Министерство образования Республики Беларусь

Учреждение образования

«Брестский государственный технический университет» Кафедра ИИТ

ОТЧЕТ

По лабораторной работе №7 Дисциплина «ОCиСП» за IV семестр

Выполнила:

студент группы ПО-3

Ковалёва А. И.

Проверила:

Давидюк Ю. И.

Брест 2020

**Лабораторная работа 7**

**Семафоры**

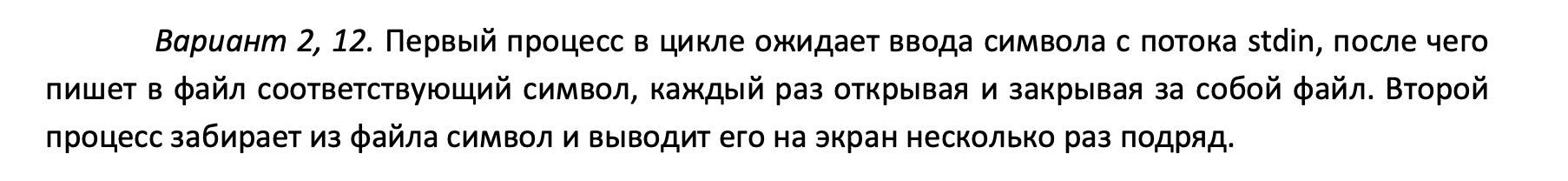
Вариант 12

**Цель:**

Ознакомиться с реализацией семафоров в ОС Linux.

**Задание:**

Написать две (или более) программы, которые, работая параллельно зациклено, обмениваются информацией согласно варианту. Передачу и получение информации каждым из процессов сопровождать выводом на экран нформации типа "процесс такой-то передал/получил такую-то информацию". Синхронизацию работы процессов реализовать с помощью семафоров. Учтите, что при организации совместного доступа к разделяемому ресурсу (например, файлу) вам понадобится применять, например, мьютексы.



**Код программы:**

**main1.c**

#include<stdio.h>

#include<fcntl.h>

#include<sys/types.h>

#include<unistd.h>

#include<stdlib.h>

#include<semaphore.h>

#include <pthread.h>

static pthread\_mutex\_t mutex = PTHREAD\_MUTEX\_INITIALIZER;

int main() {

sem\_t \*sem1, \*sem2;

int f;

if((sem1 = sem\_open("semaphore1", O\_RDWR|O\_CREAT, 0666,1)) == SEM\_FAILED) {

printf("ERROR sem\_open");

return 1;

}

if((sem2 = sem\_open("semaphore2", O\_RDWR|O\_CREAT, 0666,0)) == SEM\_FAILED) {

printf("ERROR sem\_open");

return 1;

}

// sem\_unlink("semaphore1");

char symbol;

while(symbol != '0') {

printf("Send: ");

scanf("%s", &symbol);

sem\_wait(sem1);

f = open("file", O\_WRONLY | O\_CREAT, 0666);

if(f < 0) {

printf("Can\'t open file\n");

exit(-1);

}

pthread\_mutex\_lock(&mutex);

write(f, &symbol, sizeof(symbol));

close(f);

pthread\_mutex\_unlock(&mutex);

sem\_post(sem1);

sem\_post(sem2);

}

sem\_close(sem1);

sem\_close(sem2);

return 0;

}

**main2.c**

#include<stdio.h>

#include<fcntl.h>

#include<sys/types.h>

#include<unistd.h>

#include<stdlib.h>

#include<semaphore.h>

#include <pthread.h>

static pthread\_mutex\_t mutex = PTHREAD\_MUTEX\_INITIALIZER;

int main() {

sem\_t \*sem1,\*sem2;

int f;

//sem\_unlink("semaphore2");

if((sem1 = sem\_open("semaphore1", O\_RDWR)) == SEM\_FAILED) {

printf("ERROR sem\_open");

return 1;

}

if((sem2=sem\_open("semaphore2", O\_RDWR|O\_CREAT, 0666,1)) == SEM\_FAILED) {

printf("ERROR sem\_open");

return 1;

}

char symbol[2];

while(symbol[0] != '0'){

sem\_wait(sem2);

sem\_wait(sem1);

f = open("file",O\_RDONLY);

if(f < 0) {

printf("Can\'t open file\n");

exit(-1);

}

pthread\_mutex\_lock(&mutex);

read(f, symbol, sizeof(symbol));

printf("Get: %s \n", symbol);

close(f);

pthread\_mutex\_unlock(&mutex);

sem\_post(sem1);

if(symbol[0] == '0') {

break;

}

int times = rand()%10 + 1;

for( int i = 0; i < times; i++) {

printf("%s",symbol);

}

printf("\n");

}

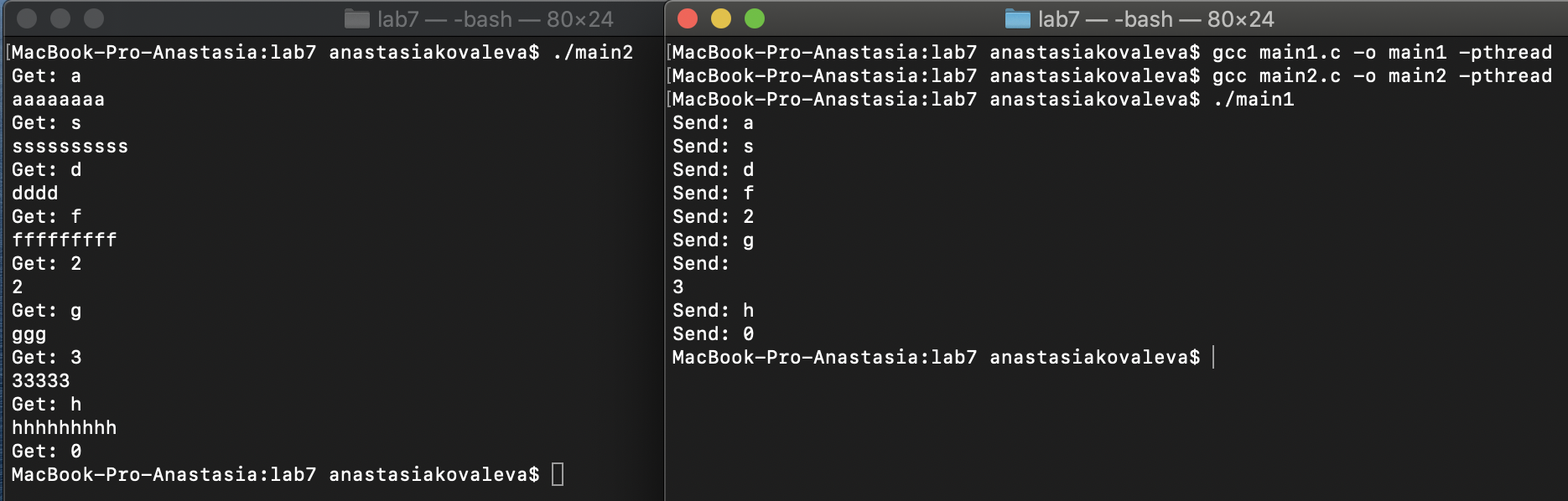
sem\_close(sem1);

sem\_close(sem2);

return 0;

}

**Результат выполнения:**



**Вывод:**

Ознакомилась с реализацией семафоров в ОС Linux.