Министерство образования Республики Беларусь
Учреждение образования
«Брестский государственный технический университет»
Кафедра ИИТ

Лабораторная работа №7
По дисциплине: «ОСиСП»
Тема: «Семафоры»

Выполнил:
Студент 2 курса
Группы ПО-7
Комиссаров А.Е.
Проверила:
Давидюк Ю.И.

Брест 2022

Цель работы: изучить основные принципы работы, ознакомиться с семафорами.

Задание :
Ознакомиться с руководством, теоретическими сведениями и лекционным материалом по использованию и функционированию средств синхронизации - семафоров Дейкстры, и их реализацией в Linux - System V IPC семафоры и POSIX-семафоры.

Написать две (или более) программы, которые, работая параллельно зацикленно, обмениваются информацией согласно варианту. Передачу и получение информации каждым из процессов сопровождать выводом на экран информации 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 мьютексы.

Для наглядности запускайте свои процессы в разных окнах терминала. Запустите программы в нескольких экземплярах (одну первую и две/три вторых, две первых и две вторых...).

Вариант 6. Первый процесс в цикле ожидает ввода символа с потока stdin, после чего пишет в файл соответствующий символ случайное количество раз, каждый раз открывая и закрывая за собой файл. Второй процесс забирает из файла символы и выводит на экран их количество.

Код программы:

1.cpp

```
#include <semaphore.h>
#include <time.h>
#include <stdio_ext.h>
#include <string.h>
#include <pthread.h>
#include <sys/types.h>
#include <unistd.h>
#include <fcntl.h>
#include <stdio.h>
#include <stdlib.h>

#include <sys/stat.h>

int main(){
        srand(time(0));
        int count = 0;
        int fd;
        sem_t *empty;
        sem_t *full;
        const char *semFull = "full";
        const char *semEmpty = "empty";
        char symbol;
        const char *path = "file.txt";
        pthread_mutex_t mutex = PTHREAD_MUTEX_INITIALIZER;
        empty = sem_open (semEmpty, O_CREAT, 0777, 1);
        full = sem_open(semFull, O_CREAT, 0777, 0);
        (void) umask(0);
        if ((fd = open(path, O_WRONLY | O_CREAT | O_TRUNC), 0777) < 0){printf("err(1)"); return -1;}
        else{
                while(1){
                        count = rand() % 25 + 1;     //count – кол-во раз, сколько мы будем вводить
символ в файл (от 1 до 25)
                        __fpurge(stdin);
                        scanf("%c", &symbol);
                        sem_wait(empty);
```

```
                    pthread_mutex_lock(&mutex);
                    fd = open(path, O_CREAT | O_WRONLY | O_TRUNC, 0777);
                    for(int i = 0; i < count; i++){            //запись символа count раз
                    write(fd, &symbol, sizeof(symbol));
                    }
                    close(fd);
                    printf("PID = %d sent symbol %c, %d times to file\n", getpid(), symbol, count);
                    pthread_mutex_unlock(&mutex);
                    sem_post(full);

            }
            pthread_mutex_destroy(&mutex);
            sem_close(empty);
            sem_close(full);
            sem_unlink(semEmpty);
            sem_unlink(semFull);
            unlink(path);
            return 0;
        }
}
```

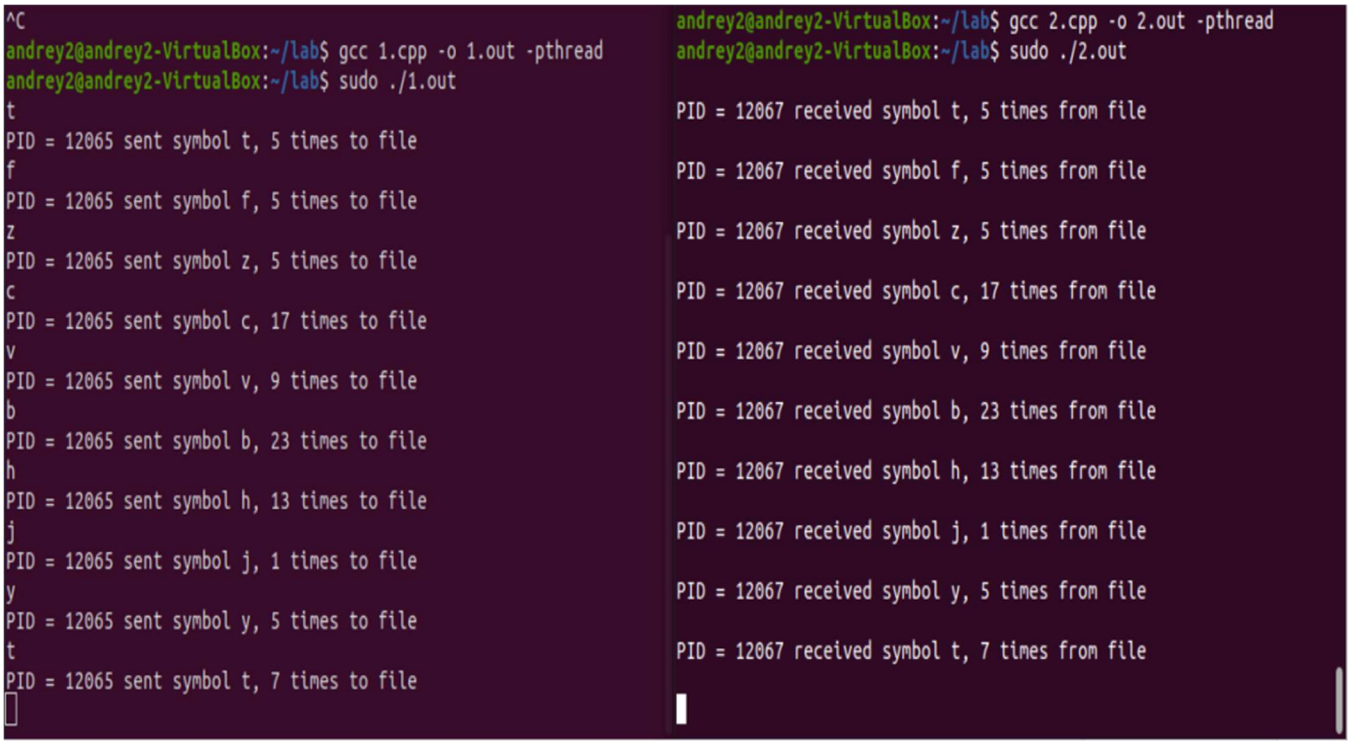
2.cpp

```
#include <semaphore.h>
#include <time.h>
#include <stdio_ext.h>
#include <string.h>
#include <pthread.h>
#include <sys/types.h>
#include <unistd.h>
#include <fcntl.h>
#include <stdio.h>
#include <stdlib.h>
#include <sys/stat.h>

int main(){
        srand(time(0));
        int fd;
    int count = 0;
        sem_t *empty;
        sem_t *full;
        const char *semFull = "full";
        const char *semEmpty = "empty";
        char symbol;
        const char *path = "file.txt";
        pthread_mutex_t mutex = PTHREAD_MUTEX_INITIALIZER;
        empty = sem_open (semEmpty, O_CREAT, 0777, 1);
        full = sem_open(semFull, O_CREAT, 0777, 0);
        (void) umask(0);
    while(1){
         printf("\n");                      //для равного вывода со вторым окном(там вводится символ)
         sem_wait(full);
         pthread_mutex_lock(&mutex);
         fd = open(path, O_RDONLY, 0777);           //открываем файл
         while(read(fd, &symbol, sizeof(char))>0){  //читаем символ и считаем количество раз
             count++;
         }
         close(fd);                                 //закрываем файл
         printf("PID = %d received symbol %c, %d times from file\n", getpid(), symbol, count);
         pthread_mutex_unlock(&mutex);
         sem_post(empty);
         count = 0;
    }
    pthread_mutex_destroy(&mutex);
        sem_close(empty);
        sem_close(full);
        sem_unlink(semEmpty);
        sem_unlink(semFull);
        unlink(path);
        return 0;
```

```
}
```

Результат работы программы:



Вывод: В ходе данной лабораторной работы изучил основные принципы работы, ознакомился с семафорами.