#include<stdio.h>

#include<unistd.h>

#include<pthread.h>

#include<semaphore.h>

#include<stdlib.h>

#define NUM\_READERS 3

#define NUM\_WRITERS 2

#define MAX\_OPERATIONS 10 // Define the maximum number of operations for readers and writers

sem\_t mutex, writeMutex;

int readCount = 0;

int totalReaderOperations = 0;

int totalWriterOperations = 0;

void \*reader(void \*arg)

{

while (totalReaderOperations < MAX\_OPERATIONS)

{

// Reader Entry Section

sem\_wait(&mutex);

readCount++;

if (readCount == 1)

{

sem\_wait(&writeMutex);

}

sem\_post(&mutex);

// Reading

printf("Reader %ld is reading...\n", (long)arg);

// Simulate reading by sleeping

sleep(2);

// Reader Exit Section

sem\_wait(&mutex);

readCount--;

if (readCount == 0)

{

sem\_post(&writeMutex);

}

sem\_post(&mutex);

// Increment the total reader operations

totalReaderOperations++;

}

pthread\_exit(NULL);

}

void \*writer(void \*arg)

{

while (totalWriterOperations < MAX\_OPERATIONS)

{

// Writer Entry Section

sem\_wait(&writeMutex);

// Writing

printf("Writer %ld is writing...\n", (long)arg);

// Simulate writing by sleeping

sleep(3);

// Writer Exit Section

sem\_post(&writeMutex);

// Increment the total writer operations

totalWriterOperations++;

}

pthread\_exit(NULL);

}

int main()

{

pthread\_t readers[NUM\_READERS], writers[NUM\_WRITERS];

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

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

for (long i = 0; i < NUM\_READERS; i++)

{

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

}

for (long i = 0; i < NUM\_WRITERS; i++)

{

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

}

for (int i = 0; i < NUM\_READERS; i++)

{

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

}

for (int i = 0; i < NUM\_WRITERS; i++)

{

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

}

return 0;

}

