Systemy operacyjne – studia stacjonarne 2022/23
Lab 8. Synchronizacja procesów przy pomocy semaforów.

Semafory to struktury danych, które mogą być używane przez wiele procesów do komunikacji (synchronizacji). Semafor przyjmuje wartości całkowite, nieujemne. Dostęp do semaforów jest możliwy poprzez funkcje systemowe semget(), semctl(), semop(). Na semaforze można wykonywać operacje semaforowe: podniesienie semafora (V - signal) i opuszczenie semafora (P - wait).
Do wywołania funkcji niezbędne są następujące pliki nagłówkowe

<sys/types.h>
<sys./ipc.h>
<sys/sem.h>

1. Zbiory semaforów, klucz do zbioru.
   Można utworzyć zbiory semaforów (np. pies: 1-100 semaforów w zbiorze). W każdym zbiorze semafory są ponumerowane kolejnymi liczbami całkowitymi nieujemnymi (tzn. 0, 1, 2, ….,n). Operacje jednoczesne można wykonywać tylko na semaforach należących do tego samego zbioru. Do tworzenia, czy do odwoływania się do zbioru semaforów potrzebny jest tzw. klucz – liczba całkowita.
   Jednoznaczne klucze można utworzyć przy pomocy funkcji ftok().

pliki nagłówkowe	<sys/types.h>, <sys/ipc.h>		
prototyp	key_t ftok(const char *path, int id)		
zwracana wartość	sukces	porażka	zmiana errno
	wartość klucza	-1	nie

   path - ścieżkowa nazwa istniejącego pliku - dostępnego procesowi
   id – zwykle pojedynczy znak, który jednoznacznie identyfikuje projekt

2. Tworzenie i uzyskiwanie dostępu do semaforów: semget().

pliki nagłówkowe	<sys/types.h>, <sys/ipc.h>, <sys/sem.h>		
prototyp	int semget (key_t key, int nsems, int semflg)		
zwracana wartość	sukces	porażka	zmiana errno
	identyfikator semafora	-1	tak

   key - klucz do zbioru semaforów (różne procesy, które chcą korzystać z tego samego zbioru, muszą użyć tego samego klucza)
   nsems – liczba semaforów w zbiorze
   semflag – flaga określająca sposób wykonania funkcji i prawa dostępu do semaforów:
   IPC_CREAT – utworzenie zbioru semaforów lub uzyskanie dostępu do istniejącego już zbioru
   IPC_EXCL – użyta w połączeniu z IPC_CREAT zwraca błąd, jeżeli dla danego klucza istnieje już zbiór semaforów
   PRAWA DOSTĘPU – podobnie jak dla pliku np. 0666
   Przykład: utworzenie zbioru 3 semaforów dla klucza key i przypisanie zmiennej semid identyfikatora tego zbioru.

```
key_t key;
int semid;
key = ftok(".", 'A');
semid=semget(key,3,IPC_CREAT|0666);
```

3. Sterowanie semaforami: funkcja semctl().

pliki nagłówkowe	<sys/types.h>, <sys/ipc.h>, <sys/sem.h>		
prototyp	semctl(int semid, int semnum, int cmd, /* union semun arg */,…)		
zwracana wartość	sukces	porażka	zmiana errno
	0 lub wartość żądana przez cmd	-1	tak

semid – identyfikator zbioru semaforów (zwracany przez semget())
semnum – numer semafora w zbiorze na którym ma by ć wykonana operacja
cmd – kod polecenia
arg – parametry polecenia lub wartości przekazywane przez semctl();

Wybrane polecenia cmd:
SETVAL – zainicjowanie semafora o numerze semnum
SETALL – zainicjowanie wszystkich semaforów
GETVAL- odczytanie wartości semafora o numerze semnum
GETNCNT – odczytanie liczby procesów czekających na podniesienie semafora
GETZCNT - – odczytanie liczby procesów czekających na opuszczenie semafora
IPC_RMID – usunięcie danego zbioru semaforów.

4. Operacje na semaforach: funkcja semop().

pliki nagłówkowe	<sys/types.h>, <sys/ipc.h>, <sys/sem.h>		
prototyp	int semop(int semid, struct sembuf *sops, size_t nsops)		
zwracana wartość	sukces	porażka	zmiana errno
	0	-1	tak

semid – identyfikator zbioru semaforów
sops – wskaźnik do tablicy struktur określający operacje na zbiorze semaforów

```
struct sembuf {
    short sem_num; /* numer semafora */
    short sem_op; /* operacja semaforowa */
    short sem_flg; /* flaga */
    };
```

Wartości sem_flg:
IPC_NOWAIT - operacja nieblokująca, tzn. jeśli operacja nie może być wykonana natychmiast, to semop natychmiast się kończy (z wartością -1), a zmiennej 'errno' zostaje nadana wartość EAGAIN,
SEM_UNDO - jeśli operacja zmienia wartość semafora, zmiana ta zostanie zniwelowana w momencie zakończenia wykonywania procesu

nsops – liczba semaforów (elementów tablicy), na których ma być wykonana operacja

Dodatkowe informacje:
http://www.linux.pl/man/index.php?command=ftok
http://www.linux.pl/man/index.php?command=semget
http://www.linux.pl/man/index.php?command=semctl
http://www.linux.pl/man/index.php?command=semop
http://www.linux.pl/man/index.php?command=ipcs
http://www.linux.pl/man/index.php?command=ipcrm

5. Użyteczne komendy:

ipcs –s	podaje informacje nt. aktywnych semaforów (patrz man ipcs)
ipcrm –s semid	usuwa zbiór semaforów o numerze semid
ipcs -l	podaje informacje dotyczące limitów

Skopiuj do swojego katalogu domowego plik **~suwada.anna/SO/ABC.tar**
Jest to archiwum z plikami:
suwada.anna@torus:~/SO/ABC2023$ tar tvf ABC.tar
-rw-r--r-- suwada.anna/suwada.anna 300 2023-12-05 20:06 A.c
-rw-r--r-- suwada.anna/suwada.anna 1143 2023-12-05 20:07 mainprog.c
-rw-r--r-- suwada.anna/suwada.anna 1413 2023-12-05 20:16 operacje.c
-rw-r--r-- suwada.anna/suwada.anna 450 2023-12-05 20:21 operacje.h
-rw-r--r-- suwada.anna/suwada.anna 380 2021-12-12 19:22 Makefile

Przygotuj bibliotekę statyczną zawierającą definicje operacji semaforowych: biblioteka.a
1 - utwórz plik operacje.o: **gcc -c operacje.c**
2 - **ar crv biblioteka.a operacje.o**

Tak przygotowaną bibliotekę można wykorzystać do utworzenia plików wykonywalnych:
gcc A.c biblioteka.a -o A

Zadanie *(programy do przedstawienia na torusie prowadzącej zajęcia)*

Rozwiązać zadanie synchronizacji trzech procesów przy pomocy semaforów:
A: a1, a2, a3
B: b1, b2
C: c1, c2, c3
Kolejność czasowa zdefiniowana jest zbiorem par G={(b1, a1), (c2, b2), (c3, a2), (a3, b3)}. Jeżeli para (a,b) należy do zbioru, to akcja b musi być poprzedzona akcją a.

Każdy z tasków ai, bi, ci (dla i=1,2,3) wyprowadza na standardowe wyjście informacje:
- nazwa tasku
- PID procesu

Każdy z tasków na zakończenie wykonuje funkcję: `sleep(m)` `1<=m<=4;`
Napisz cztery programy:
mainp:
- inicjuje mechanizmy IPC (semafory),
- uruchamia trzy procesy potomne: A, B, C w oparciu o funkcje fork(), exec(),
- czeka na zakończenie procesów A, B, C,
- usuwa zestaw semaforów

A, B, C – realizują swoje zadania zgodnie z zadaną relacją pierwszeństwa

*Treści oznaczone kursywą pochodzą z różnych źródeł internetowych.