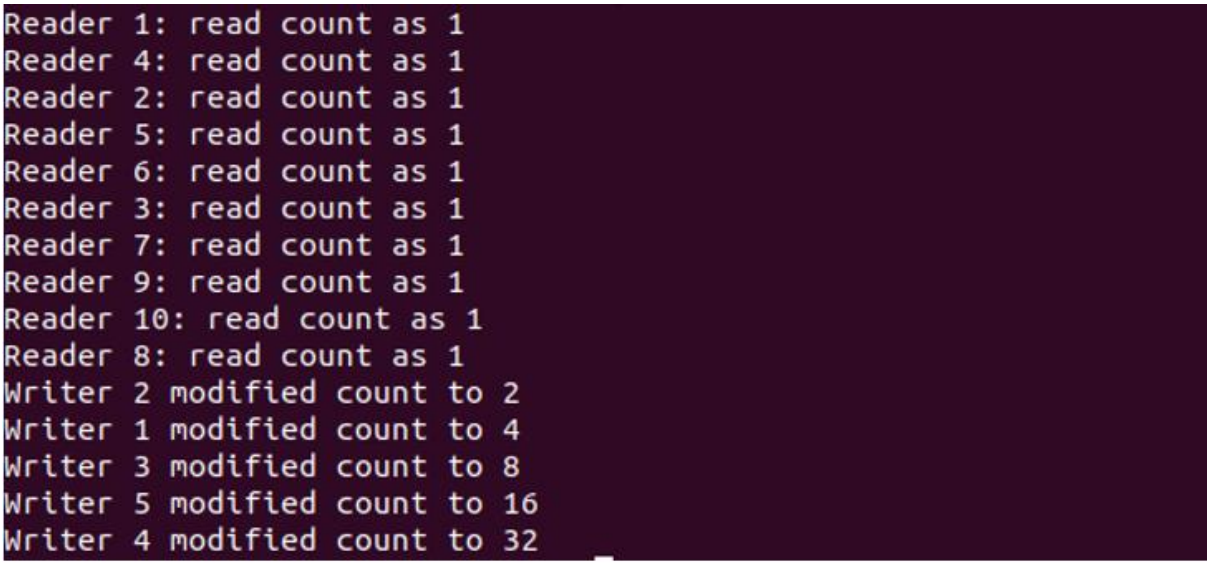
Reader Writer Problem

```
#include <pthread.h>
#include <semaphore.h>
#include <stdio.h>

sem_t w;
pthread_mutex_t mutex;
int count = 1;
int nr = 0;

void *writer(void *wn)
{
        sem_wait(&w);
        count = count*2;
        printf("Writer %d modified count to %d\n",(*((int *)wn)),count);
        sem_post(&w);

}
void *reader(void *rn)
{
        pthread_mutex_lock(&mutex);
        nr++;
        if(nr == 1) {
        sem_wait(&w);
        }
        pthread_mutex_unlock(&mutex);
        printf("Reader %d: read count as %d\n",*((int *)rn),count);
        pthread_mutex_lock(&mutex);
        nr--;
        if(nr == 0) {
        sem_post(&w);
        }
        pthread_mutex_unlock(&mutex);
}
```

```
int main()
{

        pthread_t read[10],write[5];
        pthread_mutex_init(&mutex,
        NULL);sem_init(&w,0,1);
        int i;
        int a[10] = {1, 2, 3, 4, 5, 6, 7, 8, 9, 10};
        for(i = 0; i < 10; i++) {
        pthread_create(&read[i], NULL, (void *)reader, (void *)&a[i]);
        }
        for(i = 0; i < 5; i++) {
        pthread_create(&write[i], NULL, (void *)writer, (void *)&a[i]);
        }

        for(i = 0; i < 10;
        i++) {
        pthread_join(read[i
        ], NULL);
        }
        for(i = 0; i < 5;
        i++) {
        pthread_join(write[
        i], NULL);
        }

        pthread_mutex_destroy(&mu
        tex);sem_destroy(&w);

        return 0;

}
```



LAB 11

Aim: 1. FIFO Page Replacement

2. Optimal Page Replacement

FIFO Page Replacement

Algorithm:

1. Start traversing the pages.
2. Declare the size as the length of the Page.
3. Check the need of the replacement from the page to memory.

4. Check the need of the replacement from the old page to the new page in memory.
5. Form the queue to hold all pages.
6. Insert the required page memory into the queue.
7. Check the bad replacements and page faults.
8. Get the number of processes to be inserted.
9. Show the values.

Code:

```
#include <stdio.h> int main()
{
int i, j = 0, nf, np, pg[50], frame[10], k, av, c = 0;
printf("Enter number of frames: ");
scanf("%d", &nf);
printf("Enter number of pages: "); scanf("%d", &np);
printf("Enter page reference string: "); for (i = 1; i <= np;
i++){
scanf("%d", &pg[i]);
}

for (i = 0; i < nf; i++){ frame[i] = -1;
}

printf("\n");
for (i = 1; i <= np; i++)
```

```
{
printf("%d\t\t", pg[i]); av = 0;
for (k = 0; k < nf; k++) if (frame[k] == pg[i])
av = 1; if (av == 0)
{
frame[j] = pg[i]; j = (j + 1) % nf; c++;
for (k = 0; k < nf; k++) printf("%d\t", frame[k]);
}
printf("\n");
}
printf("Page Fault: %d\n", c); return 0;
}
```