**实 验 报 告**

**课程名称** 操作系统

**实验项目** 进程管理

**实验仪器** 计算机

**系 别** 计算机科学与技术

**专 业**  计算机科学与技术

**班级/学号** 计科1606/2016012620

**学生姓名**  闫明

**实验日期**  19-1-1

**成 绩 \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**指导教师**  牛欣源

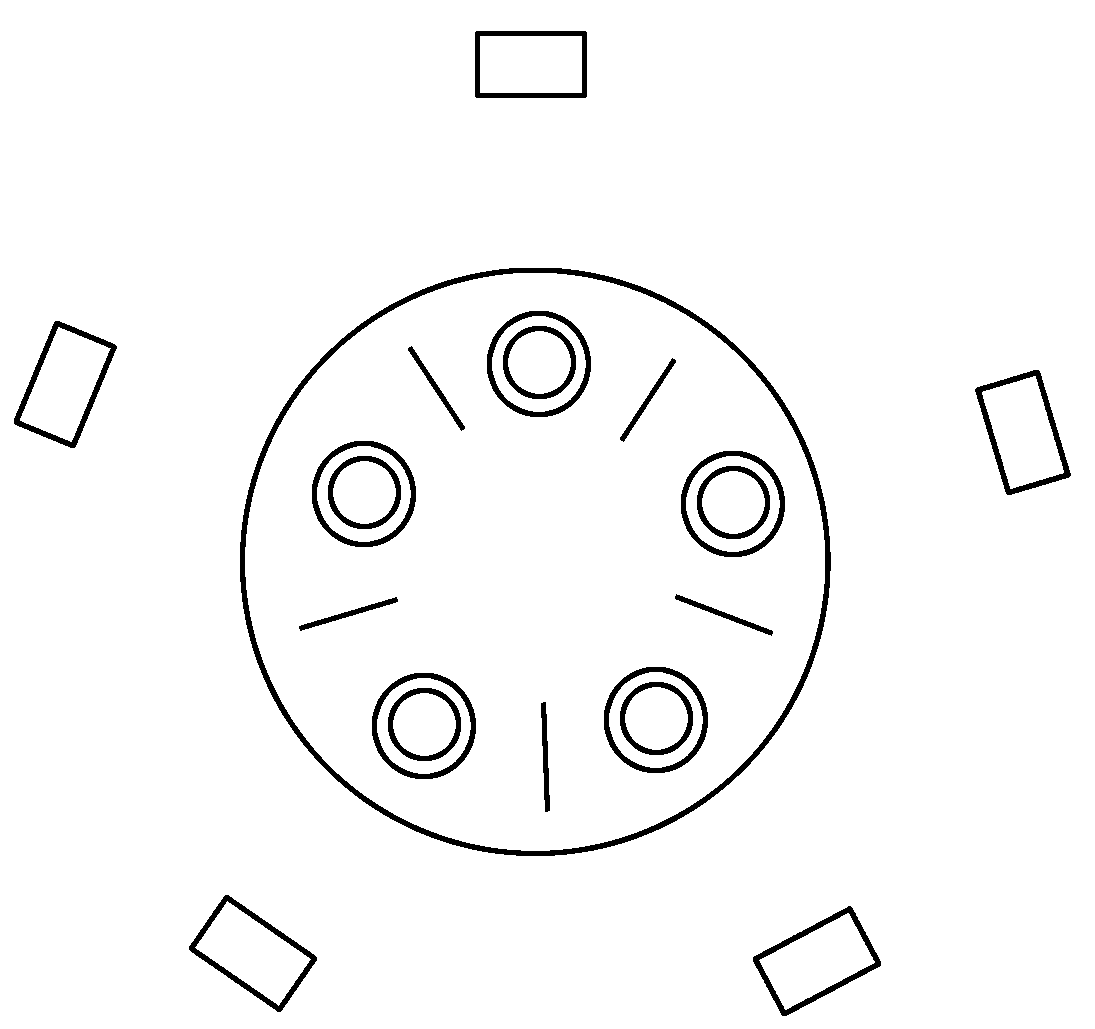
1. **实验目的**
2. 掌握进程的概念，理解进程创建过程，能够使用系统调用完成进程创建，形成多个进程并发的环境。
3. 掌握进程控制方法。通过调用相关的系统调用，对进程实验睡眠、同步、撤销等控制。
4. 进一步认识并发执行的实质，分析进程竞争资源的现象，学习避免死锁的方法。
5. **实验环境**

操作系统：Linux操作系统，Ubuntu 18.04.1 LTS  
应用系统：gcc编译器  
应用软件：vim

1. **实验内容**

哲学家问题。

问题描述为：五位哲学家围坐在一个圆桌周围，每人面前有一只碗，各碗之间分别有一根筷子，餐桌如下图：



哲学家有两种活动：吃饭和思考。当哲学家饥饿时，就分两次去取左右的筷子，每次拿一根（不抢邻座的），如果成功，就吃饭，吃完后吧筷子放回原处继续思考

1. **设计和实现**

**数据结构**：

封装Linux操作系统提供的信号量接口，封装后，有五个主要操作：信号量创建、信号量赋值、P操作、V操作，信号量删除。

**设计：**

采用最多允许四个哲学家进屋同时就餐的算法，以保证至少一个哲学家能够进餐，并且总会释放他使用的两根筷子，确保更多的哲学家进餐。选择用room当做信号量，只允许四个哲学家进入餐厅，而申请进入餐厅的哲学家进入room等待队列，根据FIFO原则，总会进入餐厅就餐，不会发生饥饿和饿死。

**关键代码：**

void philosopher(int i){//哲学家进屋

    srand((int)time(0));

    while(1){

        for(int a=0;a<i;a++){

            printf("/t/t/t/t/t");

            printf("Philosopher %d entered the room.\n", i);

            Psem(room);//申请进屋

            eat(i);//吃

            Vsem(room);//出去（释放资源）

        }

        for(int a=0;a<i;a++){

            printf("/t/t/t/t/t");

            printf("Philosopher %d left the room.\n", i);

        }

    }

}

1. **运行结果**

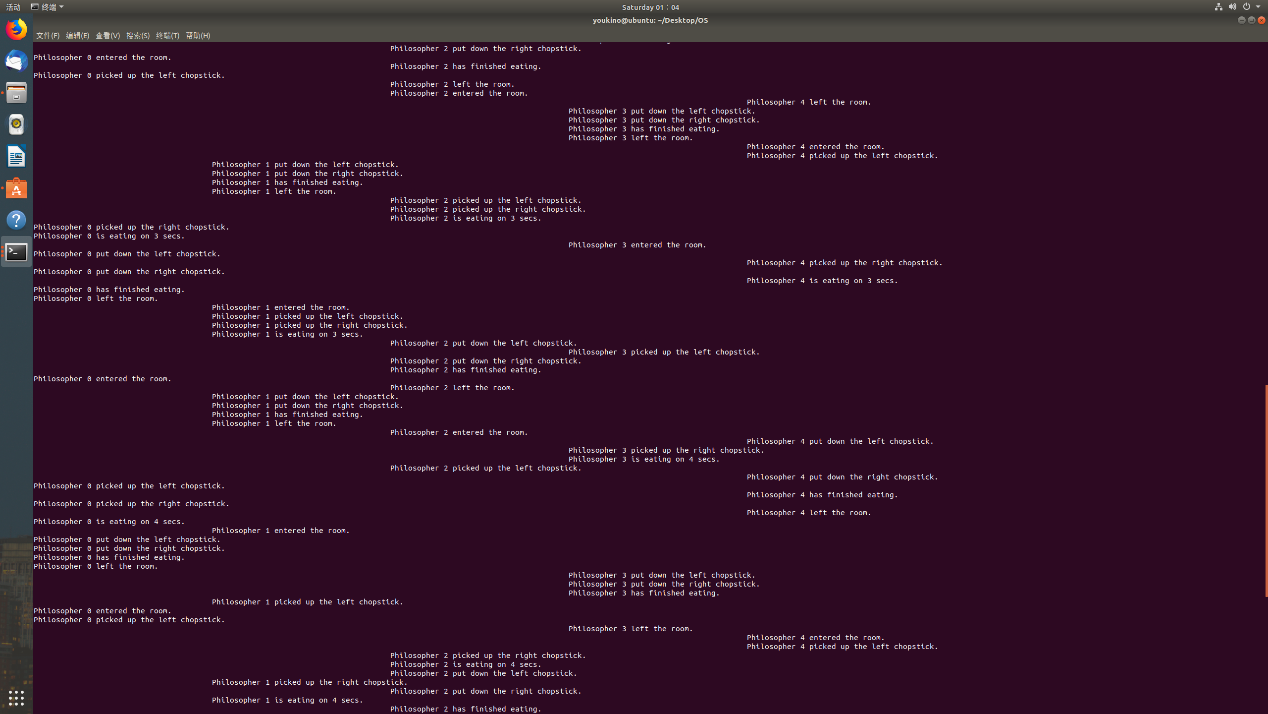


图1 哲学家进餐问题部分输出（使用control+滚轮可以缩放）

该输出显示了某一时刻系统中五个模拟哲学家的五个进程的相互动作，他们只有四个进程（哲学家）可以进入屋子（room设置初值为4），进入屋子后尝试拿筷子，拿起两只筷子后吃饭，吃完饭就放下筷子，离开屋子，以便第五个进程可以进入屋子，没拿到筷子的进程在等待队列中。可以看出五个子进程井然有序，相互交替吃饭出屋，父进程把子进程的结果输出到屏幕上。

1. **总结**

**实验中遇到的问题：**

1. 老师要求将所有进程的实时状态输出

起初并不是非常理解这个要求，也不太明白如何实现，后来经过询问同学，得知只要用一个for循环，然后输出五个/t即可，成功之后明白了老师要求的意思，按列输出五个进程的状态会使该问题非常直观，你可以直观的看到每个进程什么时刻处于一种什么状态，理解这五个进程的并行程度。真正的了解并行执行和进程到底在这个程序到底扮演了怎么样的角色。

**实验收获：**

1. **更深入的了解了进程的并行执行过程，以及进程的概念，在检查实验的过程中，发现其实自己并不是非常了解这样的状态，没有抓住系统中实际上有五个进程在并发这个要点，还是抱着直线型单线程的思维在思考这个问题。**
2. **对哲学家算法有了更深刻的了解，知道算法和实际实现了算法是截然不同的。**
3. **对如何在Linux操作系统中，封装PV操作有了一定的了解。**
4. **问题**
5. **分析哲学家进餐问题，什么时候会死锁？**

**死锁产生有四个条件，互斥条件，占有且等待条件，不可抢占条件，循环申请条件，只要满足这四个条件就会死锁。**

**所以以这四点来分析**

1. **互斥条件：**

**当哲学家拿到筷子后，就绝不放下，其他人需要等待**

1. **占有且等待条件：**

**当哲学家手里有筷子时，就去拿另一边的筷子，若没有，就等待**

1. **不可抢占条件：**

**所有哲学家都不抢别人手里的筷子**

1. **循环等待条件：**

**所有哲学家都等待别的哲学放下筷子就拿起来**

**这四点同时满足，就会死锁**

1. **哲学家问题其他算法**
2. **奇数号哲学家先取左手边筷子，偶数号哲学家取右手边筷子**
3. **当哲学家能拿起两边的筷子时，才拿筷子吃饭**