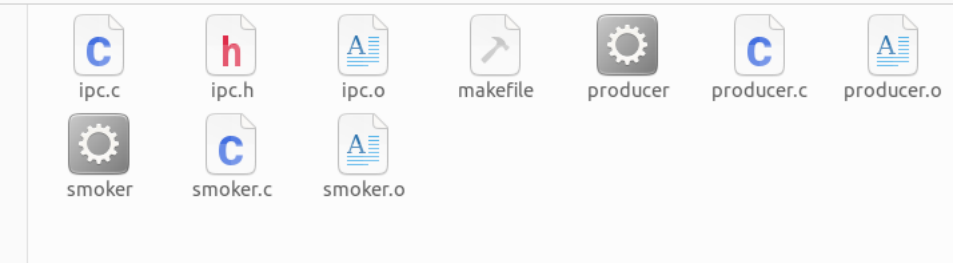
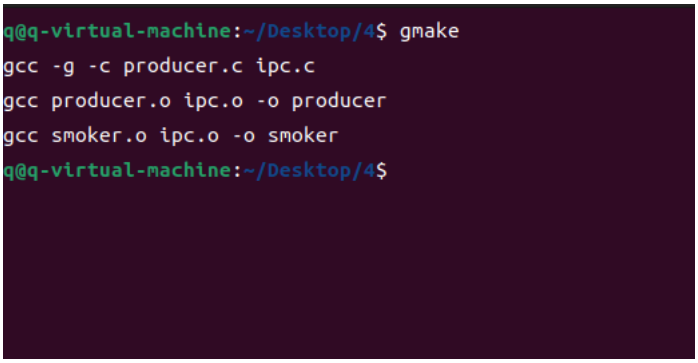
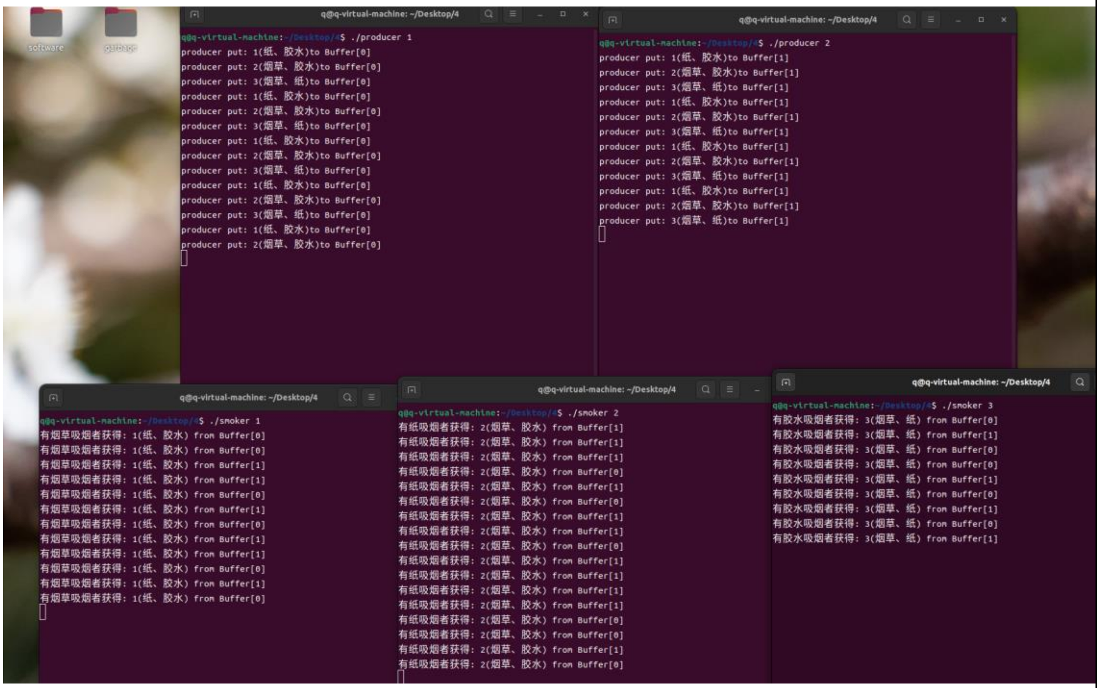
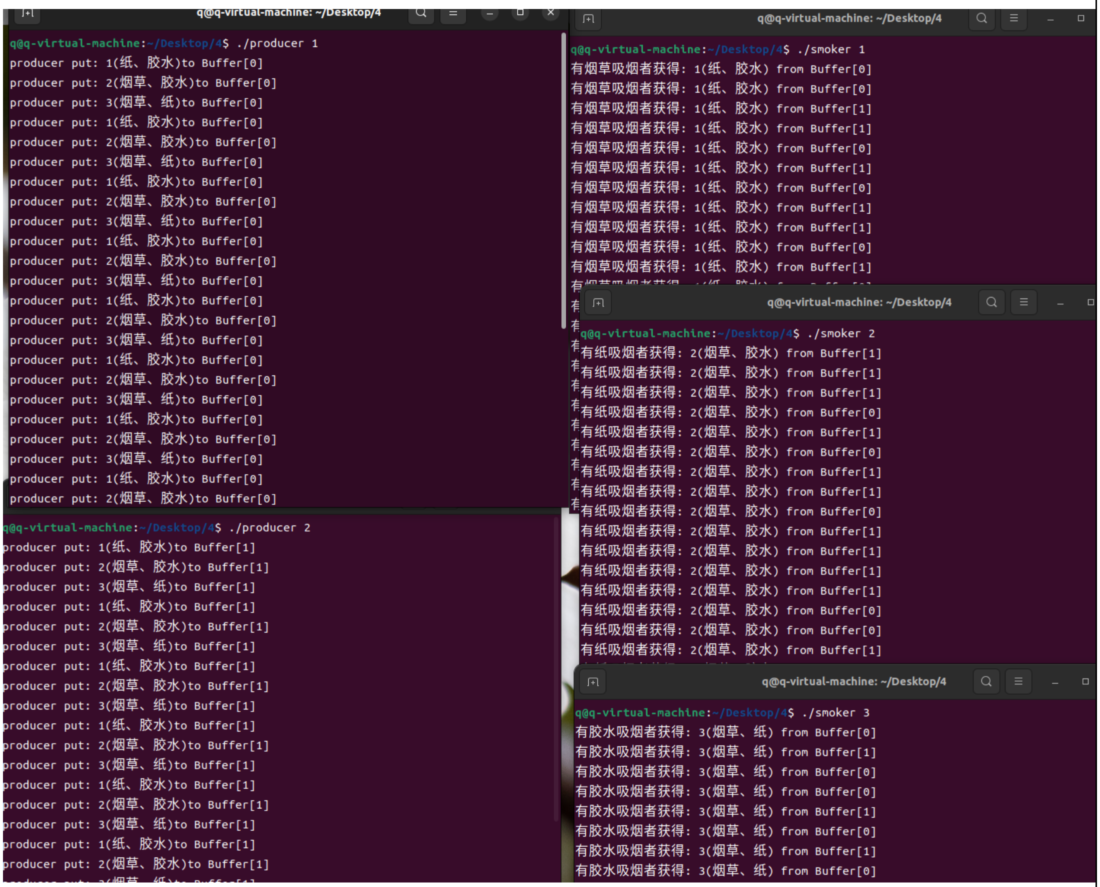
山东大学 计算机科学与技术 学院
操作系统 课程实验报告

学号：202200130223	姓名：李洋永	班级：4 班
实验题目：实验 4 进程同步		
实验学时：2	实验日期：2024.10.30	
实验目的：加深对并发协作进程同步与互斥概念的理解，观察和体验并发进程同步与互斥操作的效果，分析与研究经典进程同步与互斥问题的实际解决方案。了解 Linux 系统中 IPC 进程同步工具的用法，练习并发协作进程的同步与互斥操作的编程与调试技术。		
实验环境： Ubuntu		
源程序清单：  		
编译及运行结果：  		




q@q-virtual-machine: ~/Desktop/4
q@q-virtual-machine:~/Desktop/4$ ./producer 1
producer put: 1(纸、胶水)to Buffer[0]
producer put: 2(烟草、胶水)to Buffer[0]
producer put: 3(烟草、纸)to Buffer[0]
producer put: 1(纸、胶水)to Buffer[0]
producer put: 2(烟草、胶水)to Buffer[0]
producer put: 3(烟草、纸)to Buffer[0]
producer put: 1(纸、胶水)to Buffer[0]
producer put: 2(烟草、胶水)to Buffer[0]
producer put: 3(烟草、纸)to Buffer[0]
producer put: 1(纸、胶水)to Buffer[0]
producer put: 2(烟草、胶水)to Buffer[0]
producer put: 3(烟草、纸)to Buffer[0]
producer put: 1(纸、胶水)to Buffer[0]
producer put: 2(烟草、胶水)to Buffer[0]
producer put: 3(烟草、纸)to Buffer[0]
producer put: 1(纸、胶水)to Buffer[0]
producer put: 2(烟草、胶水)to Buffer[0]
producer put: 3(烟草、纸)to Buffer[0]
producer put: 1(纸、胶水)to Buffer[0]
producer put: 2(烟草、胶水)to Buffer[0]
producer put: 3(烟草、纸)to Buffer[0]
producer put: 1(纸、胶水)to Buffer[0]
producer put: 2(烟草、胶水)to Buffer[0]
q@q-virtual-machine:~/Desktop/4$ ./producer 2
producer put: 1(纸、胶水)to Buffer[1]
producer put: 2(烟草、胶水)to Buffer[1]
producer put: 3(烟草、纸)to Buffer[1]
producer put: 1(纸、胶水)to Buffer[1]
producer put: 2(烟草、胶水)to Buffer[1]
producer put: 3(烟草、纸)to Buffer[1]
producer put: 1(纸、胶水)to Buffer[1]
producer put: 2(烟草、胶水)to Buffer[1]
producer put: 3(烟草、纸)to Buffer[1]
producer put: 1(纸、胶水)to Buffer[1]
producer put: 2(烟草、胶水)to Buffer[1]
producer put: 3(烟草、纸)to Buffer[1]
producer put: 1(纸、胶水)to Buffer[1]
producer put: 2(烟草、胶水)to Buffer[1]
producer put: 3(烟草、纸)to Buffer[1]
producer put: 1(纸、胶水)to Buffer[1]
producer put: 2(烟草、胶水)to Buffer[1]
q@q-virtual-machine: ~/Desktop/4
q@q-virtual-machine:~/Desktop/4$ ./smoker 1
有烟草吸烟者获得: 1(纸、胶水) from Buffer[0]
有烟草吸烟者获得: 1(纸、胶水) from Buffer[0]
有烟草吸烟者获得: 1(纸、胶水) from Buffer[1]
有烟草吸烟者获得: 1(纸、胶水) from Buffer[1]
有烟草吸烟者获得: 1(纸、胶水) from Buffer[0]
有烟草吸烟者获得: 1(纸、胶水) from Buffer[1]
有烟草吸烟者获得: 1(纸、胶水) from Buffer[0]
有烟草吸烟者获得: 1(纸、胶水) from Buffer[1]
有烟草吸烟者获得: 1(纸、胶水) from Buffer[1]
有烟草吸烟者获得: 1(纸、胶水) from Buffer[0]
有烟草吸烟者获得: 1(纸、胶水) from Buffer[1]
q@q-virtual-machine: ~/Desktop/4
q@q-virtual-machine:~/Desktop/4$ ./smoker 2
有纸吸烟者获得: 2(烟草、胶水) from Buffer[1]
有纸吸烟者获得: 2(烟草、胶水) from Buffer[1]
有纸吸烟者获得: 2(烟草、胶水) from Buffer[1]
有纸吸烟者获得: 2(烟草、胶水) from Buffer[0]
有纸吸烟者获得: 2(烟草、胶水) from Buffer[1]
有纸吸烟者获得: 2(烟草、胶水) from Buffer[0]
有纸吸烟者获得: 2(烟草、胶水) from Buffer[1]
有纸吸烟者获得: 2(烟草、胶水) from Buffer[1]
有纸吸烟者获得: 2(烟草、胶水) from Buffer[0]
有纸吸烟者获得: 2(烟草、胶水) from Buffer[1]
有纸吸烟者获得: 2(烟草、胶水) from Buffer[1]
有纸吸烟者获得: 2(烟草、胶水) from Buffer[1]
有纸吸烟者获得: 2(烟草、胶水) from Buffer[1]
有纸吸烟者获得: 2(烟草、胶水) from Buffer[0]
有纸吸烟者获得: 2(烟草、胶水) from Buffer[0]
有纸吸烟者获得: 2(烟草、胶水) from Buffer[1]
q@q-virtual-machine: ~/Desktop/4
q@q-virtual-machine:~/Desktop/4$ ./smoker 3
有胶水吸烟者获得: 3(烟草、纸) from Buffer[0]
有胶水吸烟者获得: 3(烟草、纸) from Buffer[1]
有胶水吸烟者获得: 3(烟草、纸) from Buffer[0]
有胶水吸烟者获得: 3(烟草、纸) from Buffer[0]
有胶水吸烟者获得: 3(烟草、纸) from Buffer[1]
有胶水吸烟者获得: 3(烟草、纸) from Buffer[0]
有胶水吸烟者获得: 3(烟草、纸) from Buffer[1]
有胶水吸烟者获得: 3(烟草、纸) from Buffer[0]

```
    for (int i = 0; i < buff_num; i++) buff_ptr[i] = 0;
    *pput_ptr =0;
    int ra=-1;
    while (1) {
            if(type1==1){
                    down(prod_sem1);
            }
            else if(type1==2){
                    down(prod_sem2);
            }

            down(pmtx_sem);
            ra++;
            ra = ra% 3;
            sleep(rate);
            if(type1==1&&buff_ptr[0]==0){
                    switch(ra+1){
                    case 1:
                    printf("producer put: 1(纸、胶水)to Buffer[0]\n");
                    break;
                    case 2:
                    printf("producer put: 2(烟草、胶水)to Buffer[0]\n");
                    break;
                    case 3:
                    printf("producer put: 3(烟草、纸)to Buffer[0]\n");
                    break;
                    }
                    buff_ptr[0] = ra + 1;

            }

            if(type1==2&&buff_ptr[1]==0){
                    switch(ra+1){
                    case 1:
                    printf("producer put: 1(纸、胶水)to Buffer[1]\n");
                    break;
                    case 2:
                    printf("producer put: 2(烟草、胶水)to Buffer[1]\n");
                    break;
                    case 3:
                    printf("producer put: 3(烟草、纸)to Buffer[1]\n");
                    break;
                    }
                    buff_ptr[1] = ra + 1;

            }
            up(pmtx_sem);
            if (ra == 0) up(sm1_sem);
            else if (ra == 1) up(sm2_sem);
            else if (ra == 2) up(sm3_sem);


    }
    return EXIT_SUCCESS;
```

根据输入的 type1 指定是哪一个生产者，生产者 1、2 分别往 Buffer[0]、Buffer[1]放入数据，每三个一循环，根据放入的数据唤醒对应的吸烟者。

```
*cget_ptr=0;
while (1) {
        if (type == 1) {
                down(sm1_sem);
                while (buff_ptr[*cget_ptr] != 1) {
                        *cget_ptr = (*cget_ptr + 1) % buff_num;
                }
                buff_ptr[*cget_ptr] = 0;

                printf("有烟草吸烟者获得：1(纸、胶水) from Buffer[%d]\n",*cget_ptr);
        }
        else if (type == 2) {
                down(sm2_sem);
                while (buff_ptr[*cget_ptr] != 2) {
                        *cget_ptr = (*cget_ptr + 1) % buff_num;
                }
                buff_ptr[*cget_ptr] = 0;
                printf("有纸吸烟者获得：2(烟草、胶水) from Buffer[%d]\n",*cget_ptr);
        }
        else if (type == 3) {
                down(sm3_sem);
                while (buff_ptr[*cget_ptr] != 3) {
                        *cget_ptr = (*cget_ptr + 1) % buff_num;
                }
                buff_ptr[*cget_ptr] = 0;
                printf("有胶水吸烟者获得：3(烟草、纸) from Buffer[%d]\n",*cget_ptr);
        }
        if(*cget_ptr==0){
                up(prod_sem1);
        }
        if(*cget_ptr==1){
                up(prod_sem2);
        }
        *cget_ptr = (*cget_ptr + 1) % buff_num;

}

return EXIT_SUCCESS;
```

吸烟者一直查看两个缓冲区中是否有自己需要的材料，如果有则清空缓冲区且唤醒该缓冲区对应的生产者。

问题及收获:

真实操作系统中通过信号量、互斥锁、条件变量来解决同步问题，信号量的值表示可以资源的数量或状态，互斥锁用于保护共享资源，确保一次只有一个进程访问共享资源，条件变量结合互斥锁实现特定条件的同步。信号量可以用于保护一个临界区，只有一个进程进入，执行 P 操作时进入临界区，其它进程阻塞，执行 V 操作时离开临界区，可以保证同一时间只有一个进程更改共享资源，实现互斥，系统可通过信号量的增减控制资源的访问，实现同步。信号量 empty 表示缓冲区的空余空间数量，生产者每生产一个项目就减少一个空余空间，消费者消费后空余空间增加，full 表示缓冲区中的项目数，消费者每消费一个项目就减少一个项目数，生产者生产后项目数增加。