Московский Авиационный Институт

(Национальный Исследовательский Университет)

Институт №8 “Компьютерные науки и прикладная математика”

Кафедра №806 “Вычислительная математика и программирование”

Лабораторная работа №3 по курсу

«Операционные системы»

Группа: М8О-216БВ-24

Студент: Иванов И.П.

Преподаватель: Бахарев В.Д.

Оценка: _____

Дата: 22.12.25

Москва, 2025

Постановка задачи

Вариант 12.

Родительский процесс создает два дочерних процесса. Перенаправление стандартных потоков ввода-вывода показано на картинке выше. Child1 и Child2 можно «соединить» между собой дополнительным каналом. Родительский и дочерний процесс должны быть представлены разными программами. Родительский процесс принимает от пользователя строки произвольной длины и пересылает их в pipe1. Процесс child1 и child2 производят работу над строками. Child2 пересылает результат своей работы родительскому процессу. Родительский процесс полученный результат выводит в стандартный поток вывода.

Child1 переводит строки в верхний регистр. Child2 убирает все задвоенные пробелы.

Общий метод и алгоритм решения

Использованные системные вызовы:

- *pid_t fork(void)*; – создает дочерний процесс.
- *int shm_open(const char *name, int oflag, mode_t mode)*; – создает или открывает объект разделяемой памяти.
- *int ftruncate(int fd, off_t length)*; – устанавливает размер объекта разделяемой памяти.
- *void *mmap(void *addr, size_t length, int prot, int flags, int fd, off_t offset)*; – отображает разделяемую память в адресное пространство процесса.
- *sem_t *sem_open(const char *name, int oflag, mode_t mode, unsigned int value)*; – создает или открывает именованный семафор.
- *int sem_wait(sem_t *sem)*; – уменьшает значение семафора (операция P).
- *int sem_post(sem_t *sem)*; – увеличивает значение семафора (операция V).
- *int sem_close(sem_t *sem)*; – закрывает семафор.
- *int sem_unlink(const char *name)*; – удаляет именованный семафор из системы.
- *int munmap(void *addr, size_t length)*; – удаляет отображение разделяемой памяти.
- *int shm_unlink(const char *name)*; – удаляет объект разделяемой памяти из системы.
- *int execl(const char *path, const char *arg, ...)*; – заменяет образ текущей программы на указанную, принимая аргументы в качестве списка.
- *pid_t waitpid(pid_t pid, int *status, int options)*; – ожидает изменения состояния указанного дочернего процесса.

В рамках лабораторной работы была разработана программа, реализующая межпроцессное взаимодействие с использованием механизмов разделяемой памяти и семафоров. Родительский процесс создаёт два дочерних процесса с помощью системного вызова `fork()`, после чего каждый из дочерних процессов заменяет свой образ на отдельную исполняемую программу с использованием системного вызова `execl()`.

Для обмена данными между процессами используется именованная разделяемая память, отображаемая в адресное пространство процессов с помощью `mmap()`. Синхронизация доступа к разделяемой памяти осуществляется с использованием POSIX-семафоров, что позволяет обеспечить строгий порядок обработки данных и избежать состояний гонки.

Родительский процесс принимает от пользователя строки произвольной длины и записывает их в разделяемую память, после чего уведомляет первый дочерний процесс о готовности данных. Первый дочерний процесс выполняет обработку строки, переводя все символы в верхний регистр, и передаёт результат второму дочернему процессу через разделяемую память, используя семафоры для синхронизации. Второй дочерний процесс удаляет из строки все повторяющиеся пробелы и передаёт результат обратно родительскому процессу.

Родительский процесс получает обработанную строку и выводит результат в стандартный поток вывода. Завершение работы всех процессов осуществляется корректно по управляющей команде, при этом все используемые ресурсы — разделяемая память и семафоры — освобождаются. Данное решение демонстрирует использование механизмов межпроцессного взаимодействия и синхронизации процессов в операционной системе Linux.

Код программы

main.c

```
#include <unistd.h>
#include <sys/wait.h>
#include <stdlib.h>
#include <string.h>
#include <sys/mman.h>
#include <sys/stat.h>
#include <semaphore.h>
#include <fcntl.h>

#define SHM_NAME "shared_memory"

#define SEM_M2C1 "sem_m2c1"
#define SEM_C1C2 "sem_c1c2"
#define SEM_C2M  "sem_c2m"

#define SHM_SIZE 1024

int main(int argc, char *argv[])
{
    if (argc != 1)
    {
        return 1;
    }

    shm_unlink(SHM_NAME);
    sem_unlink(SEM_M2C1);
    sem_unlink(SEM_C1C2);
    sem_unlink(SEM_C2M);

    int shm_fd = shm_open(SHM_NAME, O_CREAT | O_RDWR, 0666);
```

```
if (shm_fd == -1)
{
    const char msg[] = "Error: shm_open failed\n";
    write(STDERR_FILENO, msg, sizeof(msg) - 1);
    exit(EXIT_FAILURE);
}

if (ftruncate(shm_fd, SHM_SIZE) == -1)
{
    const char msg[] = "Error: ftruncate failed\n";
    write(STDERR_FILENO, msg, sizeof(msg) - 1);
    exit(EXIT_FAILURE);
}

char *shm_ptr = mmap(NULL, SHM_SIZE,
                     PROT_READ | PROT_WRITE,
                     MAP_SHARED, shm_fd, 0);
if (shm_ptr == MAP_FAILED)
{
    const char msg[] = "Error: mmap failed\n";
    write(STDERR_FILENO, msg, sizeof(msg) - 1);
    exit(EXIT_FAILURE);
}

sem_t *sem_m2c1 = sem_open(SEM_M2C1, O_CREAT, 0666, 0);
sem_t *sem_c1c2 = sem_open(SEM_C1C2, O_CREAT, 0666, 0);
sem_t *sem_c2m  = sem_open(SEM_C2M,  O_CREAT, 0666, 0);

if (sem_m2c1 == SEM_FAILED ||
    sem_c1c2 == SEM_FAILED ||
    sem_c2m  == SEM_FAILED)
```

```
{
    const char msg[] = "Error: sem_open failed\n";
    write(STDERR_FILENO, msg, sizeof(msg) - 1);
    exit(EXIT_FAILURE);
}

pid_t pid = fork();
switch (pid)
{
    case -1:
    {
        const char msg[] = "Error: fork child1 failed\n";
        write(STDERR_FILENO, msg, sizeof(msg) - 1);
        exit(EXIT_FAILURE);
    }
    case 0:
    {
        execl("./child1", "child1", NULL);
        const char msg[] = "Error: execl child1 failed\n";
        write(STDERR_FILENO, msg, sizeof(msg) - 1);
        _exit(EXIT_FAILURE);
    }
    default:
        break;
}

pid = fork();
switch (pid)
{
    case -1:
    {
```

```
        const char msg[] = "Error: fork child2 failed\n";
        write(STDERR_FILENO, msg, sizeof(msg) - 1);
        exit(EXIT_FAILURE);
    }
    case 0:
    {
        execl("./child2", "child2", NULL);
        const char msg[] = "Error: execl child2 failed\n";
        write(STDERR_FILENO, msg, sizeof(msg) - 1);
        _exit(EXIT_FAILURE);
    }
    default:
        break;
}

const char msg[] = "Input text:\n";
write(STDOUT_FILENO, msg, sizeof(msg) - 1);

char buffer[256];
ssize_t bytes_read;

while ((bytes_read = read(STDIN_FILENO, buffer, sizeof(buffer) - 1)) > 0)
{
    buffer[bytes_read] = '\0';

    strncpy(shm_ptr, buffer, SHM_SIZE - 1);
    shm_ptr[SHM_SIZE - 1] = '\0';

    sem_post(sem_m2c1);
    sem_wait(sem_c2m);
```

```
        if (buffer[0] == 'q')
        {
            break;
        }

        write(STDOUT_FILENO, "Processed string: ", 18);
        write(STDOUT_FILENO, shm_ptr, strlen(shm_ptr));
    }

    wait(NULL);
    wait(NULL);

    munmap(shm_ptr, SHM_SIZE);
    close(shm_fd);

    sem_close(sem_m2c1);
    sem_close(sem_c1c2);
    sem_close(sem_c2m);

    shm_unlink(SHM_NAME);
    sem_unlink(SEM_M2C1);
    sem_unlink(SEM_C1C2);
    sem_unlink(SEM_C2M);

    return 0;
}
```

child1.c

```
#include <unistd.h>
#include <stdlib.h>
#include <ctype.h>
#include <string.h>
#include <sys/mman.h>
#include <sys/stat.h>
#include <semaphore.h>
```

```
#include <fcntl.h>

#define SHM_NAME "shared_memory"

#define SEM_M2C1 "sem_m2c1"
#define SEM_C1C2 "sem_c1c2"

#define SHM_SIZE 1024

int main(int argc, char *argv[])
{
   if (argc != 1)
      return 1;

   int shm_fd = shm_open(SHM_NAME, O_RDWR, 0666);
   char *shm_ptr = mmap(NULL, SHM_SIZE,
                        PROT_READ | PROT_WRITE,
                        MAP_SHARED, shm_fd, 0);

   sem_t *sem_m2c1 = sem_open(SEM_M2C1, 0);
   sem_t *sem_c1c2 = sem_open(SEM_C1C2, 0);

   while (1)
   {
      sem_wait(sem_m2c1);

      if (shm_ptr[0] == 'q')
      {
         sem_post(sem_c1c2);
         break;
      }

      for (int i = 0; shm_ptr[i]; ++i)
      {
         shm_ptr[i] = toupper((unsigned char)shm_ptr[i]);
      }

      sem_post(sem_c1c2);
   }

   munmap(shm_ptr, SHM_SIZE);
   close(shm_fd);
   sem_close(sem_m2c1);
   sem_close(sem_c1c2);

   return 0;
}
```

child2.c

```
#include <unistd.h>
#include <stdlib.h>
#include <string.h>
#include <sys/mman.h>
#include <sys/stat.h>
#include <semaphore.h>
#include <fcntl.h>

#define SHM_NAME "shared_memory"
```

```
#define SEM_C1C2 "sem_c1c2"
#define SEM_C2M  "sem_c2m"

#define SHM_SIZE 1024

int main(int argc, char *argv[])
{
   if (argc != 1)
      return 1;

   int shm_fd = shm_open(SHM_NAME, O_RDWR, 0666);
   char *shm_ptr = mmap(NULL, SHM_SIZE,
                        PROT_READ | PROT_WRITE,
                        MAP_SHARED, shm_fd, 0);

   sem_t *sem_c1c2 = sem_open(SEM_C1C2, 0);
   sem_t *sem_c2m  = sem_open(SEM_C2M, 0);

   while (1)
   {
      sem_wait(sem_c1c2);

      if (shm_ptr[0] == 'q')
      {
         sem_post(sem_c2m);
         break;
      }

      char result[SHM_SIZE];
      int i = 0, j = 0;
      int space = 0;

      while (shm_ptr[i])
      {
         if (shm_ptr[i] == ' ')
         {
            if (!space)
               result[j++] = ' ';
            space = 1;
         }
         else
         {
            result[j++] = shm_ptr[i];
            space = 0;
         }
         i++;
      }
      result[j] = '\0';

      strncpy(shm_ptr, result, SHM_SIZE - 1);
      shm_ptr[SHM_SIZE - 1] = '\0';

      sem_post(sem_c2m);
   }

   munmap(shm_ptr, SHM_SIZE);
   close(shm_fd);
   sem_close(sem_c1c2);
   sem_close(sem_c2m);
   return 0;
}
```

Протокол работы программы

```
/home/ivan/Documents/labs/os_lab3/cmake-build-debug/main
Input text:
asafasfa      afs aa s  a
Processed string: ASAFASFA AFS AA S A
aaffff  as a    a  a a
Processed string: AAFFFF AS A A A A
as a   as as s llflffl
Processed string: AS A AS AS S LLFLFFL
q

Process finished with exit code 0
```

```
→  cmake-build-debug git:(main)  strace -o strace_res.txt -f ./main
Input text:
asf as asas asas a
Processed string: ASF AS ASAS ASAS A
sssss   sksksksksksk
Processed string: SSSSS SKSKSKSKSKSK
sfak fksak kk     ka k k k
Processed string: SFAK FKSAK KK KA K K K
q
```

Strace:

65837 execve("./main", ["./main"], 0x7ffef3bed288 /* 86 vars */) = 0
65837 brk(NULL) = 0x5576c30bb000
65837 access("/etc/ld.so.preload", R_OK) = -1 ENOENT (No such file or directory)
65837 openat(AT_FDCWD, "/etc/ld.so.cache", O_RDONLY|O_CLOEXEC) = 3
65837 fstat(3, {st_mode=S_IFREG|0644, st_size=171339, ...}) = 0
65837 mmap(NULL, 171339, PROT_READ, MAP_PRIVATE, 3, 0) = 0x7f8af49ed000
65837 close(3) = 0
65837 openat(AT_FDCWD, "/usr/lib/libc.so.6", O_RDONLY|O_CLOEXEC) = 3
65837 read(3, "\177ELF\2\1\1\3\0\0\0\0\0\0\0\0\3\0>\0\1\0\0\0000x\2\0\0\0\0\0"..., 832) = 832
65837 pread64(3, "\6\0\0\0\4\0\0\0@\0\0\0\0\0\0\0@\0\0\0\0\0\0\0@\0\0\0\0\0\0\0"..., 896, 64) = 896
65837 fstat(3, {st_mode=S_IFREG|0755, st_size=2149728, ...}) = 0
65837 mmap(NULL, 8192, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_ANONYMOUS, -1, 0) = 0x7f8af49eb000
65837 pread64(3, "\6\0\0\0\4\0\0\0@\0\0\0\0\0\0\0@\0\0\0\0\0\0\0@\0\0\0\0\0\0\0"..., 896, 64) = 896
65837 mmap(NULL, 2174000, PROT_READ, MAP_PRIVATE|MAP_DENYWRITE, 3, 0) = 0x7f8af4600000
65837 mmap(0x7f8af4624000, 1515520, PROT_READ|PROT_EXEC, MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE, 3, 0x24000) = 0x7f8af4624000
65837 mmap(0x7f8af4796000, 454656, PROT_READ, MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE, 3, 0x196000) = 0x7f8af4796000
65837 mmap(0x7f8af4805000, 24576, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE, 3, 0x204000) = 0x7f8af4805000
65837 mmap(0x7f8af480b000, 31792, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_FIXED|MAP_ANONYMOUS, -1, 0) = 0x7f8af480b000
65837 close(3) = 0
65837 mmap(NULL, 12288, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_ANONYMOUS, -1, 0) = 0x7f8af49e8000
65837 arch_prctl(ARCH_SET_FS, 0x7f8af49e8740) = 0
65837 set_tid_address(0x7f8af49e8a10) = 65837
65837 set_robust_list(0x7f8af49e8a20, 24) = 0
65837 rseq(0x7f8af49e8680, 0x20, 0, 0x53053053) = 0
65837 mprotect(0x7f8af4805000, 16384, PROT_READ) = 0
65837 mprotect(0x5576b0b6a000, 4096, PROT_READ) = 0
65837 mprotect(0x7f8af4a56000, 8192, PROT_READ) = 0
65837 prlimit64(0, RLIMIT_STACK, NULL, {rlim_cur=8192*1024, rlim_max=RLIM64_INFINITY}) = 0
65837 getrandom("\xd3\x12\xf0\xa6\xfe\xb5\x0c\xd5", 8, GRND_NONBLOCK) = 8
65837 munmap(0x7f8af49ed000, 171339) = 0
65837 unlink("/dev/shm/shared_memory") = -1 ENOENT (No such file or directory)
65837 unlink("/dev/shm/sem.sem_m2c1") = -1 ENOENT (No such file or directory)
65837 unlink("/dev/shm/sem.sem_c1c2") = -1 ENOENT (No such file or directory)
65837 unlink("/dev/shm/sem.sem_c2m") = -1 ENOENT (No such file or directory)
65837 openat(AT_FDCWD, "/dev/shm/shared_memory", O_RDWR|O_CREAT|O_NOFOLLOW|O_CLOEXEC, 0666) = 3
65837 ftruncate(3, 1024) = 0
65837 mmap(NULL, 1024, PROT_READ|PROT_WRITE, MAP_SHARED, 3, 0) = 0x7f8af4a16000
65837 openat(AT_FDCWD, "/dev/shm/sem.sem_m2c1", O_RDWR|O_NOFOLLOW|O_CLOEXEC) = -1 ENOENT (No such file or directory)
65837 rt_sigprocmask(SIG_BLOCK, ~[], [], 8) = 0
65837 mmap(NULL, 4096, PROT_READ|PROT_WRITE, MAP_DROPPABLE|MAP_ANONYMOUS, -1, 0) = 0x7f8af4a15000
65837 mmap(NULL, 4096, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_ANONYMOUS, -1, 0) = 0x7f8af4a14000
65837 rt_sigprocmask(SIG_SETMASK, [], NULL, 8) = 0
65837 getrandom("\x24\x17\xd2\xfa\xf9\x0d\x73\xf0\x4a\x70\x3b\xa6\x13\xf2\x44\x89\x8c\x18\x90\xd2\x99\x9d\xbb\xcd\x13\xcf\xc4\xd4\xcd\x2a\xa3\x95", 32, 0) = 32

65837 newfstatat(AT_FDCWD, "/dev/shm/sem.WNKDvW", 0x7ffd41ae4850, AT_SYMLINK_NOFOLLOW) = -1 ENOENT (No such file or directory)
65837 openat(AT_FDCWD, "/dev/shm/sem.WNKDvW", O_RDWR|O_CREAT|O_EXCL|O_NOFOLLOW|O_CLOEXEC, 0666) = 4
65837 write(4, "\0\0\0\0\0\0\0\200\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0", 32) = 32
65837 mmap(NULL, 32, PROT_READ|PROT_WRITE, MAP_SHARED, 4, 0) = 0x7f8af4a13000
65837 link("/dev/shm/sem.WNKDvW", "/dev/shm/sem.sem_m2c1") = 0
65837 fstat(4, {st_mode=S_IFREG|0644, st_size=32, ...}) = 0
65837 brk(NULL) = 0x5576c30bb000
65837 brk(0x5576c30dc000) = 0x5576c30dc000
65837 unlink("/dev/shm/sem.WNKDvW") = 0
65837 close(4) = 0
65837 openat(AT_FDCWD, "/dev/shm/sem.sem_c1c2", O_RDWR|O_NOFOLLOW|O_CLOEXEC) = -1 ENOENT (No such file or directory)
65837 newfstatat(AT_FDCWD, "/dev/shm/sem.gW18h8", 0x7ffd41ae4850, AT_SYMLINK_NOFOLLOW) = -1 ENOENT (No such file or directory)
65837 openat(AT_FDCWD, "/dev/shm/sem.gW18h8", O_RDWR|O_CREAT|O_EXCL|O_NOFOLLOW|O_CLOEXEC, 0666) = 4
65837 write(4, "\0\0\0\0\0\0\0\200\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0", 32) = 32
65837 mmap(NULL, 32, PROT_READ|PROT_WRITE, MAP_SHARED, 4, 0) = 0x7f8af4a12000
65837 link("/dev/shm/sem.gW18h8", "/dev/shm/sem.sem_c1c2") = 0
65837 fstat(4, {st_mode=S_IFREG|0644, st_size=32, ...}) = 0
65837 unlink("/dev/shm/sem.gW18h8") = 0
65837 close(4) = 0
65837 openat(AT_FDCWD, "/dev/shm/sem.sem_c2m", O_RDWR|O_NOFOLLOW|O_CLOEXEC) = -1 ENOENT (No such file or directory)
65837 newfstatat(AT_FDCWD, "/dev/shm/sem.NUJTvg", 0x7ffd41ae4850, AT_SYMLINK_NOFOLLOW) = -1 ENOENT (No such file or directory)
65837 openat(AT_FDCWD, "/dev/shm/sem.NUJTvg", O_RDWR|O_CREAT|O_EXCL|O_NOFOLLOW|O_CLOEXEC, 0666) = 4
65837 write(4, "\0\0\0\0\0\0\0\200\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0", 32) = 32
65837 mmap(NULL, 32, PROT_READ|PROT_WRITE, MAP_SHARED, 4, 0) = 0x7f8af4a11000
65837 link("/dev/shm/sem.NUJTvg", "/dev/shm/sem.sem_c2m") = 0
65837 fstat(4, {st_mode=S_IFREG|0644, st_size=32, ...}) = 0
65837 unlink("/dev/shm/sem.NUJTvg") = 0
65837 close(4) = 0
65837 rt_sigprocmask(SIG_BLOCK, ~[], [], 8) = 0
65837 clone(child_stack=NULL, flags=CLONE_CHILD_CLEARTID|CLONE_CHILD_SETTID|SIGCHLD, child_tidptr=0x7f8af49e8a10) = 65838
65837 rt_sigprocmask(SIG_SETMASK, [], NULL, 8) = 0
65838 set_robust_list(0x7f8af49e8a20, 24 <unfinished ...>
65837 rt_sigprocmask(SIG_BLOCK, ~[] <unfinished ...>
65838 <... set_robust_list resumed>) = 0
65837 <... rt_sigprocmask resumed>, [], 8) = 0
65837 clone(child_stack=NULL, flags=CLONE_CHILD_CLEARTID|CLONE_CHILD_SETTID|SIGCHLD <unfinished ...>
65838 rt_sigprocmask(SIG_SETMASK, [], NULL, 8) = 0
65837 <... clone resumed>, child_tidptr=0x7f8af49e8a10) = 65839
65839 set_robust_list(0x7f8af49e8a20, 24 <unfinished ...>
65837 rt_sigprocmask(SIG_SETMASK, [] <unfinished ...>
65839 <... set_robust_list resumed>) = 0
65837 <... rt_sigprocmask resumed>, NULL, 8) = 0
65838 execve("./child1", ["child1"], 0x7ffd41ae4e68 /* 86 vars */ <unfinished ...>
65837 write(1, "Input text:\n", 12 <unfinished ...>
65839 rt_sigprocmask(SIG_SETMASK, [] <unfinished ...>
65837 <... write resumed>) = 12
65839 <... rt_sigprocmask resumed>, NULL, 8) = 0

65837 read(0 <unfinished ...>
65839 execve("./child2", ["child2"], 0x7ffd41ae4e68 /* 86 vars */ <unfinished ...>
65838 <... execve resumed>) = 0
65839 <... execve resumed>) = 0
65838 brk(NULL) = 0x55922108d000
65839 brk(NULL) = 0x56495e27f000
65838 access("/etc/ld.so.preload", R_OK) = -1 ENOENT (No such file or directory)
65839 access("/etc/ld.so.preload", R_OK <unfinished ...>
65838 openat(AT_FDCWD, "/etc/ld.so.cache", O_RDONLY|O_CLOEXEC <unfinished ...>
65839 <... access resumed>) = -1 ENOENT (No such file or directory)
65838 <... openat resumed>) = 3
65839 openat(AT_FDCWD, "/etc/ld.so.cache", O_RDONLY|O_CLOEXEC <unfinished ...>
65838 fstat(3 <unfinished ...>
65839 <... openat resumed>) = 3
65838 <... fstat resumed>, {st_mode=S_IFREG|0644, st_size=171339, ...}) = 0
65839 fstat(3 <unfinished ...>
65838 mmap(NULL, 171339, PROT_READ, MAP_PRIVATE, 3, 0 <unfinished ...>
65839 <... fstat resumed>, {st_mode=S_IFREG|0644, st_size=171339, ...}) = 0
65838 <... mmap resumed>) = 0x7ff67b465000
65839 mmap(NULL, 171339, PROT_READ, MAP_PRIVATE, 3, 0 <unfinished ...>
65838 close(3 <unfinished ...>
65839 <... mmap resumed>) = 0x7f26f3ccf000
65838 <... close resumed>) = 0
65839 close(3) = 0
65838 openat(AT_FDCWD, "/usr/lib/libc.so.6", O_RDONLY|O_CLOEXEC <unfinished ...>
65839 openat(AT_FDCWD, "/usr/lib/libc.so.6", O_RDONLY|O_CLOEXEC <unfinished ...>
65838 <... openat resumed>) = 3
65839 <... openat resumed>) = 3
65838 read(3 <unfinished ...>
65839 read(3 <unfinished ...>
65838 <... read resumed>, "\177ELF\2\1\1\3\0\0\0\0\0\0\0\0\3\0>\0\1\0\0\0000x\2\0\0\0\0\0"..., 832) = 832
65839 <... read resumed>, "\177ELF\2\1\1\3\0\0\0\0\0\0\0\0\3\0>\0\1\0\0\0000x\2\0\0\0\0\0"..., 832) = 832
65838 pread64(3 <unfinished ...>
65839 pread64(3 <unfinished ...>
65838 <... pread64 resumed>, "\6\0\0\0\4\0\0\0@\0\0\0\0\0\0\0@\0\0\0\0\0\0\0@\0\0\0\0\0\0\0"..., 896, 64) = 896
65839 <... pread64 resumed>, "\6\0\0\0\4\0\0\0@\0\0\0\0\0\0\0@\0\0\0\0\0\0\0@\0\0\0\0\0\0\0"..., 896, 64) = 896
65838 fstat(3 <unfinished ...>
65839 fstat(3 <unfinished ...>
65838 <... fstat resumed>, {st_mode=S_IFREG|0755, st_size=2149728, ...}) = 0
65839 <... fstat resumed>, {st_mode=S_IFREG|0755, st_size=2149728, ...}) = 0
65838 mmap(NULL, 8192, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_ANONYMOUS, -1, 0 <unfinished ...>
65839 mmap(NULL, 8192, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_ANONYMOUS, -1, 0 <unfinished ...>
65838 <... mmap resumed>) = 0x7ff67b463000
65839 <... mmap resumed>) = 0x7f26f3ccd000
65838 pread64(3 <unfinished ...>
65839 pread64(3 <unfinished ...>
65838 <... pread64 resumed>, "\6\0\0\0\4\0\0\0@\0\0\0\0\0\0\0@\0\0\0\0\0\0\0@\0\0\0\0\0\0\0"..., 896, 64) = 896
65839 <... pread64 resumed>, "\6\0\0\0\4\0\0\0@\0\0\0\0\0\0\0@\0\0\0\0\0\0\0@\0\0\0\0\0\0\0"..., 896, 64) = 896
65838 mmap(NULL, 2174000, PROT_READ, MAP_PRIVATE|MAP_DENYWRITE, 3, 0 <unfinished ...>
65839 mmap(NULL, 2174000, PROT_READ, MAP_PRIVATE|MAP_DENYWRITE, 3, 0) =

0x7f26f3a00000

```
65838 <... mmap resumed>)               = 0x7ff67b200000
65839 mmap(0x7f26f3a24000, 1515520, PROT_READ|PROT_EXEC,
MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE, 3, 0x24000 <unfinished ...>
65838 mmap(0x7ff67b224000, 1515520, PROT_READ|PROT_EXEC,
MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE, 3, 0x24000 <unfinished ...>
65839 <... mmap resumed>)               = 0x7f26f3a24000
65838 <... mmap resumed>)               = 0x7ff67b224000
65839 mmap(0x7f26f3b96000, 454656, PROT_READ, MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE,
3, 0x196000 <unfinished ...>
65838 mmap(0x7ff67b396000, 454656, PROT_READ, MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE,
3, 0x196000 <unfinished ...>
65839 <... mmap resumed>)               = 0x7f26f3b96000
65838 <... mmap resumed>)               = 0x7ff67b396000
65839 mmap(0x7f26f3c05000, 24576, PROT_READ|PROT_WRITE,
MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE, 3, 0x204000 <unfinished ...>
65838 mmap(0x7ff67b405000, 24576, PROT_READ|PROT_WRITE,
MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE, 3, 0x204000 <unfinished ...>
65839 <... mmap resumed>)               = 0x7f26f3c05000
65838 <... mmap resumed>)               = 0x7ff67b405000
65839 mmap(0x7f26f3c0b000, 31792, PROT_READ|PROT_WRITE,
MAP_PRIVATE|MAP_FIXED|MAP_ANONYMOUS, -1, 0 <unfinished ...>
65838 mmap(0x7ff67b40b000, 31792, PROT_READ|PROT_WRITE,
MAP_PRIVATE|MAP_FIXED|MAP_ANONYMOUS, -1, 0 <unfinished ...>
65839 <... mmap resumed>)               = 0x7f26f3c0b000
65838 <... mmap resumed>)               = 0x7ff67b40b000
65839 close(3 <unfinished ...>
65838 close(3 <unfinished ...>
65839 <... close resumed>)              = 0
65838 <... close resumed>)              = 0
65839 mmap(NULL, 12288, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_ANONYMOUS, -1, 0
<unfinished ...>
65838 mmap(NULL, 12288, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_ANONYMOUS, -1, 0
<unfinished ...>
65839 <... mmap resumed>)               = 0x7f26f3cca000
65838 <... mmap resumed>)               = 0x7ff67b460000
65839 arch_prctl(ARCH_SET_FS, 0x7f26f3cca740 <unfinished ...>
65838 arch_prctl(ARCH_SET_FS, 0x7ff67b460740 <unfinished ...>
65839 <... arch_prctl resumed>)         = 0
65838 <... arch_prctl resumed>)         = 0
65839 set_tid_address(0x7f26f3ccaa10 <unfinished ...>
65838 set_tid_address(0x7ff67b460a10 <unfinished ...>
65839 <... set_tid_address resumed>)    = 65839
65838 <... set_tid_address resumed>)    = 65838
65839 set_robust_list(0x7f26f3ccaa20, 24 <unfinished ...>
65838 set_robust_list(0x7ff67b460a20, 24 <unfinished ...>
65839 <... set_robust_list resumed>)    = 0
65838 <... set_robust_list resumed>)    = 0
65839 rseq(0x7f26f3cca680, 0x20, 0, 0x53053053 <unfinished ...>
65838 rseq(0x7ff67b460680, 0x20, 0, 0x53053053 <unfinished ...>
65839 <... rseq resumed>)               = 0
65838 <... rseq resumed>)               = 0
65839 mprotect(0x7f26f3c05000, 16384, PROT_READ <unfinished ...>
65838 mprotect(0x7ff67b405000, 16384, PROT_READ <unfinished ...>
65839 <... mprotect resumed>)           = 0
65838 <... mprotect resumed>)           = 0
65839 mprotect(0x564953bc0000, 4096, PROT_READ <unfinished ...>
```

```
65838 mprotect(0x5591f084f000, 4096, PROT_READ <unfinished ...>
65839 <... mprotect resumed>)          = 0
65838 <... mprotect resumed>)          = 0
65839 mprotect(0x7f26f3d38000, 8192, PROT_READ) = 0
65838 mprotect(0x7ff67b4ce000, 8192, PROT_READ <unfinished ...>
65839 prlimit64(0, RLIMIT_STACK, NULL <unfinished ...>
65838 <... mprotect resumed>)          = 0
65839 <... prlimit64 resumed>, {rlim_cur=8192*1024, rlim_max=RLIM64_INFINITY}) = 0
65838 prlimit64(0, RLIMIT_STACK, NULL <unfinished ...>
65839 getrandom( <unfinished ...>
65838 <... prlimit64 resumed>, {rlim_cur=8192*1024, rlim_max=RLIM64_INFINITY}) = 0
65839 <... getrandom resumed>"\xe3\xa4\x05\x3b\x33\x5c\x2e\xd8", 8, GRND_NONBLOCK) = 8
65839 munmap(0x7f26f3ccf000, 171339 <unfinished ...>
65838 getrandom( <unfinished ...>
65839 <... munmap resumed>)            = 0
65838 <... getrandom resumed>"\x11\x71\x9d\xec\xb7\x50\x3e\x65", 8, GRND_NONBLOCK) = 8
65839 openat(AT_FDCWD, "/dev/shm/shared_memory", O_RDWR|O_NOFOLLOW|O_CLOEXEC
<unfinished ...>
65838 munmap(0x7ff67b465000, 171339 <unfinished ...>
65839 <... openat resumed>)            = 3
65839 mmap(NULL, 1024, PROT_READ|PROT_WRITE, MAP_SHARED, 3, 0 <unfinished ...>
65838 <... munmap resumed>)            = 0
65839 <... mmap resumed>)              = 0x7f26f3cf8000
65838 openat(AT_FDCWD, "/dev/shm/shared_memory", O_RDWR|O_NOFOLLOW|O_CLOEXEC
<unfinished ...>
65839 openat(AT_FDCWD, "/dev/shm/sem.sem_c1c2", O_RDWR|O_NOFOLLOW|O_CLOEXEC
<unfinished ...>
65838 <... openat resumed>)            = 3
65839 <... openat resumed>)            = 4
65838 mmap(NULL, 1024, PROT_READ|PROT_WRITE, MAP_SHARED, 3, 0 <unfinished ...>
65839 fstat(4, {st_mode=S_IFREG|0644, st_size=32, ...}) = 0
65838 <... mmap resumed>)              = 0x7ff67b48e000
65839 brk(NULL <unfinished ...>
65838 openat(AT_FDCWD, "/dev/shm/sem.sem_m2c1", O_RDWR|O_NOFOLLOW|O_CLOEXEC
<unfinished ...>
65839 <... brk resumed>)               = 0x56495e27f000
65838 <... openat resumed>)            = 4
65839 brk(0x56495e2a0000 <unfinished ...>
65838 fstat(4 <unfinished ...>
65839 <... brk resumed>)               = 0x56495e2a0000
65838 <... fstat resumed>, {st_mode=S_IFREG|0644, st_size=32, ...}) = 0
65839 mmap(NULL, 32, PROT_READ|PROT_WRITE, MAP_SHARED, 4, 0 <unfinished ...>
65838 brk(NULL <unfinished ...>
65839 <... mmap resumed>)              = 0x7f26f3cf7000
65838 <... brk resumed>)               = 0x55922108d000
65839 close(4 <unfinished ...>
65838 brk(0x5592210ae000 <unfinished ...>
65839 <... close resumed>)             = 0
65839 openat(AT_FDCWD, "/dev/shm/sem.sem_c2m", O_RDWR|O_NOFOLLOW|O_CLOEXEC
<unfinished ...>
65838 <... brk resumed>)               = 0x5592210ae000
65839 <... openat resumed>)            = 4
65839 fstat(4 <unfinished ...>
65838 mmap(NULL, 32, PROT_READ|PROT_WRITE, MAP_SHARED, 4, 0 <unfinished ...>
65839 <... fstat resumed>, {st_mode=S_IFREG|0644, st_size=32, ...}) = 0
65838 <... mmap resumed>)              = 0x7ff67b48d000
65839 mmap(NULL, 32, PROT_READ|PROT_WRITE, MAP_SHARED, 4, 0 <unfinished ...>
```

65838 close(4 <unfinished ...>
65839 <... mmap resumed>) = 0x7f26f3cf6000
65838 <... close resumed>) = 0
65839 close(4 <unfinished ...>
65838 openat(AT_FDCWD, "/dev/shm/sem.sem_c1c2", O_RDWR|O_NOFOLLOW|O_CLOEXEC
<unfinished ...>
65839 <... close resumed>) = 0
65838 <... openat resumed>) = 4
65839 futex(0x7f26f3cf7000, FUTEX_WAIT_BITSET|FUTEX_CLOCK_REALTIME, 0, NULL, FUTEX_BITSET_MATCH_ANY <unfinished ...>
65838 fstat(4, {st_mode=S_IFREG|0644, st_size=32, ...}) = 0
65838 mmap(NULL, 32, PROT_READ|PROT_WRITE, MAP_SHARED, 4, 0) = 0x7ff67b48c000
65838 close(4) = 0
65838 futex(0x7ff67b48d000, FUTEX_WAIT_BITSET|FUTEX_CLOCK_REALTIME, 0, NULL, FUTEX_BITSET_MATCH_ANY <unfinished ...>
65837 <... read resumed>, "asf as asas asas a \n", 255) = 20
65837 futex(0x7f8af4a13000, FUTEX_WAKE, 1) = 1
65838 <... futex resumed>) = 0
65837 futex(0x7f8af4a11000, FUTEX_WAIT_BITSET|FUTEX_CLOCK_REALTIME, 0, NULL, FUTEX_BITSET_MATCH_ANY <unfinished ...>
65838 futex(0x7ff67b48c000, FUTEX_WAKE, 1) = 1
65839 <... futex resumed>) = 0
65838 futex(0x7ff67b48d000, FUTEX_WAIT_BITSET|FUTEX_CLOCK_REALTIME, 0, NULL, FUTEX_BITSET_MATCH_ANY <unfinished ...>
65839 futex(0x7f26f3cf6000, FUTEX_WAKE, 1 <unfinished ...>
65837 <... futex resumed>) = 0
65839 <... futex resumed>) = 1
65837 write(1, "Processed string: ", 18 <unfinished ...>
65839 futex(0x7f26f3cf7000, FUTEX_WAIT_BITSET|FUTEX_CLOCK_REALTIME, 0, NULL, FUTEX_BITSET_MATCH_ANY <unfinished ...>
65837 <... write resumed>) = 18
65837 write(1, "ASF AS ASAS ASAS A \n", 20) = 20
65837 read(0, "sssss   sksksksksksk   \n", 255) = 26
65837 futex(0x7f8af4a13000, FUTEX_WAKE, 1) = 1
65838 <... futex resumed>) = 0
65837 futex(0x7f8af4a11000, FUTEX_WAIT_BITSET|FUTEX_CLOCK_REALTIME, 0, NULL, FUTEX_BITSET_MATCH_ANY <unfinished ...>
65838 futex(0x7ff67b48c000, FUTEX_WAKE, 1 <unfinished ...>
65839 <... futex resumed>) = 0
65838 <... futex resumed>) = 1
65839 futex(0x7f26f3cf6000, FUTEX_WAKE, 1 <unfinished ...>
65838 futex(0x7ff67b48d000, FUTEX_WAIT_BITSET|FUTEX_CLOCK_REALTIME, 0, NULL, FUTEX_BITSET_MATCH_ANY <unfinished ...>
65839 <... futex resumed>) = 1
65837 <... futex resumed>) = 0
65839 futex(0x7f26f3cf7000, FUTEX_WAIT_BITSET|FUTEX_CLOCK_REALTIME, 0, NULL, FUTEX_BITSET_MATCH_ANY <unfinished ...>
65837 write(1, "Processed string: ", 18) = 18
65837 write(1, "SSSSS SKSKSKSKSKSK \n", 20) = 20
65837 read(0, "sfak fksak kk   ka k k k   \n", 255) = 31
65837 futex(0x7f8af4a13000, FUTEX_WAKE, 1) = 1
65838 <... futex resumed>) = 0
65837 futex(0x7f8af4a11000, FUTEX_WAIT_BITSET|FUTEX_CLOCK_REALTIME, 0, NULL, FUTEX_BITSET_MATCH_ANY <unfinished ...>
65838 futex(0x7ff67b48c000, FUTEX_WAKE, 1) = 1
65839 <... futex resumed>) = 0
65838 futex(0x7ff67b48d000, FUTEX_WAIT_BITSET|FUTEX_CLOCK_REALTIME, 0, NULL,

```
FUTEX_BITSET_MATCH_ANY <unfinished ...>
65839 futex(0x7f26f3cf6000, FUTEX_WAKE, 1 <unfinished ...>
65837 <... futex resumed>)              = 0
65839 <... futex resumed>)              = 1
65837 write(1, "Processed string: ", 18) = 18
65839 futex(0x7f26f3cf7000, FUTEX_WAIT_BITSET|FUTEX_CLOCK_REALTIME, 0, NULL,
FUTEX_BITSET_MATCH_ANY <unfinished ...>
65837 write(1, "SFAK FKSAK KK KA K K K \n", 24) = 24
65837 read(0, "q\n", 255)               = 2
65837 futex(0x7f8af4a13000, FUTEX_WAKE, 1) = 1
65838 <... futex resumed>)              = 0
65837 futex(0x7f8af4a11000, FUTEX_WAIT_BITSET|FUTEX_CLOCK_REALTIME, 0, NULL,
FUTEX_BITSET_MATCH_ANY <unfinished ...>
65838 futex(0x7ff67b48c000, FUTEX_WAKE, 1) = 1
65839 <... futex resumed>)              = 0
65838 munmap(0x7ff67b48e000, 1024 <unfinished ...>
65839 futex(0x7f26f3cf6000, FUTEX_WAKE, 1 <unfinished ...>
65838 <... munmap resumed>)             = 0
65838 close(3 <unfinished ...>
65837 <... futex resumed>)              = 0
65839 <... futex resumed>)              = 1
65838 <... close resumed>)              = 0
65837 wait4(-1 <unfinished ...>
65838 munmap(0x7ff67b48d000, 32 <unfinished ...>
65839 munmap(0x7f26f3cf8000, 1024 <unfinished ...>
65838 <... munmap resumed>)             = 0
65839 <... munmap resumed>)             = 0
65838 munmap(0x7ff67b48c000, 32 <unfinished ...>
65839 close(3 <unfinished ...>
65838 <... munmap resumed>)             = 0
65839 <... close resumed>)              = 0
65838 exit_group(0 <unfinished ...>
65839 munmap(0x7f26f3cf7000, 32 <unfinished ...>
65838 <... exit_group resumed>)         = ?
65839 <... munmap resumed>)             = 0
65839 munmap(0x7f26f3cf6000, 32 <unfinished ...>
65838 +++ exited with 0 +++
65839 <... munmap resumed>)             = 0
65837 <... wait4 resumed>, NULL, 0, NULL) = 65838
65837 --- SIGCHLD {si_signo=SIGCHLD, si_code=CLD_EXITED, si_pid=65838, si_uid=1000, si_status=0,
si_utime=0, si_stime=0} ---
65839 exit_group(0 <unfinished ...>
65837 wait4(-1 <unfinished ...>
65839 <... exit_group resumed>)         = ?
65839 +++ exited with 0 +++
65837 <... wait4 resumed>, NULL, 0, NULL) = 65839
65837 --- SIGCHLD {si_signo=SIGCHLD, si_code=CLD_EXITED, si_pid=65839, si_uid=1000, si_status=0,
si_utime=0, si_stime=0} ---
65837 munmap(0x7f8af4a16000, 1024)      = 0
65837 close(3)                          = 0
65837 munmap(0x7f8af4a13000, 32)        = 0
65837 munmap(0x7f8af4a12000, 32)        = 0
65837 munmap(0x7f8af4a11000, 32)        = 0
65837 unlink("/dev/shm/shared_memory")  = 0
65837 unlink("/dev/shm/sem.sem_m2c1")   = 0
65837 unlink("/dev/shm/sem.sem_c1c2")   = 0
65837 unlink("/dev/shm/sem.sem_c2m")    = 0
```

```
65837 exit_group(0)                     = ?
65837 +++ exited with 0 +++
```

Вывод

В ходе выполнения лабораторной работы была успешно реализована программа межпроцессного взаимодействия с использованием механизмов разделяемой памяти и семафоров.