```
/* 实验 8 参考代码 */
/* 2sem_producer.c */
/* 【方法 1】2 个信号量，1 个负责统计仓库中已有资源的数量，1 个负责统计仓库中剩余空间的数量，
文件锁负责互斥 */

#include <sys/types.h>
#include <unistd.h>
#include <fcntl.h>
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <sys/ipc.h>
#include <sys/sem.h>
#include "sem_com.c"
#include "lock_set.c"
#define store_file "/tmp/store"
#define MAXLEN 10

int produce()
{
int store_fd,sem_key,sem_id;
unsigned int s_start,s_count,s_size;
static unsigned int counter=0;
char buff[MAXLEN];
if((store_fd=open(store_file,O_CREAT|O_WRONLY|O_APPEND,0666))<0)
{
printf("Open error!\n");
exit(1);
}
s_start='0';
s_count=10;
sprintf(buff,"%c",(s_start+counter));
counter=(counter+1)%s_count;
lock_set(store_fd,F_WRLCK);
if((s_size=write(store_fd,buff,strlen(buff)))<0)
{
printf("Producer: write error!\n");
return -1;
}
lock_set(store_fd,F_UNLCK);
printf("Produced '%c'",buff[0]);
```

```
close(store_fd);
return 0;
}

int main()
{
key_t sem_key;
int sem_id;

int x;

sem_key = ftok("/",'x');
sem_id = semget(sem_key,2,0777|IPC_CREAT);

if((x=semctl(sem_id,0,GETVAL,NULL))<=0)

{
sem_init(sem_id,0,0); /*初始化已生产的资源，初值为 0*/
sem_init(sem_id,1,100); /*初始化仓库空余位置，初值为 100，最多生产 100 个资源*/
}
printf("!!!!!semctl = %d\n",x);
printf("This is producer!\n");
while(1)
{
if((semctl(sem_id,1,GETVAL,NULL))>0)
{
printf("============================\n");
printf("Before produce:\n");
printf("Produced %d, Free %d\n",semctl(sem_id,0,GETVAL,NULL),semctl(sem_id,1,GETVAL,NULL));
printf("\nNow producing...\n");
sem_p(sem_id,1);/*仓库空位减少 1 个*/
sem_v(sem_id,0);/*生产者生产 1 个资源，仓库中资源数量增加 1 个*/
produce();
printf("\n\nAfter produced:\n");
printf("Produced %d, Free
%d\n\n",semctl(sem_id,0,GETVAL,NULL),semctl(sem_id,1,GETVAL,NULL));
}
else
{
printf("============================\n");
printf("Full!\n\n");
```

```
}
sleep(1);
}
}
```

```
/* 实验 8 参考代码 */
/* 2sem_customer.c */
/* 【方法 1】2 个信号量，1 个负责统计仓库中已有资源的数量，1 个负责统计仓库中剩余空>间的数量，
文件锁负责互斥 */

#include <sys/types.h>
#include <unistd.h>
#include <stdio.h>
#include <stdlib.h>
#include <sys/ipc.h>
#include <string.h>
#include <fcntl.h>
#include <sys/sem.h>
#include "sem_com.c"
#include "lock_set.c"
#define store_file "/tmp/store"
#define tmp_file "/tmp/tmp"
#define MAX_FILE_SIZE 100*1024


int myfilecopy(const char *sour_file, const char * dest_file, int offset, int count)
{
int in_file,out_file;
char buff;
if((in_file=open(sour_file,O_RDONLY|O_NONBLOCK))<0)
{
printf("Sour_file error!\n");
return -1;
}
if((out_file=open(dest_file,O_RDWR|O_CREAT|O_TRUNC|O_NONBLOCK,0666))<0)
{
printf("Dest_file error!\n");
return -1;
}
lseek(in_file,offset,SEEK_SET);
while((read(in_file,&buff,1)) == 1)
{
write(out_file,&buff,1);
}
close(in_file);
close(out_file);
```

```
return 0;
}

int customing()
{
int store_fd;
char buff;
if((store_fd=open(store_file,O_RDWR))<0)
{
printf("Customing error!\n");
return -1;
}
printf("Enjoy: ");
lseek(store_fd,0,SEEK_SET);
if((read(store_fd,&buff,1)) == 1)
{
fputc(buff,stdout);
printf("\n");
}
lock_set(store_fd,F_WRLCK);
myfilecopy(store_file,tmp_file,1,MAX_FILE_SIZE);
myfilecopy(tmp_file,store_file,0,MAX_FILE_SIZE);
lock_set(store_fd,F_UNLCK);
unlink(tmp_file);
close(store_fd);
return 0;
}

int main()
{
int sem_key,sem_id;
sem_key = ftok("/",'x');
sem_id = semget(sem_key,2,0777);
printf("This is customer!\n");
int x;
while(1)
{
if((x=semctl(sem_id,0,GETVAL,NULL))>0)
{
printf("x is %d\n",x);
printf("============================\n");
```

```
printf("Resource(s) of the store before custom are: \n");
system("cat /tmp/store");
printf("\n");
printf("Customed %d, Remaining
%d\n",semctl(sem_id,1,GETVAL,NULL),semctl(sem_id,0,GETVAL,NULL));
printf("\nNow customing...\n");
sem_p(sem_id,0);/*消费者消费 1 个资源，仓库中资源数量减少 1 个*/
sem_v(sem_id,1);/*仓库空位增加 1 个*/
customing();
printf("Resource(s) of the store after customed are: \n");
system("cat /tmp/store");
printf("\n");
printf("Customed %d, Remaining
%d\n\n",semctl(sem_id,1,GETVAL,NULL),semctl(sem_id,0,GETVAL,NULL));
}
else
{
printf("============================\n");
printf("None!\n");
}
sleep(1);
}
}
```