Readers Writers Semaphores

Implementation :

When reader(s) is/are reading or a writer is writing a writer can't write.
Similarly, when a writer is writing a reader can't read nor does a writer can write.

A single FIFO waitqueue of readers and writers is maintained.

If at a given time there are no readers and writers having access to data following things will happen:

a) If at the head of the queue a writer is present, then atmost one writer is allowed to access the data.
b) If at the head of the queue a reader is present, then the whole contiguous block of readers starting at the head are allowed to access the data.

Thus more than one readers can read at a time but more than one writers can not write at a time.

Data Structures Used :

1) Reader Writer Semaphore :

```
typedef struct rwsem {
    char * name;                  //name of semaphore
    int runread, runwrite;        //number of readers/writers currently having access to data
    Qelt * waitqueue;             //FIFO queue of requests
    Qelt * usebegcallqueue;       // list of processes which have previously called
                                  //rw_sem_use_begin on this semaphore
    struct rwsem * next;          //pointer to next semaphore
} rw_sem;
```

2) Qelt :

```
typedef struct qelt {
    pid_t pid;              //process id
    int type;               //for reader type = 0 ; for writer type = 1;
    struct qelt * next;     //pointer to next Qelt
} Qelt;
```

Prekshu Ajmera 03d05006
Sanchit Kr. Garg 03d05005