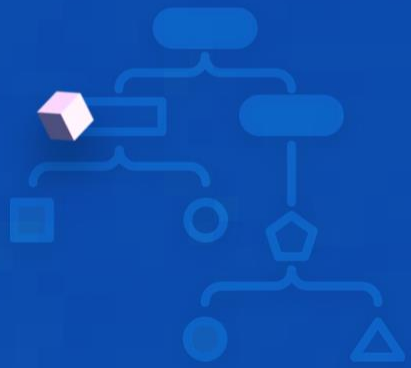
강원혁신플랫폼

리눅스프로그래밍

세마포어



Linux 시스템에서 공유 리소스에 대한 액세스를 제어하고 프로세스 간의 활동을 조정하는 데 사용되는 동기화 기법으로, 경쟁 조건을 피하고 상호 배제를 보장하기 위해 프로세스가 서로 조정하고 통신하는 메커니즘을 무엇이라고 하나요?

세마포어

학습목표

학습 내용

1 세마포어의 주요 함수

학습 목표

세마포어의 주요 함수를 파악할 수 있다.

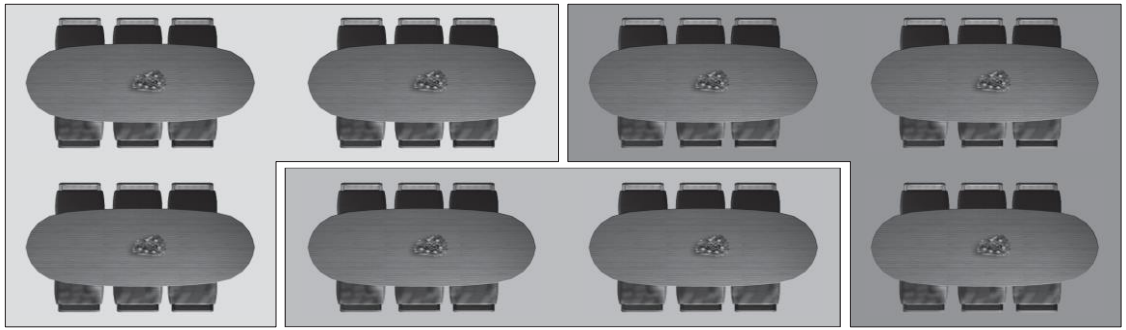
세마포어의 주요 함수

세마포어

특징

- 공유 리소스에 대한 액세스를 제어하고 프로세스 간의 활동을 조정하는 데 사용되는 동기화 프리미티브
- 경쟁 조건을 피하고 상호 배제를 보장하기 위해 프로세스가 서로 조정하고 통신하는 메커니즘 제공

유명 맛집의 대기번호



세마포어 semid_ds

- 세마포어 semid_ds 구조체를 사용

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

struct semid_ds{
        struct ipc_perm sem_perm;
        struct sem *sem_base;
        u_short sem_nsems;
        time_t sem_otime;
        time_t sem_ctime;
};
```

세마포어의 주요 함수

semget(): 새로운 세마포어를 생성하거나 기존 세마포어의 식별자를 얻는 데 사용

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

int semget(key_t key, int nsems, int semflg);
```

세마포어의 주요 함수

semop(): 대기(P) 및 신호(V) 작업과 같은 세마포어 작업을 수행하는 데 사용

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

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

세마포어의 주요 함수

semctl(): 세마포어를 제어하고 관리하는 데 사용

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

int semctl(int semid, int semnum, int cmd, ...);
```

세마포어의 주요 함수

- struct sembuf
  - 세마포어 연산을 나타냄
  - 세마포 인덱스, 작업 유형(대기의 경우 음수, 신호의 경우 양수) 및 선택적 플래그를 지정
- IPC_CREAT 및 IPC_EXCL
  - 세마포어의 생성 및 존재 확인을 지정하기 위해 semget()과 함께 사용할 수 있는 플래그 값
- union semun
  - 세마포어에서 특정 제어 작업을 수행할 때 semctl()의 인수로 사용

따라하기 sem.c


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

int cnt = 0;
static int semid;

void p() /* 세마포어의 P 연산 */
{
    struct sembuf pbuf;
    pbuf.sem_num = 0;
    pbuf.sem_op = -1;
```

```
pbuf.sem_flg = SEM_UNDO;

    /* 세마포어의 감소 연산을 수행한다. */
    if(semop(semid, &pbuf, 1) == -1)
        perror("p : semop()");
}

void v() /* 세마포어의 V 연산 */
{
    struct sembuf vbuf;
    vbuf.sem_num = 0;
    vbuf.sem_op = 1;
```

```
vbuf.sem_flg = SEM_UNDO;

    /* 세마포어의 증가 연산을 수행한다. */
    if(semop(semid, &vbuf, 1) == -1)
        perror("v : semop()");
}

int main(int argc, char **argv)
{
    union semun {/* semun 공용체 */
        int val;
        struct semid_ds *buf;
```

```
unsigned short int *arrary;
    } arg;
/* 세마포어에 대한 채널 얻기 */
    if((semid = semget(IPC_PRIVATE, 1, IPC_CREAT | 0666)) == -1) {
        perror("semget()");
        return -1;
    }
    arg.val = 1;      /* 세마포어 값을 1로 설정 */
    if(semctl(semid, 0, SETVAL, arg) == -1) {
        perror("semctl() : SETVAL");
        return -1;
    }
```

```
while(1) {
        if(cnt >= 8) {
            cnt--;
            p();
            printf("decrease : %d\n", cnt);
            break;
        } else {
            cnt++;
            v();
            printf("increase : %d\n", cnt);
            usleep(100);
```

```
}
    }

    /* 세마포어에 대한 채널 삭제 */
    if(semctl(semid, 0, IPC_RMID, arg) == -1) {
        perror("semctl() : IPC_RMID");
        return -1;
    }

    return 0;
}
```

따라하기 sem.c

실행결과

```
$ ./sem
increase : 1
increase : 2
increase : 3
increase : 4
increase : 5
increase : 6
increase : 7
increase : 8
decrease : 7
$
```

01 • 세마포어의 주요 함수

<sys/sem.h>
semget( )
semctl( )
semop( )