第四章作业

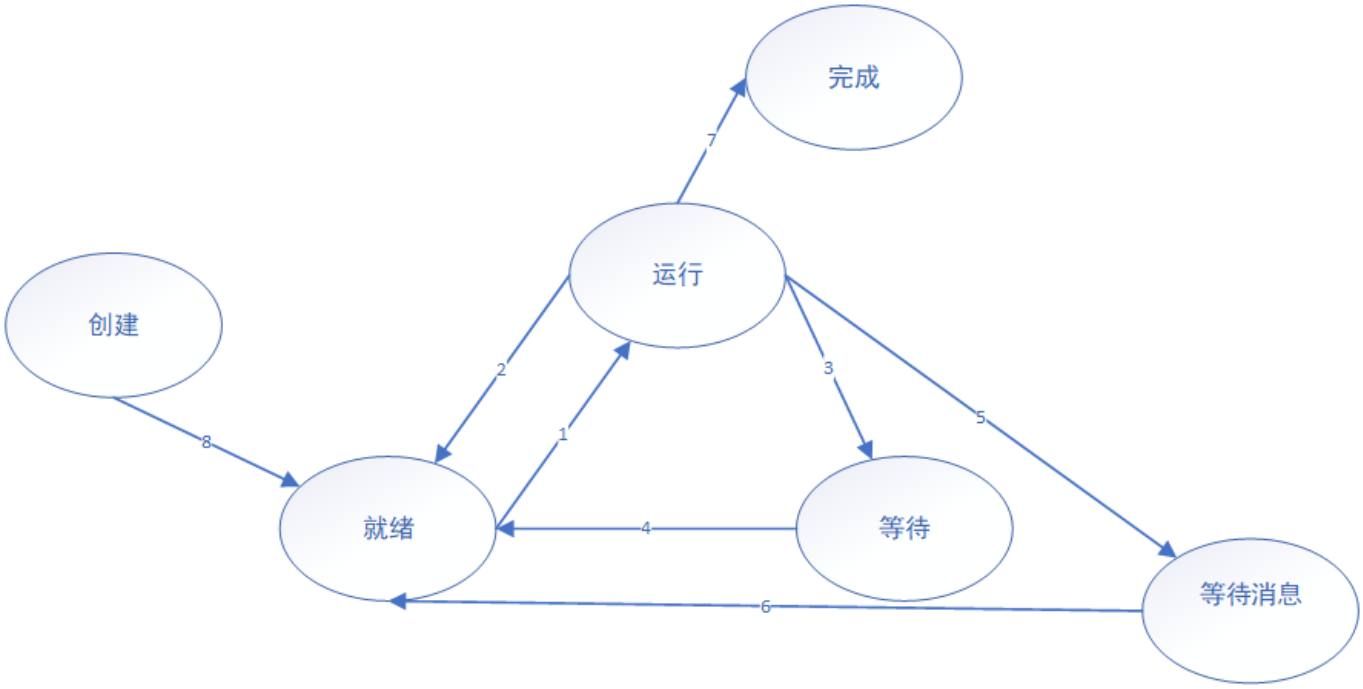
4-3

- 进程是程序在处理机上的一次执行过程
- 与程序的区别：
  - 程序是指令的有序集合，是一个静态概念，其本身没有任何运行的含义。而进程是动态的概念。程序可以作为一种软件资料长期保存，而进程则是有一定生命周期的，它能够动态的产生和消亡
  - 进程是一个能独立运行的单位，能与其他进程并行地活动
  - 进程是竞争计算机系统有限资源的基本单位，也是进行处理机调度的基本单位

4-9

- (1)
  - 变迁2：分配给进程的时间片执行完成或者有更高优先度的进程到达
  - 变迁3：等待事件。如向操作系统请求共享资源失败，等待某种操作完成、新数据尚未到达（I/O操作）、等待新任务的到达
  - 变迁4：等待事件发生
- (2)
  - 就绪队列非空
- (3)
  - a 一定会发生
  - b 不可能发生
  - c 可能发生 就绪队列为空或者转为就绪态的进程优先度更高

4-10



1. cpu空闲，进程调度
2. 时间片到
3. 等待事件
4. 等待事件发生
5. 等待消息
6. 接受到消息
7. 任务完成
8. 进程创建

4-12

- 程序描述

```
main(){
        int mutex=1;//Q互斥信号灯
        cobegin
        P1;P2;...Pn;
        coend
}
Pi(){
        P(mutex);
        使用Q;
        V(mutex);
}
```

- 信号灯取值范围$[-(n-1),1]$

- mutex=1 :没有进程
- mutex=0 :一个进程
- mutex=-(n-1) :有一个进程进入临界段执行，且有（n-1）个进程正在等待进入

4-13

- (a)

```
main(){
        int s2=0;s3=0;s4=0;
        cobegin
                P1;P2;P3;P4;
        coend
}
P1(){
        ...
        V(s2);
        V(s3);
        V(s4);
}
P2(){
        P(s2);
        ...
}
P3(){
        P(s3);
        ...
}
P4(){
        P(s4);
        ...

}
```

- (b)

```
main(){
        int s13=0,s23=0;
        cobegin
                P1;P2;P3;
        coend
}
P1(){
        ...
```

```
        V(s13);
}
P2(){
        ...
        V(s23);
}
P3(){
        P(s13);
        P(s23);
        ...
}
```

4-16

- **互斥**：在操作系统中，当某一进程正在访问某一存储区域时，就不允许其他进程读出或者修改该存储区的内容，否则，就会发生后果无法估计的错误。进程之间的这种相互制约关系称为互斥。
- **同步**：并发进程在一些关键点上可能需要互相等待与互通消息，这种相互制约的等待与互通信息称为进程同步
- **联系**：这两个概念都属于同步范畴，描述并发进程相互之间的制约关系，互斥是同步的一种特例
- **区别**：同步是指并发进程按照他们之间的约束关系，在执行的先后次序上必须满足这种约束关系，而互斥是指并发进程按照他们之间的约束关系，在某一点上一个时刻只允许一个进程执行，一个进程做完了，另一个进程才能执行，而不管谁先做这个操作。

4-22

- 线程是进程的一个执行路径
- **区别**：
    1. 线程是进程的一个组成部分，一个进程可以有多个线程，而且至少有一个可执行的线程
    2. 进程是资源分配的基本单位，它拥有自己的地址空间和各种资源；线程是处理机调度的基本单位，它只能和其他线程共享进程的资源，而本身并没有任何资源。
    3. 进程切换时，消耗的资源大，效率高。所以涉及到频繁的切换时，使用线程要好于进程。
    4. 多进程要比多线程健壮，一个进程崩溃后，在保护模式下不会对其他进程产生影响，但是一个线程崩溃整个进程都死掉
    5. 每个独立的进程有一个程序运行的入口、顺序执行序列和程序出口，执行开销大。但是线程不能独立执行，必须依存在应用程序中，由应用程序提供多个线程执行控制，执行开销小。

4-29

- 代码描述：

```
main(){
        int seats=5;
        int mutex=0;
        cobegin
                P1;P2;P3;...Pn;
        coend
}
Pi{
        P(mutex);
        if(count==0){
                V(mutex);
                return;.
        }
        seats--;//坐下
        V(mutex);
        ...//休息
        P(mutex);
        seats++;//离开
        V(mutex);
}
```