Semafory

- wykorzystanie semaforów zapobiega niedozwolonemu wykonaniu operacji na określonych danych jednocześnie przez większą liczbę procesów
- przez odpowiednie wykorzystywanie semaforów można zapobiec sytuacji w której wystąpi **zakleszczenie** (ang. ***deadlock*)** lub **zagłodzenie** (ang. **starvation**)
- **Semafor binarny**
  - będący tylko w stanie podniesienia lub opuszczenia
  - reprezentowany przez liczbę binarną $S \in \{0, 1\}$
- **Semafor uogólniony**
  - możliwe wiele stanów
  - reprezentowany przez liczbę binarną $S \in \{0, 1, ..., \infty\}$
- Pojęcie semafora pierwszy raz zdefiniowane przez holendra **Edgara Dijkstrę** – [skróty: **v** dla operacji podniesienia semafora i **p** dla opuszczenia semafora]
- Praktyczna definicja semafora uogólnionego wg. Ben-Ariego:

Semafor jest pewną całkowitą liczbą nieujemną S.

Opuszczenie semafora jest równoważne wykonaniu instrukcji:
- **jeśli S>0 to S=S-1,**
- **w przeciwnym razie wstrzymaj działanie procesu próbującego opuścić semafor**

Podniesienie semafora:
- **jeśli są procesy wstrzymane przy próbie opuszczenia semafora S to wznów jeden z nich,**
- **w przeciwnym wypadku S=S+1**

- Obszar programu składający się z instrukcji które może wykonywać tylko określona liczba procesów = **sekcja krytyczna**
- Operacje wykonywane na semaforze są **atomowe**.
- Semafor można traktować jako licznik, który jest zmniejszany („zamykany") o 1 gdy jest zajmowany i zwiększany o 1 gdy jest zwalniany („podnoszony")
- Semafor trzeba **zainicjować** wywołując operację podniesienia !!!
- Struktura opisująca obiekt będący semaforem: `semid_ds`
- W systemie Unix/Linux występują tzw. **zestawy semaforów**. Każdy semafor z tego zestawu posiada strukturę z nim związaną:

```
Struct sem{
   ushort semval  // wartość semafora
   pid_t sempid   // identyfikator procesu ostatnio wykonującego operację na semaforze
   ushort semncnt // liczba procesów, które czekają aż wartość semafora będzie zwiększona
   ushort semzcnt // liczba procesów które czekają aż semafor osiągnie wartość 0
}
```

Deklaracje funkcji znajdują się w plikach nagłówkowych :

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

Funkcja SEMGET

`int semget (key_t key, int nsems, int semflags)`

- tworzenie semafora

- funkcja odpowiada funkcjom **`msgget()`**, **`semget()`**
- *key* – wartość klucza (**`ftok(), IPC_PRIVATE`**)
- *nsems* – liczba semaforów w zestawie (ponumerowanych od 0) //można utworzyć zestaw składający się z pojedynczego semafora
- *semflags* – podaje dodatkowe opcje dla tworzonego zestawu semaforów (**`IPC_CREAT, IPC_EXCL`**)
- funkcja zwraca identyfikator zestawu semaforów lub –1 w przypadku błędu

Funkcja SEMCTL

`int semctl (int semid, int semnum ,int cmd, union semun ctl_arg)`

- funkcje kontrolujące pojedynczy semafor lub ich zestaw

- *semid* – identyfikator zestawu semaforów
- *semnum* – identyfikuje semafor
- *cmd* – podaje funkcję, może należeć do jednej z trzech grup :

  tradycyjne operacje IPC
  - **`IPC_STAT`** – zwraca wartości struktury *semid_ds* dla semafora (lub zestawu) o identyfikatorze *semid*, informacja jest umieszczana w strukturze wskazywanej przez 4 argument
  - **`IPC_SET`** – modyfikuje wartości struktury *semid_ds*
  - **`IPC_RMID`** – usuwa zestaw semaforów o identyfikatorze *semid* z systemu

  operacje na pojedynczym semaforze (dotyczą semafora określonego przez semnum)
  - **`GETVAL`** – zwraca wartość semafora (*semval*)
  - **`SETVAL`** – ustawia wartość semafora w strukturze
  - **`GETPID`** – zwraca wartość *sempid*
  - **`GETNCNT`** – zwraca *semcnt*
  - **`GETZCNT`** – zwraca *semzcnt*

  operacje na wszystkich semaforach:
  - **`GETALL`** – umieszcza wszystkie wartości semaforów w tablicy podanej jako 4 argument
  - **`SETALL`** – ustawia wszystkie wartości zgodnie z zawartością tablicy podanej jako 4 argument

- ostatni (4) argument jest unią.

```
union semun (
      int val;
      struct semid_ds *buf;
      unsigned short *array
)
```

.

Zastosowanie:

- ustawienie początkowej wartości semafora
- zbadanie zmiany właściciela semafora, praw dostępu do niego, czasu ostatniej zmiany , ilości procesów oczekujących na semafor i identyfikatora procesu ostatnio zmieniającego wartość semafora, itd.

Funkcja SEMOP

`int semop (int semid, stuct sembuf *sops, unsigned nsops)`

-zajmowanie i zwalnianie semafora

- zwraca 0 przy poprawnym wykonaniu, -1 w przypadku niepowodzenia
- *semid* - identyfikator zestawu semaforów otrzymany z funkcji **`semget()`**
- *sops* – tablica struktur `sembuf`

```
struct sembuf{
   short sem_num     //index semafora w zestawie ( numeracja od 0 )
   short sem_op      //operacja
   short sem_flg     //opcje operacji
}
```

- *nsops* – liczba struktur sembuf w tablicy
- jeśli jedna z operacji nie może być wykonana, żadna nie będzie wykonana. (wszystko albo nic)
- *sem_flg*:
  - IPC-NOWAIT – jeśli operacja na semaforze nie może być wykonana to przy ustawionej fladze następuje powrót do procesu, nie jest on blokowany
  - SEM_UNDO – pozwala na operację wycofania, gdy proces się zakończy. Dla każdego procesu utrzymywana jest wartość semadj, do tej wartości jest dodawane sem_op
- *sem_op* może przyjmować wartości:
  - >0 – zwiększenie wartości semafora – zwolnienie zasobu
  - <0 – zmniejszenie semafora – próba zajęcia lub otrzymanie zasobu
  - =0 – sprawdzenie, czy wartość semafora wynosi 0