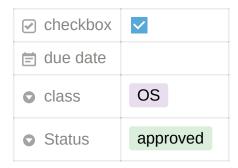


OS2 Lab16



Task

```
Решите задачу 11 с использованием двух процессов (а не нитей) и именованных семафоров-счетчиков
```

Notes

```
#include <stdio.h>
#include <stdlib.h>
#include <unistd.h>
#include <sys/types.h>
#include <semaphore.h>
#define NUM_STRINGS 10
int main() {
    sem_t* semParent = sem_open("/ParentSem", 0_CREAT, 0777, 0);
    sem_t* semChild = sem_open("/ChildSem", 0_CREAT, 0777, 1);
    switch (fork()) {
        case -1:
            perror("Fork failed");
            sem_destroy(semParent);
            sem_destroy(semChild);
            exit(-1);
        case 0:
            for (int i = 1; i \le NUM_STRINGS; i++) {
                sem_wait(semChild);
                printf("PARENT: %d\n", i);
                sem_post(semParent);
            break;
        default:
            for (int i = 1; i <= NUM_STRINGS; i++) \{
                sem_wait(semParent);
                printf("CHILD: %d\n", i);
                sem_post(semChild);
            }
            break;
    sem_close(semParent);
    sem_close(semChild);
    return 0;
}
```

OS2 Lab16

Именованные семафоры создаются функцией sem_open(3RT). При помощи этой же функции можно получить доступ к уже существующему именованному семафору. Эта функция имеет два обязательных параметра и два необязательных:

- const char * name имя семафора. Имя должно начинаться с символа ' / ' и не должно содержать других символов ' / '. Рекомендуется, чтобы имя не превышало 14 символов. В зависимости от реализации, объект с таким именем может либо появляться либо не появляться в корневом каталогекорневой файловой системы (в Solaris и Linux не появляется). В любом случае, для создания семафора не обязательно иметь право создания файлов в корневом каталоге.
- int flags флаги. Может принимать значения 0, 0_CREAT и 0_CREAT и 0_EXCL константы, определенные в <sys/fcntl.h>. Смысл этих значений аналогичен соответствующим значениям флагов в параметрах open(2). 0 означает попытку доступа к уже существующему семафору, 0_CREAT доступ к существующему семафору или попытку создания, если такого семафора нет, 0_EXCL ошибку, если при попытке создания обнаруживается, что такой семафор уже существует.
- mode t mode необязательный параметр, который используется, только если flags содержит бит O_CREAT. Обозначает права доступа к семафору, которые девятибитовой маской доступа, похожей на маску доступа к файлам. Как и у файла, у семафора есть идентификаторы хозяина и Идентификатор хозяина группы. устанавливается равным эффективному идентификатору пользователя процесса, создавшего семафор, идентификатор группы - эффективному идентификатору группы процесса.
- unsigned int value необязательный параметр, который

OS2 Lab16 2

Функция sem_open (3RT) возвращает указатель на семафор (sem_t *). При ошибке она возвращает нулевой указатель и устанавливает errno. Если процесс попытается несколько раз открыть один и тот же семафор, ему будут возвращать один и тот же указатель. Именованные семафоры всегда разделяемые между процессами. При доступе к существующему семафору проверяются права доступа по той же схеме, по которой в Unix системам проверяются права доступа к файлам. Для доступа к семафору процесс должен иметь права чтения и записи.

Для отсоединения от семафора и освобождения памяти из-под него необходимо использовать функцию sem_close(3RT). Эта функция неявно вызывается при exit(2) и exec(2). Однако закрытие именованного семафора процессом не прекращает существования семафора. Чтобы удалить семафор, необходимо вызвать функцию sem unlink (3RT). Это лишит новые процессы возможность видеть семафор как существующий (попытка исполнить sem open с именем этого семафора без флага О СВЕАТ приведет к ошибке) и позволит создать новый семафор с тем же именем. Однако если в момент sem unlink(3RT) один или несколько процессов работали с семафором, семафор продолжит свое существование до момента, пока все эти процессы не выполнят sem close (3RT). Исполнять семафора sem unlink(3RT) могут только владелец суперпользователь.

Во всем остальном именованный семафор не отличается от неименованного. Над ним можно выполнять те же операции, что и над неименованным, при помощи тех же самых функций.

Reading list