

Лабораторная работа № 7 – Семафоры и синхронизация

Цель работы

Освоение семафоров (semaphores) как эффективных средств синхронизации доступа процессов к разделяемым ресурсам операционной системы, а также синхронизации доступа потоков (в части 2) к разделяемым ресурсам процесса.

Пункт 1

Скомпилируйте и выполните программу `gener_sem.cpp`, иллюстрирующую создание наборов с семафорами или получение доступа к ним. Запустите программу несколько раз и после каждого ее завершения выполните команду `ipcs -s`. Поясните зависимость процедуры создания семафоров от используемых в вызове `semget()` флагов.

```
dani@dani-VirtualBox:~/Рабочий стол/os/spbstu-os-labs/labs/lab7/src1$ ipcs -s
----- Массивы семафоров -----
ключ  semid      владелец права nsems
-----
dani@dani-VirtualBox:~/Рабочий стол/os/spbstu-os-labs/labs/lab7/src1$ ./gener
sem1 identifier is 0
semget: IPC_CREATE | IPC_EXCL | 0666: File exists
sem2 identifier is -1
sem3 identifier is 1
dani@dani-VirtualBox:~/Рабочий стол/os/spbstu-os-labs/labs/lab7/src1$ ipcs -s
----- Массивы семафоров -----
ключ  semid      владелец права nsems
-----
0x5305811b 0      dani      666          3
0x00000000 1      dani      600          3
dani@dani-VirtualBox:~/Рабочий стол/os/spbstu-os-labs/labs/lab7/src1$ ./gener
sem1 identifier is 0
semget: IPC_CREATE | IPC_EXCL | 0666: File exists
sem2 identifier is -1
sem3 identifier is 2
dani@dani-VirtualBox:~/Рабочий стол/os/spbstu-os-labs/labs/lab7/src1$ ipcs -s
----- Массивы семафоров -----
ключ  semid      владелец права nsems
-----
0x5305811b 0      dani      666          3
0x00000000 1      dani      600          3
0x00000000 2      dani      600          3
dani@dani-VirtualBox:~/Рабочий стол/os/spbstu-os-labs/labs/lab7/src1$ ./gener
sem1 identifier is 0
semget: IPC_CREATE | IPC_EXCL | 0666: File exists
sem2 identifier is -1
sem3 identifier is 3
dani@dani-VirtualBox:~/Рабочий стол/os/spbstu-os-labs/labs/lab7/src1$ ipcs -s
----- Массивы семафоров -----
ключ  semid      владелец права nsems
-----
0x5305811b 0      dani      666          3
0x00000000 1      dani      600          3
0x00000000 2      dani      600          3
0x00000000 3      dani      600          3
dani@dani-VirtualBox:~/Рабочий стол/os/spbstu-os-labs/labs/lab7/src1$
```

Рис. 7-1 Работа программы `gener_sem.cpp`.

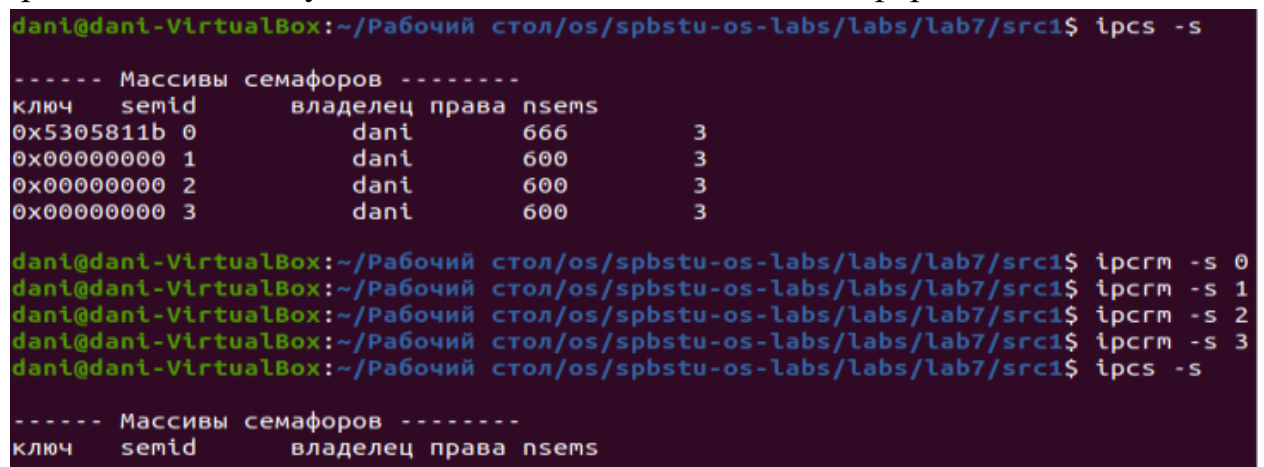
Первый набор создается с флагом **IPC_CREATE**, если набора с таким ключом нет — создается новый, а если такой набор существует — **semget()** вернет его идентификатор. Это мы видим из вывода программы — набор было создан лишь один раз (один идентификатор).

При создании второго набора был дополнительно указан флаг **IPC_EXCL**, что означает, что мы всегда хотим создавать новый набор. Так как мы уже создали набор (**sem1**) с таким же ключом, то мы получим ошибку.

Флаг **IPC_PRIVATE** — Набор **sem3** будет создаваться при каждом новом запуске программы. Причем каждый раз с новым уникальным идентификатором.

Пункт 2

Удалите созданные на предыдущем шаге семафоры с помощью команды **ipcrm** с соответствующей опцией и значением **id** семафора или ключа.



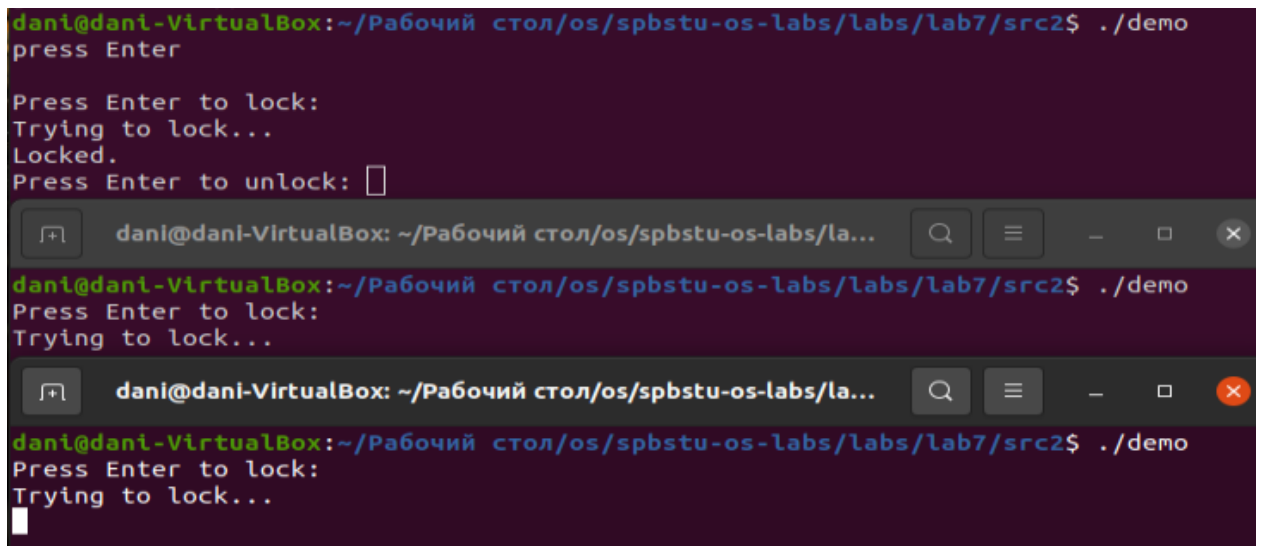
```
dani@dani-VirtualBox:~/Рабочий стол/os/spbstu-os-labs/labs/lab7/src1$ ipcs -s
----- Массивы семафоров -----
ключ  semid      владелец права nsems
0x5305811b 0          dani      666      3
0x00000000 1          dani      600      3
0x00000000 2          dani      600      3
0x00000000 3          dani      600      3

dani@dani-VirtualBox:~/Рабочий стол/os/spbstu-os-labs/labs/lab7/src1$ ipcrm -s 0
dani@dani-VirtualBox:~/Рабочий стол/os/spbstu-os-labs/labs/lab7/src1$ ipcrm -s 1
dani@dani-VirtualBox:~/Рабочий стол/os/spbstu-os-labs/labs/lab7/src1$ ipcrm -s 2
dani@dani-VirtualBox:~/Рабочий стол/os/spbstu-os-labs/labs/lab7/src1$ ipcrm -s 3
dani@dani-VirtualBox:~/Рабочий стол/os/spbstu-os-labs/labs/lab7/src1$ ipcs -s
----- Массивы семафоров -----
ключ  semid      владелец права nsems
```

Рис. 7-2 Удаление семафоров.

Пункт 3

Скомпилируйте **semdemo.cpp**, демонстрирующую организацию разделения доступа к общему ресурсу между несколькими процессами с помощью технологии семафоров. Запустите сразу несколько процессов на разных терминалах и проанализируйте их взаимодействие и соблюдение очередности в попытках получения общего ресурса.



```
dani@dani-VirtualBox:~/Рабочий стол/os/spbstu-os-labs/labs/lab7/src2$ ./demo
press Enter

Press Enter to lock:
Trying to lock...
Locked.
Press Enter to unlock: 

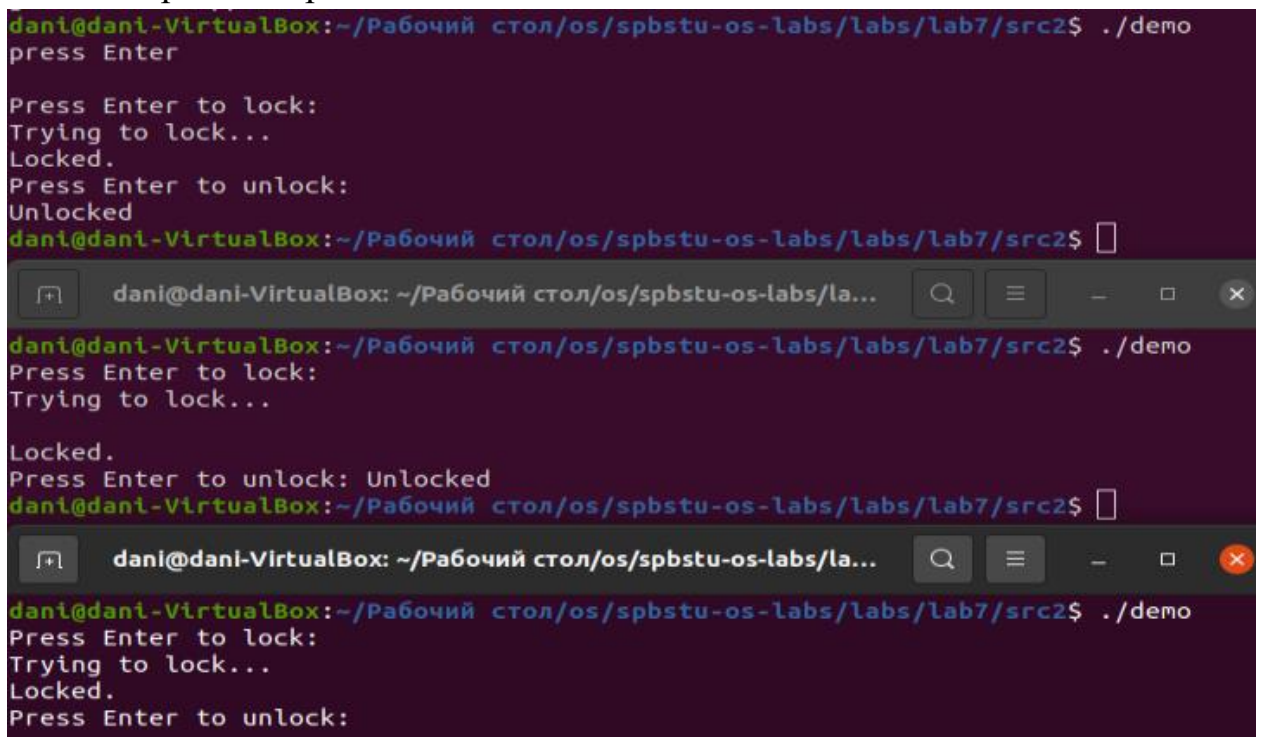
dani@dani-VirtualBox: ~/Рабочий стол/os/spbstu-os-labs/la...
dani@dani-VirtualBox:~/Рабочий стол/os/spbstu-os-labs/labs/lab7/src2$ ./demo
Press Enter to lock:
Trying to lock...

dani@dani-VirtualBox:~/Рабочий стол/os/spbstu-os-labs/labs/lab7/src2$ ./demo
Press Enter to lock:
Trying to lock...

```

Рис. 7-3 Демонстрация работы программы semdemo.cpp.

Мы заблокировали наш семафор в первом процессе, второй и третий процесс ожидают разблокировки.



```
dani@dani-VirtualBox:~/Рабочий стол/os/spbstu-os-labs/labs/lab7/src2$ ./demo
press Enter

Press Enter to lock:
Trying to lock...
Locked.
Press Enter to unlock:
Unlocked
dani@dani-VirtualBox:~/Рабочий стол/os/spbstu-os-labs/labs/lab7/src2$ 

dani@dani-VirtualBox: ~/Рабочий стол/os/spbstu-os-labs/la...
dani@dani-VirtualBox:~/Рабочий стол/os/spbstu-os-labs/labs/lab7/src2$ ./demo
Press Enter to lock:
Trying to lock...

Locked.
Press Enter to unlock: Unlocked
dani@dani-VirtualBox:~/Рабочий стол/os/spbstu-os-labs/labs/lab7/src2$ 

dani@dani-VirtualBox: ~/Рабочий стол/os/spbstu-os-labs/la...
dani@dani-VirtualBox:~/Рабочий стол/os/spbstu-os-labs/labs/lab7/src2$ ./demo
Press Enter to lock:
Trying to lock...
Locked.
Press Enter to unlock:
```

Рис. 7-4 Демонстрация работы программы semdemo.cpp.

Разблокируем семафор в первом процессе, второй процесс займет семафор. Третий продолжит ожидание. Таким образом каждый процесс будет по очереди занимать семафор.

Пункт 4

Скомпилируйте программу semrm.cpp и произведите с ее помощью удаление созданного на предыдущем шаге семафора. Поясните, почему данная программа удаляет только те семафоры, которые были созданы при выполнении программы semdemo.cpp.

```

dani@dani-VirtualBox:~/Рабочий стол/os/spbstu-os-labs/labs/lab7/src2$ ipcs -s

----- Массивы семафоров -----
ключ   semid   владелец права nsems
0x4a05811c 7      dani      666        1

dani@dani-VirtualBox:~/Рабочий стол/os/spbstu-os-labs/labs/lab7/src2$ cd ../src1
dani@dani-VirtualBox:~/Рабочий стол/os/spbstu-os-labs/labs/lab7/src1$ ./gener
sem1 identifier is 8
semget: IPC_CREATE | IPC_EXCL | 0666: File exists
sem2 identifier is -1
sem3 identifier is 9
dani@dani-VirtualBox:~/Рабочий стол/os/spbstu-os-labs/labs/lab7/src1$ ipcs -s

----- Массивы семафоров -----
ключ   semid   владелец права nsems
0x4a05811c 7      dani      666        1
0x5305811b 8      dani      666        3
0x00000000 9      dani      600        3

dani@dani-VirtualBox:~/Рабочий стол/os/spbstu-os-labs/labs/lab7/src1$ cd ../src2
dani@dani-VirtualBox:~/Рабочий стол/os/spbstu-os-labs/labs/lab7/src2$ ./semrm
dani@dani-VirtualBox:~/Рабочий стол/os/spbstu-os-labs/labs/lab7/src2$ ipcs -s

----- Массивы семафоров -----
ключ   semid   владелец права nsems
0x5305811b 8      dani      666        3
0x00000000 9      dani      600        3

dani@dani-VirtualBox:~/Рабочий стол/os/spbstu-os-labs/labs/lab7/src2$

```

Рис. 7-5 Удаление семафора.

Как видно программа **semrm** удаляет только семафор оставшийся от **semdemo**, потому что они используют один и тот же ключ.

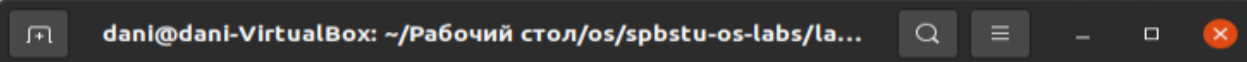
Пункт 5

Попробуйте удалить семафор с помощью запуска **semrm.cpp** во время исполнения **semdemo.cpp** и проанализируйте ситуацию.

```

dani@dani-VirtualBox:~/Рабочий стол/os/spbstu-os-labs/labs/lab7/src2$ ./demo
Press Enter to lock:
Trying to lock...
semop: Invalid argument
dani@dani-VirtualBox:~/Рабочий стол/os/spbstu-os-labs/labs/lab7/src2$

```



```

dani@dani-VirtualBox:~/Рабочий стол/os/spbstu-os-labs/labs/lab7/src2$ ./semrm
dani@dani-VirtualBox:~/Рабочий стол/os/spbstu-os-labs/labs/lab7/src2$

```

Рис. 7-5 Удаление семафора во время работы.

Получили ошибку, так как семафор был удалён посреди выполнения программы.

Пункт 6

Попробуйте улучшить программу **semdemo.cpp**, например, предоставив процессу возможность после освобождения ресурса становиться снова в очередь на повторное его занятие (а не завершаться), организовав при этом завершение процесса по вводу какого-либо символа.

```

77 int main(void)
78 {
79     key_t key;
80     int semid;
81     char u_char = 'J';
82     struct sembuf sb;
83     int c;
84
85     sb.sem_num = 0;
86     sb.sem_op = -1; /* set to allocate resource */
87     sb.sem_flg = SEM_UNDO;
88
89     if ((key = ftok(".", u_char)) == -1) {
90         perror("ftok");
91         exit(1);
92     }
93
94     /* grab the semaphore set created by initsem: */
95     if ((semid = initsem(key, 1)) == -1) {
96         perror("initsem");
97         exit(1);
98     }
99
100    while(1) {
101        printf("Press Enter to lock (q to finish): ");
102        while((c = getchar()) != '\n') {
103            if (c == 'q') return 0;
104        }
105        printf("Trying to lock...\n");
106
107        if (semop(semid, &sb, 1) == -1) {
108            perror("semop");
109            exit(1);
110        }
111
112        printf("Locked.\n");
113        printf("Press Enter to unlock: ");
114        while((c = getchar()) != '\n');
115
116        sb.sem_op = 1; /* free resource */
117        if (semop(semid, &sb, 1) == -1) {
118            perror("semop");
119            exit(1);
120        }
121
122        printf("Unlocked\n");
123    }
124
125    return 0;
126 }

```

Рис. 7-6 Модифицированная часть программы semdemo.cpp.

Программа модифицирована таким образом, чтобы процесс мог циклически занимать семафор, а для завершения процесса нужно прописать 'q' на этапе занятия семафора.

Теперь программа работает следующим образом:


```

dani@dani-VirtualBox:~/Рабочий стол/os/spbstu-os-labs/labs/lab7/src4$ ./demo
Press Enter to lock (q to finish):
Trying to lock...
Locked.
Press Enter to unlock:
Unlocked
Press Enter to lock (q to finish): another lock
Trying to lock...
Locked.
Press Enter to unlock:
Unlocked
Press Enter to lock (q to finish): q
dani@dani-VirtualBox:~/Рабочий стол/os/spbstu-os-labs/labs/lab7/src4$

```

Рис. 7-7 Демонстрация работы модифицированной semdemo.cpp.

Пункт 7

Составьте программу, позволяющую мониторить количество процессов (типа semdemo), находящихся в состоянии ожидания освобождения ресурса (Trying to lock...) в каждый момент времени. Программа строится на основе вызова semctl() с соответствующими параметрами и запускается на отдельном терминале.

Листинг 7-1 monitorWaiting.cpp

```

#include <stdio.h>
#include <stdlib.h>
#include <unistd.h>
#include <sys/types.h>
#include <sys/ipc.h>
#include <sys/sem.h>
#define MAX_RETRIES 10

int main(void)
{
    char u_char = 'J';
    key_t key = ftok(".", u_char);
    int semid;
    if ((semid = semget(key, 1, 0)) < 0) {
        perror("semget");
        exit(1);
    }

    int waitingCount;
    while(1) {
        waitingCount = semctl(semid, 0, GETNCNT);
        if (waitingCount < 0) {
            perror("semctl");
            exit(1);
        }
        printf("Processes in queue: %d\n", waitingCount);
        sleep(10);
    }

    return 0;
}

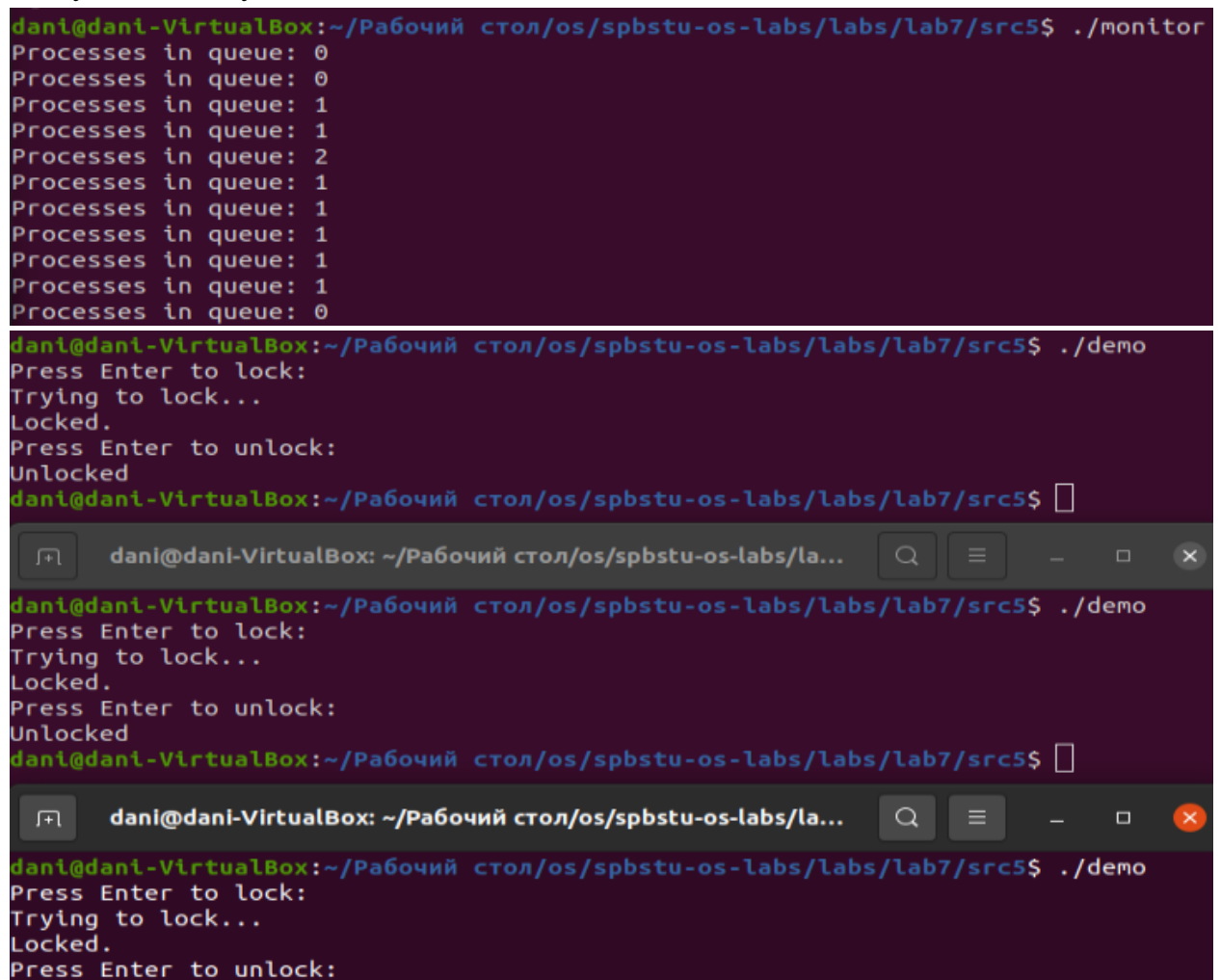
```

Для начала получим ключ, который будет точно таким же, как и в программе semdemo.cpp.

Затем, при помощи функции semget(), получим идентификатор множества семафоров, ассоциированного с ключом key.

После чего, в цикле мониторим число процессов, ожидающих разблокировки ресурса при помощи функции semctl() и значения GETNCNT.

Получаем следующее поведение:



```
dani@dani-VirtualBox:~/Рабочий стол/os/spbstu-os-labs/labs/lab7/src5$ ./monitor
Processes in queue: 0
Processes in queue: 0
Processes in queue: 1
Processes in queue: 1
Processes in queue: 2
Processes in queue: 1
Processes in queue: 1
Processes in queue: 1
Processes in queue: 1
Processes in queue: 1
Processes in queue: 1
Processes in queue: 0

dani@dani-VirtualBox:~/Рабочий стол/os/spbstu-os-labs/labs/lab7/src5$ ./demo
Press Enter to lock:
Trying to lock...
Locked.
Press Enter to unlock:
Unlocked
dani@dani-VirtualBox:~/Рабочий стол/os/spbstu-os-labs/labs/lab7/src5$ 
dani@dani-VirtualBox: ~/Рабочий стол/os/spbstu-os-labs/la...
dani@dani-VirtualBox:~/Рабочий стол/os/spbstu-os-labs/labs/lab7/src5$ ./demo
Press Enter to lock:
Trying to lock...
Locked.
Press Enter to unlock:
Unlocked
dani@dani-VirtualBox:~/Рабочий стол/os/spbstu-os-labs/labs/lab7/src5$ 
dani@dani-VirtualBox: ~/Рабочий стол/os/spbstu-os-labs/la...
dani@dani-VirtualBox:~/Рабочий стол/os/spbstu-os-labs/labs/lab7/src5$ ./demo
Press Enter to lock:
Trying to lock...
Locked.
Press Enter to unlock:
```

Рис. 7-8 Демонстрация работы программы мониторинга ожидания процессов.

Здесь у нас идет следующая цепочка:

Мы на процессе 1 блокируем семафор – в очереди 0;

На процессе 2 встаем в ожидание – в очереди 1;

На процессе 3 встаем в ожидание – в очереди 2;

Процесс 1 освобождает семафор, второй процесс занимает его – в очереди 1;

Процесс 2 освобождает семафор, третий процесс занимает его – в очереди 0;

Вывод

В данной лабораторной работе мы познакомились с созданием семафоров и осуществлением с их помощью синхронизации между процессами в операционной системе Linux.