```c
/* 实验 10 参考代码 */
/* 2sem_producer.c */
/* 方法 3：1 个信号量，统计仓库中资源数量 */

#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"
#define store_file "/tmp/store2"
#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;
if((s_size=write(store_fd,buff,strlen(buff)))<0)
{
printf("Producer: write error!\n");
return -1;
}
printf("Produced '%c'",buff[0]);
close(store_fd);
return 0;
}
```

```
int main()
{
key_t sem_key;
int sem_id;
int x;

int i=0,y,fd;
char bufer[1000];

sem_key = ftok("/",'z');
sem_id = semget(sem_key,1,0777|IPC_CREAT);


sem_init(sem_id,0,1);

printf("This is producer!\n");
while(1)
{
sem_p(sem_id,0);/*生产者使用信号量，消费者不能使用*/
fd=open(store_file,O_RDONLY|O_CREAT,0666);
y=read(fd,bufer,sizeof(bufer));
close(fd);
printf("y = %d\n",y);
if(y>=0 && y<100)
{
printf("============================\n");
printf("----> Before produce:  <----\n");
printf("Resource(s) of the store before produce: \n");
system("cat /tmp/store2");
printf("\n");
printf("Produced %d, Free %d\n",y,100-y);
printf("Now producing...\n");
produce();
printf("\n====> After produced: <====\n");
printf("Resource(s) of the store after produce: \n");
system("cat /tmp/store2");
printf("\n");
printf("Produced %d, Free %d\n",y+1,100-y-1);
}
else
{
```

```
printf("============================\n");
printf("Full!\n\n");
}
sem_v(sem_id,0);/*生产者释放信号量，消费者可以使用*/
sleep(1);
}
}
```

```c
/* 实验 10 参考代码 */
/* 1sem_customer.c */
/* 方法 3：1 个信号量，统计仓库中资源数量 */

#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"
#define store_file "/tmp/store2"
#define tmp_file "/tmp/tmp22"
#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");
}
myfilecopy(store_file,tmp_file,1,MAX_FILE_SIZE);
myfilecopy(tmp_file,store_file,0,MAX_FILE_SIZE);
unlink(tmp_file);
close(store_fd);
return 0;
}

int main()
{
int sem_key,sem_id;
sem_key = ftok("/",'z');
sem_id = semget(sem_key,1,0777);
printf("This is customer!\n");
int x;

int i=0,y,fd;
char bufer[1000];

while(1)
{
sem_p(sem_id,0);/*消费者使用信号量，生产者不能使用*/

fd=open(store_file,O_RDONLY);
y=read(fd,bufer,sizeof(bufer));
```

```
close(fd);
printf("y = %d\n",y);

if(y>0 && y<=100)
{
printf("============================\n");
printf("----> Before custom:  <----\n");
printf("Resource(s) of the store before custom are: \n");
system("cat /tmp/store2");
printf("\n");
printf("Customed %d, Free %d\n",100-y,100-y);
printf("\nNow customing...\n");
customing();
printf("\n====> After customed: <====\n");
printf("Resource(s) of the store after customed are: \n");
system("cat /tmp/store2");
printf("\n");
printf("Customed %d, Free %d\n",100-y+1,y-1);
}
else
{
printf("============================\n");
printf("None!\n");
}
sem_v(sem_id,0);/*消费者释放信号量，生产者可以使用*/
sleep(1);
}
}
```