/\*

Title : Reader Writer

Problem Statement :Thread synchronization and mutual exclusion using mutex. Application to demonstrate: Reader-Writer problem.

\*/

#include <pthread.h>

#include <semaphore.h>

#include <stdio.h>

sem\_t wrt;

pthread\_mutex\_t mutex;

int cnt = 1;

int numreader = 0;

void \*writer(void \*wno)

{

sem\_wait(&wrt);

printf("\n Writer %d is Writing \n",(\*(int \*)wno));

cnt = cnt\*2;

printf("Writer %d modified data to %d\n",(\*((int \*)wno)),cnt);

sem\_post(&wrt);

}

void \*reader(void \*rno)

{

// Reader acquire the lock before modifying numreader

pthread\_mutex\_lock(&mutex);

numreader++;

if(numreader == 1) {

sem\_wait(&wrt); // If this id the first reader, then it will block the writer

}

pthread\_mutex\_unlock(&mutex);

// Reading Section

printf("\n Reader %d got the data section \n",(\*(int \*)rno));

printf("Reader %d: read data as %d\n",\*((int \*)rno),cnt);

// Reader acquire the lock before modifying numreader

pthread\_mutex\_lock(&mutex);

numreader--;

if(numreader == 0) {

sem\_post(&wrt); // If this is the last reader, it will wake up the writer.

}

pthread\_mutex\_unlock(&mutex);

}

int main()

{

pthread\_t read[10],write[5];

pthread\_mutex\_init(&mutex, NULL);

sem\_init(&wrt,0,1);

printf("\nEnter the Number of Reader : ");

int R;

scanf("%d",&R);

printf("\nEnter the Number of Writer : ");

int W;

scanf("%d",&W);

while(1)

{

for(int i = 0; i < R; i++) {

pthread\_create(&read[i], NULL, (void \*)reader, (void \*)&i);

}

for(int i = 0; i < W; i++) {

pthread\_create(&write[i], NULL, (void \*)writer, (void \*)&i);

}

for(int i = 0; i < 10; i++) {

pthread\_join(read[i], NULL);

}

for(int i = 0; i < 5; i++) {

pthread\_join(write[i], NULL);

}

}

pthread\_mutex\_destroy(&mutex);

sem\_destroy(&wrt);

return 0;

}