Lab Assignment 6: Semaphore

Name:Shivam Ganesh Gavandi
Roll no: 80
Class: TY-A

Reader writer using Mutex

```c
#include <stdio.h>
#include <pthread.h>
#include <string.h>
int hours = 23, mins = 59, secs = 53;
void update();
void display();
pthread_mutex_t timer_lock;
int main(void)
{
   void *status;
   pthread_t r_thr, w_thr;
   pthread_mutex_init(&timer_lock, 0);
   pthread_create(&r_thr, NULL, (void *)&display, (void *)NULL);
   pthread_create(&w_thr, NULL, (void *)&update, (void *)NULL);
   pthread_join(r_thr, &status);
   pthread_join(w_thr, &status);
}

void update()
{
```

```
   void *status;
   while (1)
   {
       pthread_mutex_lock(&timer_lock);
       secs = secs + 1;
       if (secs == 60)
       {
           mins = mins + 1;
           secs = 0;
       }
       if (mins == 60)
       {
           hours = hours + 1;
           mins = 0;
       }
       if (hours == 24)
       {
           hours = 0;
       }
       pthread_mutex_unlock(&timer_lock);
       sleep(1);
   }
   pthread_exit(&status);
}

void display()
{
   void *status;
   while (1)
```

```
    {
        pthread_mutex_lock(&timer_lock);
        printf("\n DISPLAY:");
        printf("\t %d %d %d", hours, mins, secs);
        pthread_mutex_unlock(&timer_lock);
        // sleep(1);
    }
    pthread_exit(&status);
}
```

Reader writer using semaphore

```
#include <stdio.h>
#include <pthread.h>
#include <semaphore.h>
#include <unistd.h>
sem_t r, w;
int h = 23, m = 59, s = 55;
void *reader(), *writer();
int main()
{
    pthread_t rth, wth;
    void *status;
    sem_init(&r, 0, 0);
    sem_init(&w, 0, 1);

    pthread_create(&rth, NULL, (void *)&reader, NULL);
    pthread_create(&wth, NULL, (void *)&writer, NULL);
```

```
    pthread_join(rth, status);
    pthread_join(wth, status);
    sem_destroy(&w);
    sem_destroy(&r);
}

void *writer()
{
    while (1)
    {
        sem_wait(&w);
        s = s + 1;
        if (s == 60)
        {
            m++;
            s = 0;
        }
        if (m == 60)
        {
            h++;
            m = 0;
        }
        if (h == 24)
        {
            h = 1;
        }
        // sleep(1);
        sem_post(&r);
    }
```

```
}


void *reader()
{
   while (1)
   {
        sem_wait(&r);
        printf("\n Display:\t");
        printf("%d:%d:%d", h, m, s);
        sem_post(&w);
   }
}
```

Producer consumer using Semaphore

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


pthread_t producer_thr;
pthread_t consumer_thr;
sem_t full;
sem_t empty;
sem_t mutex;
int buf[3], item_no = 0, buf_index = 0;


void *producer()
{
```

```
   int cntr;
   for (cntr = 0; cntr < 5; cntr++)
   {
       printf("Producer produced item %d\n", item_no);
       printf("Producer is checking if basket is having space\n");

       if (buf_index == 3)
           printf("Producer cannot insert as basket is full\n");

       sem_wait(&empty);

       sem_wait(&mutex);
       printf("Producer is inserting item %d in the basket\n",
item_no);
       buf[buf_index] = item_no;
       buf_index++;
       item_no++;
       sem_post(&mutex);

       sem_post(&full);
   }
}
void *consumer()
{
   int item, cntr;
   for (cntr = 0; cntr < 5; cntr++)
   {
       printf("\tConsumer is checking if buffer is having an item\n");
```

```
        if (buf_index == 0)
            printf("\tConsumer cannot consume as buffer is empty\n");

        sem_wait(&full);

        sem_wait(&mutex);
        item = buf[buf_index];
        printf("\tConsumer is removing item %d from the basket\n",
item);
        buf_index--;
        sleep(3);
        sem_post(&mutex);

        sem_post(&empty);
    }
}

void main()
{
    sem_init(&mutex, 0, 1);
    sem_init(&full, 0, 0);
    sem_init(&empty, 0, 3);

    pthread_create(&producer_thr, NULL, producer, NULL);
    pthread_create(&consumer_thr, NULL, consumer, NULL);

    pthread_join(producer_thr, NULL);
    pthread_join(consumer_thr, NULL);
}
```

Write a program to solve producer-consumer problem using Thread & mutexe

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

char buffer[20];
void *produce();
void *consume();
pthread_mutex_t mut;

int main()
{
   void *status;
   pthread_t p_thr, c_thr;
   pthread_mutex_init(&mut, 0);
   pthread_create(&p_thr, NULL, (void *)&produce, NULL);
   pthread_create(&c_thr, NULL, (void *)&consume, NULL);
   pthread_join(p_thr, &status);
   pthread_join(c_thr, &status);
   return 0;
}
```

```
void *produce()
{
   char str[20];

   while (1)
   {

       pthread_mutex_lock(&mut);
       printf("\nENTER A STRING:");
       scanf("%s", str);

       strcpy(buffer, str);
       pthread_mutex_unlock(&mut);
       sleep(1);
   }
}

void *consume()
{
   char str1[20];

   while (1)
   {
       pthread_mutex_lock(&mut);
       strcpy(str1, buffer);
       printf("\nTHE CONSUMED STRING IS :%s", str1);
       pthread_mutex_unlock(&mut);
       sleep(1);
```

```
    }

}
```