Politechnika Świętokrzyska w Kielcach
Wydział Elektrotechniki, Automatyki i Informatyki

Programowanie Współbieżne

laboratorium

Temat:

Semafory
(laboratorium nr 6)

Autor: **Mateusz Snoch**

Grupa: **3ID11A**

Data odbycia się laboratorium : **23.11.2017**

Data sporządzenia sprawozdania: **24.11.2017**

Przydatne dane:

semget – tworzenie lub otwieranie semafora
semop – wykonywanie operacji na semaforach:
>0 wartość tą dodaj do bieżącego semafora (uwolnij zasoby) operacja V(s)
=0 proces wywołujący funkcję semop chce czekać, aż wartością semafora stanie się 0
<0 proces wywołujący czeka aż wartość semafora stanie się większa niż (lub taka sama jak) wartość bezwzględna tego pola. Następnie zostaną zsumowane, czyli przydział zasobów operacja P(s). Np. s=1+(-1). s=0 semafor opuszczony.
Flagi semop:
SEM_UNDO - cofanie zmian wykonanych przez proces na tym semaforze jeżeli proces "padnie"
IPC_NOWAIT - na sem_flag informuje system że nie chcemy czekać , aż operacja będzie ukończona.
semctl – sterowanie semaforami:
IPC_RMID - usunięcie semafora
GETVAL - pobranie wartości semafora, funkcja zwróci jego wartość
SETVAL - nadawanie wartości semaforowych val w unii semun;

Zadanie 2

zmienić nazwę programu i uruchomić, jakie dostaliśmy identyfikatory?
zmienić nr projektu lub/i ścieżkę, porównać otrzymane wyniki

```
student@st06-lab319:~$ ./l6z2
ftok 0x3080001
student@st06-lab319:~$ ./l6z2a
ftok 0x3080001
student@st06-lab319:~$ ./l6z2b
ftok 0x5080001
```

Po zmianie nazwy programu identyfikator jest taki sam. Po zmianie ścieżki identyfikator jest różny.

Zadanie 3

```
#include <stdio.h>
#include <sys/types.h>
#include <sys/ipc.h>
#include <sys/sem.h>

#define PERMS 0666

static struct sembuf op_lock[1] = {
    0, -1, 0
};

static struct sembuf op_unlock[1] = {
```

```
    0, 1, 0
};

void blokuj(int semid)
{
if (semop(semid, &op_lock[0], 1)<0)
        perror("blad lokowania semafora");
}

void odblokuj(int semid)
{
if (semop(semid, &op_unlock[0], 1) < 0)
    perror("blad odlokowania semafora");
}


main()
{
int semid = -1; // identyfikator semafora
int co;
int jeszcze;
if  ((semid = semget(ftok("/tmp",0), 1, IPC_CREAT | PERMS)) < 0)
    perror("blad tworzenia semafora");
printf("Podaj Polecenie\n 1 - podnies semafor\n 2 - opusc semafor\n 0 -
wyjscie\n");
for (jeszcze = 1;jeszcze;)
    {
    scanf("%d",&co);
    printf("wybrano %d\n",co);
    switch(co)
        {
        case 2:
            {
            printf("przed blokuj\n");
            blokuj(semid);
            printf("po blokuj\n");
            break;
            }
        case 1:
            {
            printf("przed odblokuj\n");
            odblokuj(semid);
            printf("po odblokuj\n");
            break;
            }
        case 0:
            {
            jeszcze = 0;
            break;
            }
        default:
            {
            printf("nie rozpoznana komenda %d\n",co);
            }
        }
    }
}

student@st06-lab319:~$ ./l6z3
Podaj Polecenie
 1 - podnies semafor
```

```
 2 - opusc semafor
 0 - wyjscie
1
wybrano 1
przed odblokuj
po odblokuj

0
wybrano 0
student@st06-lab319:~$ ./l6z3
Podaj Polecenie
 1 - podnies semafor
 2 - opusc semafor
 0 - wyjscie
1
wybrano 1
przed odblokuj
po odblokuj
1
wybrano 1
przed odblokuj
po odblokuj
0
wybrano 0
student@st06-lab319:~$ ./l6z3
Podaj Polecenie
 1 - podnies semafor
 2 - opusc semafor
 0 - wyjscie
1
wybrano 1
przed odblokuj
po odblokuj
1
wybrano 1
przed odblokuj
po odblokuj
2
wybrano 2
przed blokuj
po blokuj
2
wybrano 2
przed blokuj
po blokuj
2
wybrano 2
przed blokuj
po blokuj
2
wybrano 2
przed blokuj
po blokuj
2
wybrano 2
przed blokuj
po blokuj
2
```

wybrano 2
przed blokuj

Co się dzieje gdy wykonamy operacje V (podniesienie) za pierwszym razem? Dodano wartość 1 do semafora
Co się stanie gdy wykonamy operacje V za drugim razem? Ponownie dodano 1 do semafora
Ile razy możemy teraz wykonać operację P (opuszczenie)? Możemy opuścić 6 razy, następnie dochodzi do zablokowania
Otworzyć drugą konsolę i uruchomić nasz program, podnieść semafor, co się stało na pierwszym terminalu? Na pierwszym terminalu doszło do odblokowania
Opuścić semafor w pierwszym terminalu, co się stało? W pierwszym terminalu doszło do zablokowania, w drugim nie
Otworzyć trzeci terminal i podnieść semafor, czy na obu terminalach doszło do odblokowania? Pierwszy terminal został odblokowany
Na trzecim terminalu jeszcze raz podnieść semafor. Ponowne podniesienie nie zmieniło niczego (poza podniesieniem wartosci semafora), bo terminale byly odblokowane
Gdy wszystkie terminale odblokowane wyjść przez 0.
0
Komendą ipcs sprawdzić czy semafor został w systemie.
student@st06-lab319:~$ ipcs

```
------ Segmenty pamięci dzielonej ----
klucz      id_shm     właściciel uprawn.    bajtów    podłączeń stan
0x00000000 1048576    student    600        393216     2          dest
0x00000000 393217     student    600        524288     2          dest
0x00000000 425986     student    600        524288     2          dest
0x00000000 688131     student    600        524288     2          dest
0x00000000 786436     student    600        524288     2          dest
0x00000000 1179653    student    600        524288     2          dest
0x00000000 917510     student    600        33554432   2          dest
0x00000000 1015815    student    600        524288     2          dest
0x00000000 1146888    student    600        1048576    2          dest
0x00000000 1277961    student    600        524288     2          dest
0x00000000 1376266    student    600        524288     2          dest
0x00000000 1474571    student    600        524288     2          dest
0x00000000 1507340    student    600        2097152    2          dest

------ Tablice semaforów -------
klucz      id_sem     właściciel uprawn.    lsem
0x00080001 0          student    666        1

------ Kolejki komunikatów ---
klucz      id_msq     właściciel uprawn.    bajtów      komunikatów
```

Komendą ipcrm -s usunąć semafor z systemu

```
student@st06-lab319:~$ ipcrm -s
ipcrm: opcja musi mieć argument -- 's'
Składnia: ipcrm [ [-q id_msg] [-m id_shm] [-s id_sem]
             [-Q klucz_msg] [-M klucz_shm] [-S klucz_sem] ... ]
student@st06-lab319:~$ ipcrm -s 0
student@st06-lab319:~$ ipcs

------ Segmenty pamięci dzielonej ----
klucz      id_shm     właściciel uprawn.    bajtów    podłączeń stan
0x00000000 1048576    student    600        393216     2          dest
```

```
0x00000000 393217     student    600        524288     2          dest
0x00000000 425986     student    600        524288     2          dest
0x00000000 688131     student    600        524288     2          dest
0x00000000 786436     student    600        524288     2          dest
0x00000000 1179653    student    600        524288     2          dest
0x00000000 917510     student    600        33554432   2          dest
0x00000000 1015815    student    600        524288     2          dest
0x00000000 1146888    student    600        1048576    2          dest
0x00000000 1277961    student    600        524288     2          dest
0x00000000 1376266    student    600        524288     2          dest
0x00000000 1474571    student    600        524288     2          dest
0x00000000 1507340    student    600        2097152    2          dest

------ Tablice semaforów -------
klucz      id_sem     właściciel uprawn.    lsem

------ Kolejki komunikatów ---
klucz      id_msq     właściciel uprawn.    bajtów      komunikatów
```

zmodyfikować tablicę operacji tak by operacja opuszczenia semafora była nie blokująca (IPC_NOWAIT)

```
static struct sembuf op_lock[1] = {
    0, -1, IPC_NOWAIT
};
```

Nie usuwać semafora, uruchomić program jeszcze raz tym razem próbując opuścić semafor. Czy stan semafora został zapamiętany?

Tak, został zapamiętany

Wprowadzić modyfikacje do tablic operacji polegającą na dodaniu opcji SEM_UNDO

```
static struct sembuf op_lock[1] = {
    0, -1, SEM_UNDO
};

static struct sembuf op_unlock[1] = {
    0, 1, SEM_UNDO
};
```

Uruchomić i spróbować opuścić semafor. Czy udało nam się zapamiętać ten stan?

Nieudało się zapamiętać

Napisać osobny "programik" automatycznie usuwający poprzednio stworzony semafor

```
#include <stdio.h>
#include <sys/types.h>
#include <sys/ipc.h>
#include <sys/sem.h>
#include <stdlib.h>
#include <unistd.h>

#define PERMS 0666
```

```
int main()
{
        int semid = -1; // identyfikator semafora

        if  ((semid = semget(ftok("/tmp",0), 1, PERMS)) < 0)
                perror("blad tworzenia semafora");

        if(semctl(semid, 0, IPC_RMID,0)<0){
                perror("blad usuwania semafora");
        }

        return 0;
}
```

Wnioski

Na tych zajęciach laboratoryjnych poznałem zasadę działania semaforów i mogłem zaobserwować ich tworzenie, pracę nad nimi i usuwanie. Najwięcej problemów sprawiło mi zmodyfikowanie programu tak, by odwrotne semafory działały. Niestety nie udało mi się tego zrobić.