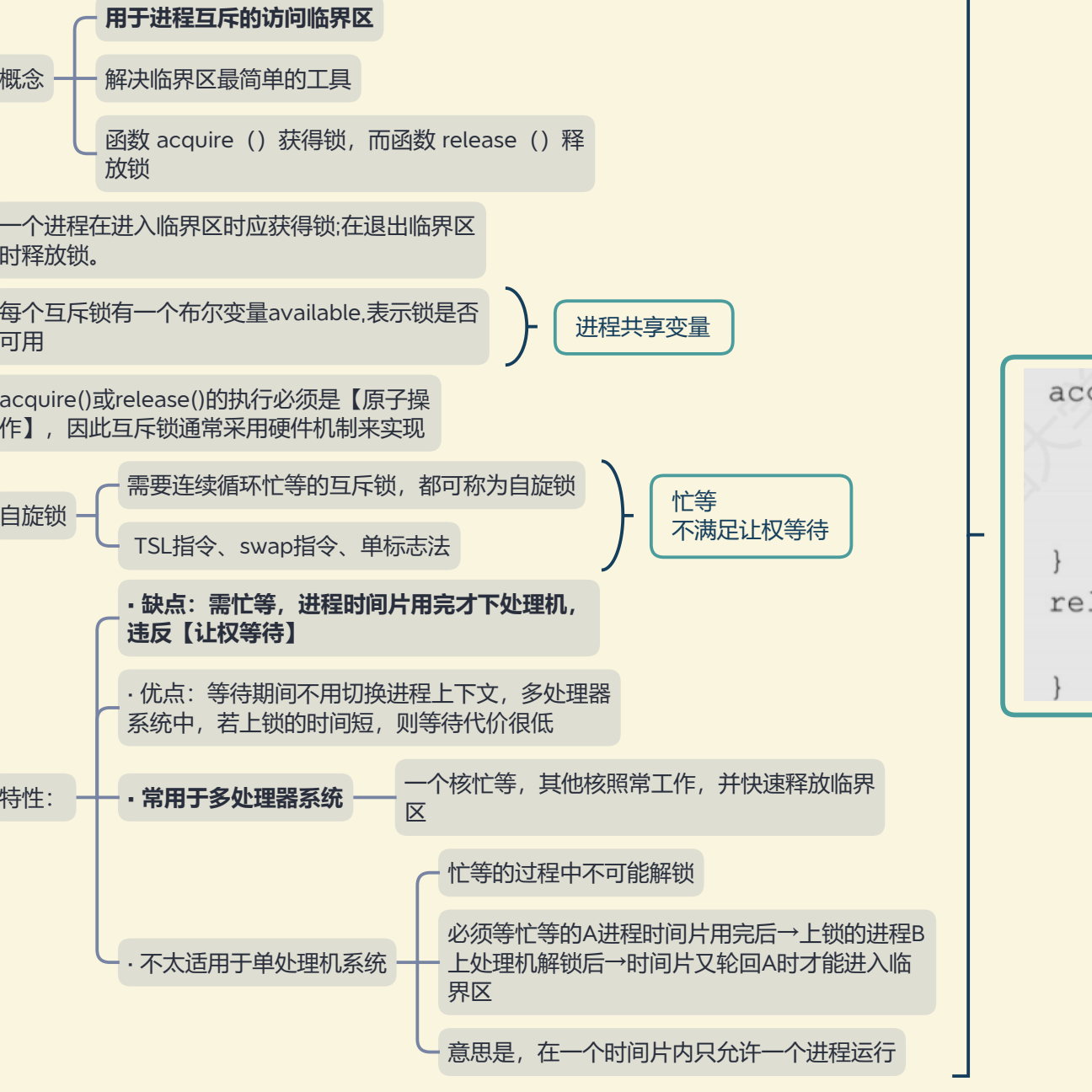
课后题常出现



问必须遵守的是——没有标准答案

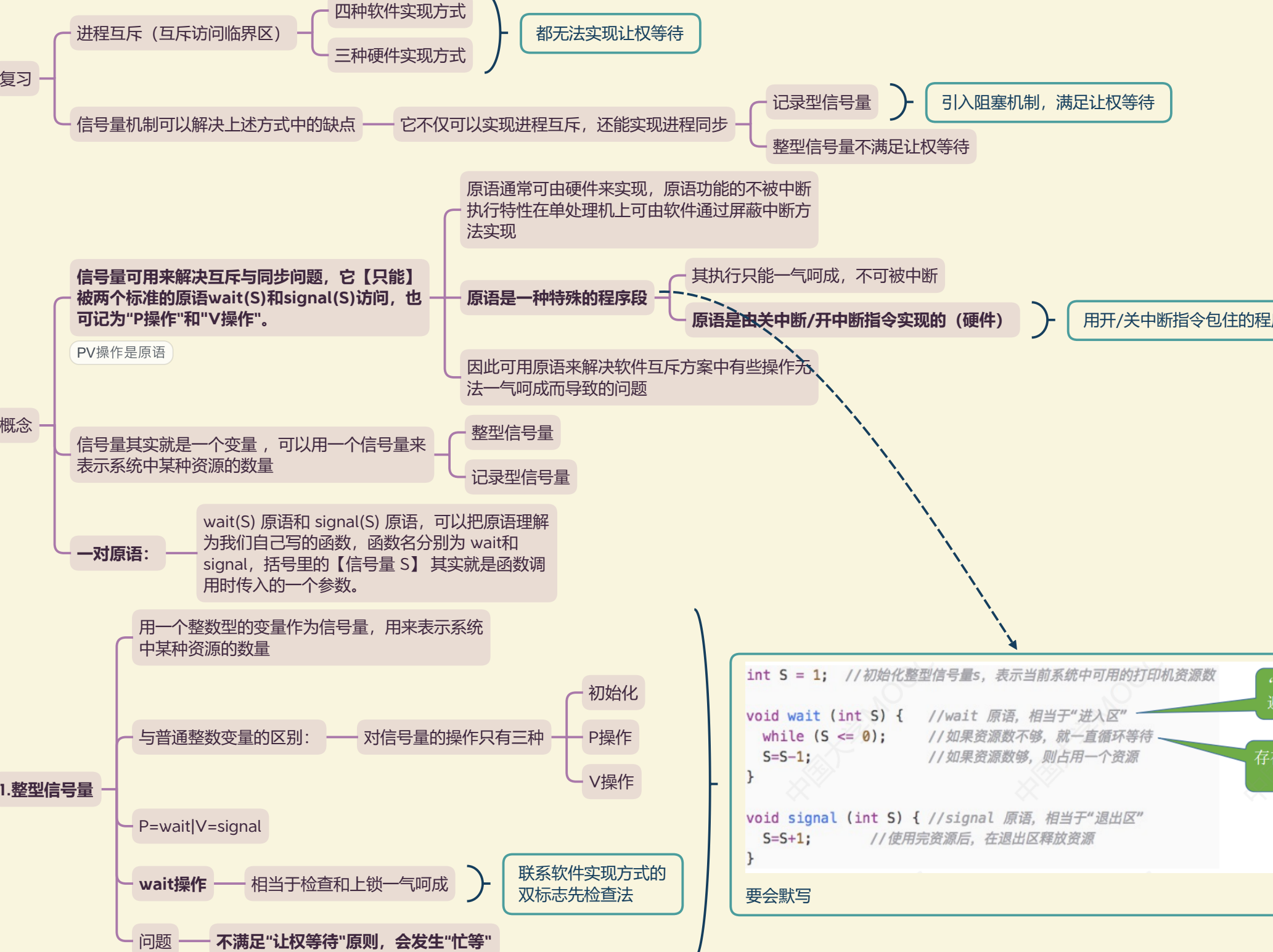
让权等待优先最低——好多算法都没有实现让权等待

互斥锁



```
acquire()
while(!available)
    ; //忙等待
available = false; //获得锁
}
release() {
    available = true; //释放锁
}
```

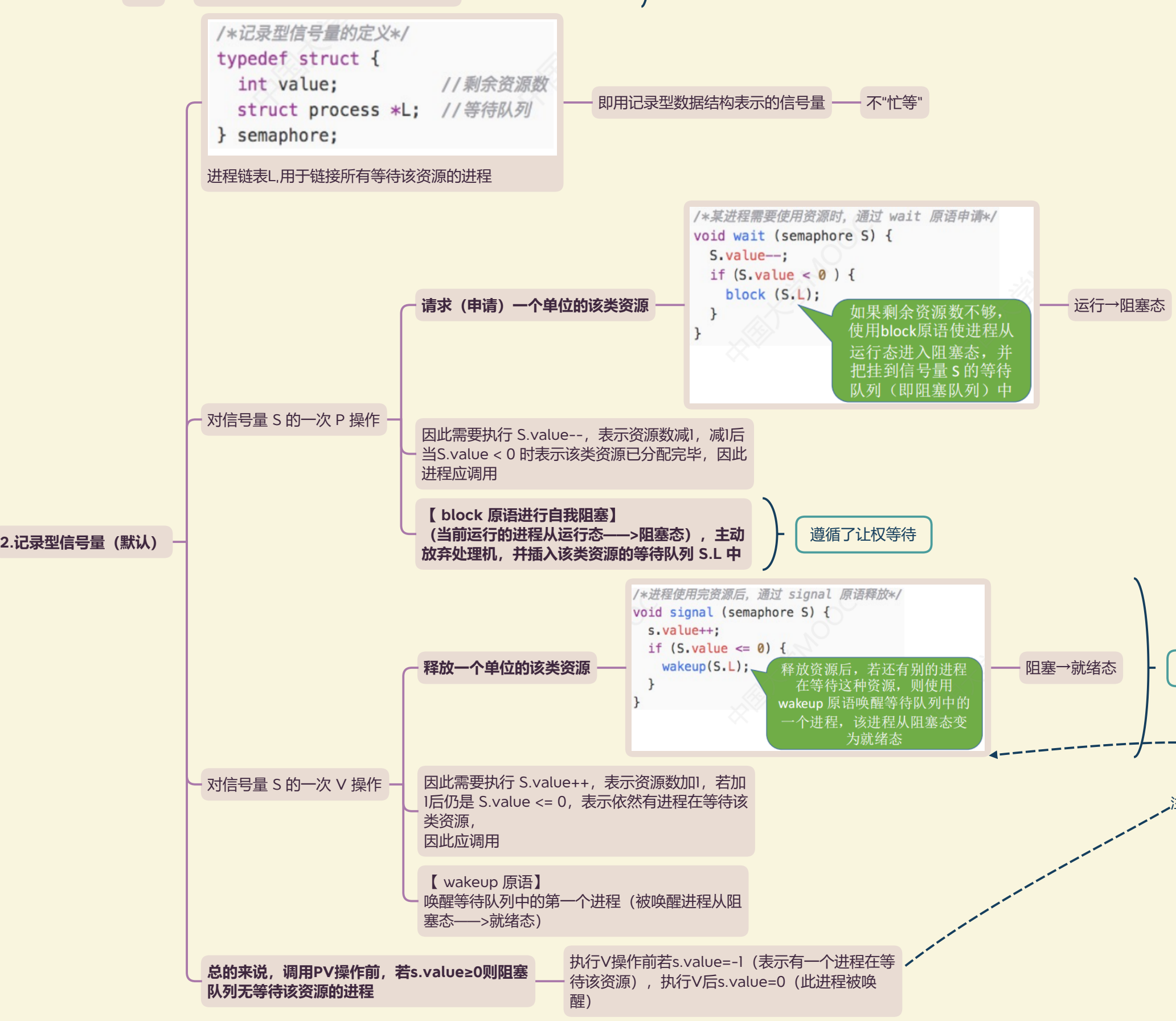
信号量



```
int S = 1; //初始化整型信号量S，表示当前系统中可用的打印机资源数
void wait (int S) { //wait 原语，相当于“进入区”
    while (S <= 0) { //如果资源数不等，就一直循环等待
        S=S-1; //如果资源数不够，则占用一个资源
    }
}
void signal (int S) { //signal 原语，相当于“退出区”
    S=S+1; //使用完资源后，在退出区释放资源
}
```

“检查”和“上锁”一气呵成，避免了并发、异步导致的问题

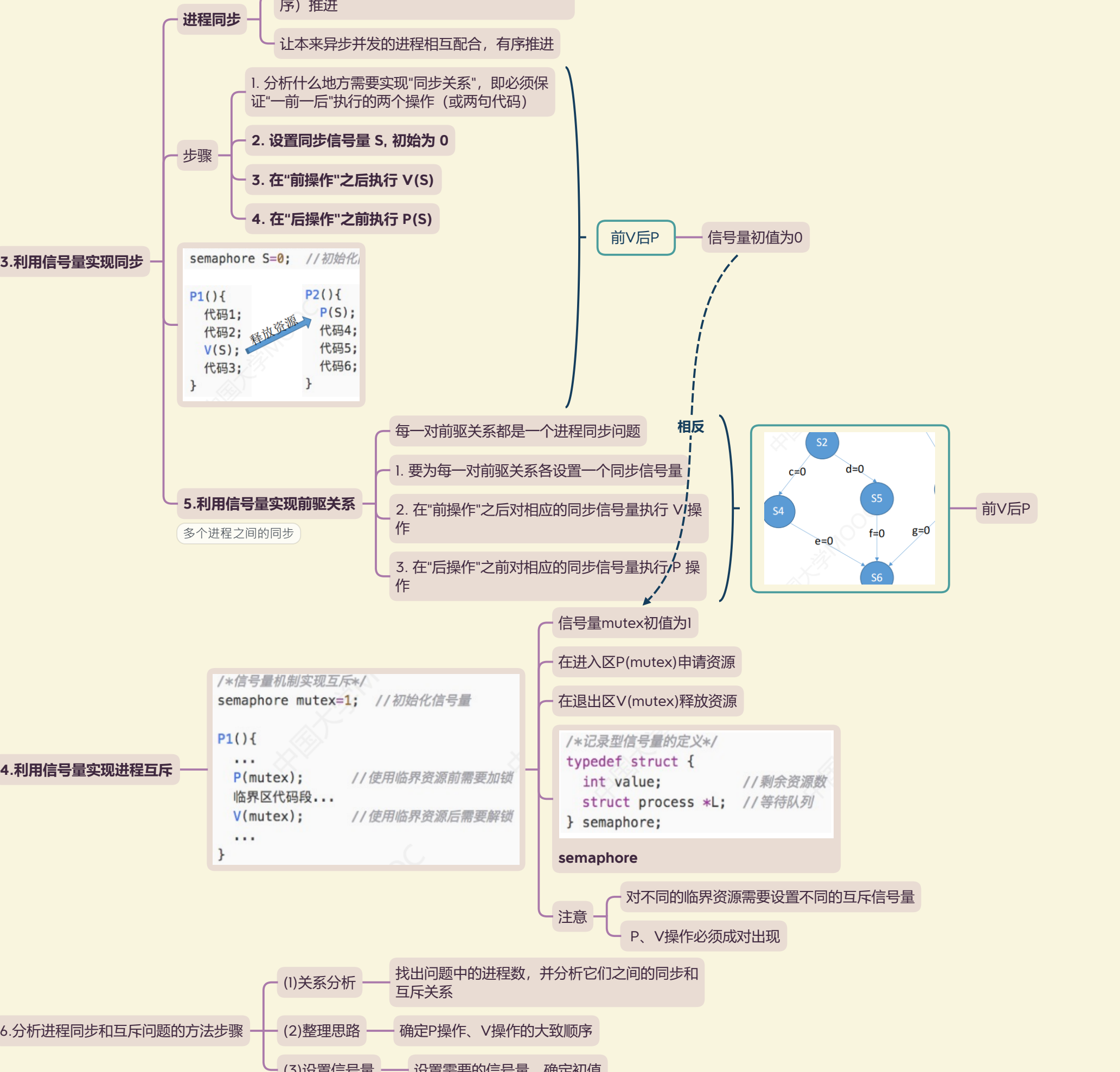
存在的问题：不满足“让权等待”原则，会发生“忙等”



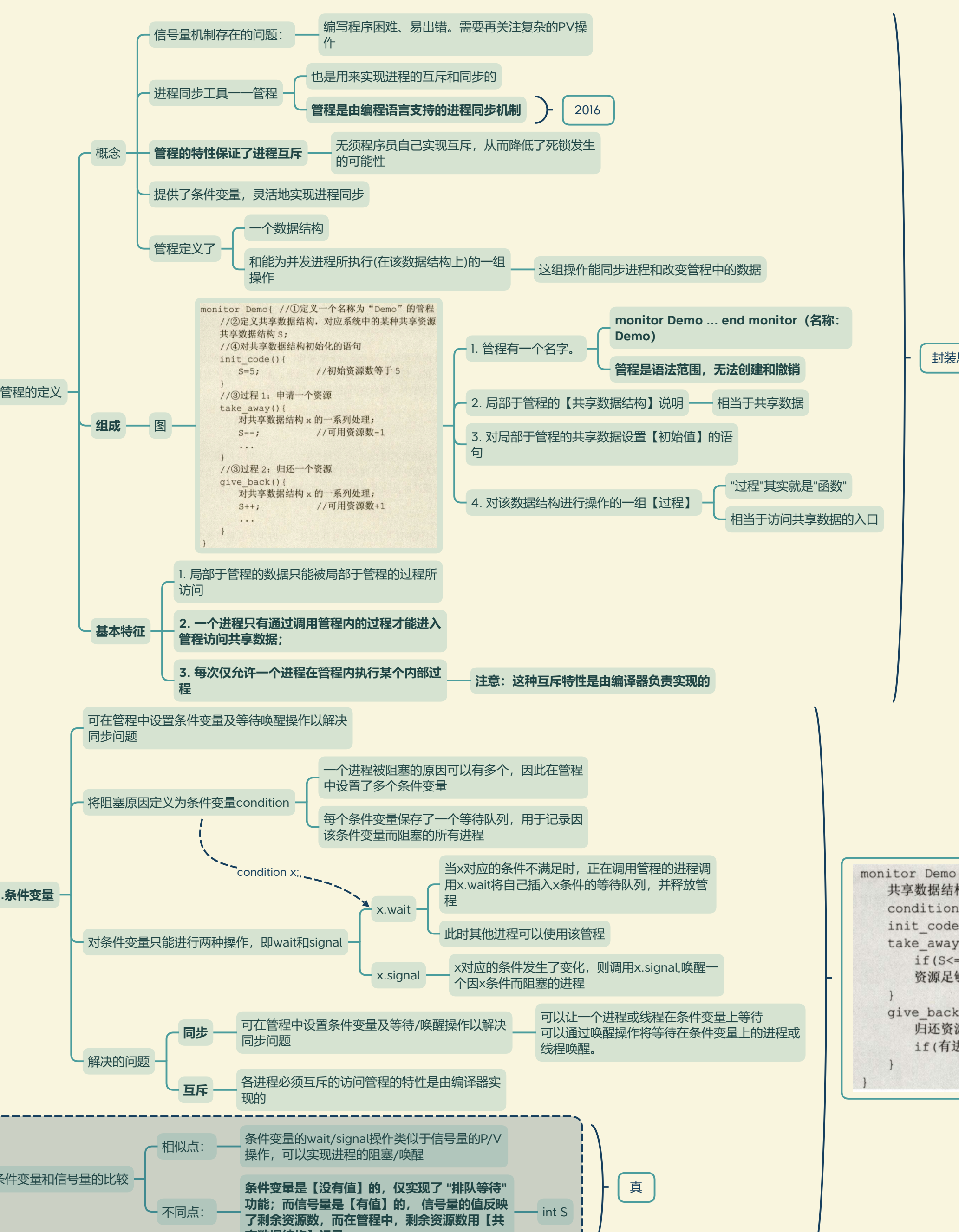
考试中出现 P(S)、V(S) 的操作，除非特别说明，否则默认 S 为记录型信号量

同步与互斥

信号量



管程



```
monitor Demo{
    共享数据结构 S;
    condition x; //定义一个条件变量 x
    init_code() { ... }
    take_away() { if (S<0) x.wait(); //资源不够，在条件变量 x 上阻塞等待资源足够，分配资源，做一系列相应处理; }
    give_back() { 归还资源，做一系列相应处理; if (有进程在等待) x.signal; //唤醒一个阻塞进程 }
}
```