CS304

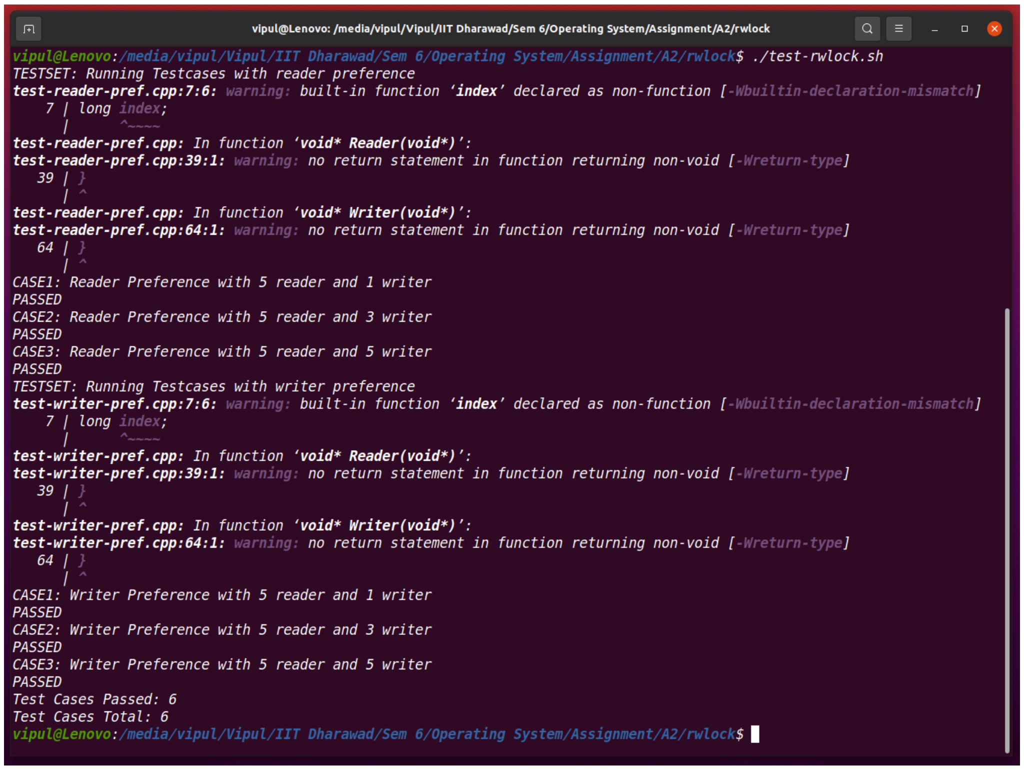
Assignment 2

Vipul Yuvraj Nikam

180010041

We had to write the structure that captures reader-writer lock in rwlock.h, then we had to complete the functions mentioned in the rwlock-reader-pref & rwlock-writer-pref files. Writer lock and unlock features remain unchanged for both, only the ReaderLock feature changed. For the reader's preference, we simply removed the waiting writer condition from the while condition, which was to check if any writer is waiting, then we don't allow any readers to enter. So we just remove that condition for the reader's preference.

#output:


vipul@Lenovo: /media/vipul/Vipul/IIT Dharawad/Sem 6/Operating System/Assignment/A2/rwlock

```
vipul@Lenovo:/media/vipul/Vipul/IIT Dharawad/Sem 6/Operating System/Assignment/A2/rwlock$ ./test-rwlock.sh
TESTSET: Running Testcases with reader preference
test-reader-pref.cpp:7:6: warning: built-in function ‘index’ declared as non-function [-Wbuiltin-declaration-mismatch]
    7 | long index;
      |      ^~~~~
test-reader-pref.cpp: In function ‘void* Reader(void*)’:
test-reader-pref.cpp:39:1: warning: no return statement in function returning non-void [-Wreturn-type]
   39 | }
      | ^
test-reader-pref.cpp: In function ‘void* Writer(void*)’:
test-reader-pref.cpp:64:1: warning: no return statement in function returning non-void [-Wreturn-type]
   64 | }
      | ^
CASE1: Reader Preference with 5 reader and 1 writer
PASSED
CASE2: Reader Preference with 5 reader and 3 writer
PASSED
CASE3: Reader Preference with 5 reader and 5 writer
PASSED
TESTSET: Running Testcases with writer preference
test-writer-pref.cpp:7:6: warning: built-in function ‘index’ declared as non-function [-Wbuiltin-declaration-mismatch]
    7 | long index;
      |      ^~~~~
test-writer-pref.cpp: In function ‘void* Reader(void*)’:
test-writer-pref.cpp:39:1: warning: no return statement in function returning non-void [-Wreturn-type]
   39 | }
      | ^
test-writer-pref.cpp: In function ‘void* Writer(void*)’:
test-writer-pref.cpp:64:1: warning: no return statement in function returning non-void [-Wreturn-type]
   64 | }
      | ^
CASE1: Writer Preference with 5 reader and 1 writer
PASSED
CASE2: Writer Preference with 5 reader and 3 writer
PASSED
CASE3: Writer Preference with 5 reader and 5 writer
PASSED
Test Cases Passed: 6
Test Cases Total: 6
vipul@Lenovo:/media/vipul/Vipul/IIT Dharawad/Sem 6/Operating System/Assignment/A2/rwlock$ 
```

#rwlock
```
struct read_write_lock
{
   int writersWaiting, readerCount;
   pthread_mutex_t lock;
   pthread_cond_t condVariable;
   int writerActive, fls = 0, tru = 1;
};
```

#rwlock-reader-pref & rwlock-writer-pref
```
void InitalizeReadWriteLock(struct read_write_lock * rw){
   //  Write the code for initializing your read-write lock.
   rw->readerCount = rw->writersWaiting = 0;
   rw->writerActive = rw->fls;
   rw->condVariable = PTHREAD_COND_INITIALIZER;
   rw->lock = PTHREAD_MUTEX_INITIALIZER;
}
```

#rwlock-writer-pref
```
void ReaderLock(struct read_write_lock * rw){
   //  Write the code for aquiring read-write lock by the reader.
   pthread_mutex_lock(&rw->lock);
   while (rw->writersWaiting > 0 || rw->writerActive != rw->fls){
       pthread_cond_wait(&rw->condVariable, &rw->lock);
   }
   rw->readerCount ++;
   pthread_mutex_unlock(&rw->lock);
}
```

#rwlock-reader-pref
```
void ReaderLock(struct read_write_lock * rw){
   //  Write the code for aquiring read-write lock by the reader.
   pthread_mutex_lock(&rw->lock);
   while (rw->writerActive != rw->fls){
       pthread_cond_wait(&rw->condVariable, &rw->lock);
   }
   rw->readerCount ++;
   pthread_mutex_unlock(&rw->lock);
}
```

#rwlock-reader-pref & rwlock-writer-pref

```c
void ReaderUnlock(struct read_write_lock * rw)
{
   //  Write the code for releasing read-write lock by the reader.
   pthread_mutex_lock(&rw->lock);
   rw->readerCount --;
   if (rw->readerCount == 0)
   {
       pthread_cond_broadcast(&rw->condVariable);
   }
   pthread_mutex_unlock(&rw->lock);
}
```

#rwlock-reader-pref & rwlock-writer-pref

```c
void WriterLock(struct read_write_lock * rw)
{
   //  Write the code for aquiring read-write lock by the writer.
   pthread_mutex_lock(&rw->lock);
   rw->writersWaiting ++;

   while (rw->readerCount > 0 || rw->writerActive != rw->fls)
   {
       pthread_cond_wait(&rw->condVariable, &rw->lock);
   }
   rw->writersWaiting --;
   rw->writerActive = rw->tru;
   pthread_mutex_unlock(&rw->lock);
}
```

#rwlock-reader-pref & rwlock-writer-pref

```c
void WriterUnlock(struct read_write_lock * rw)
{
   //  Write the code for releasing read-write lock by the writer.
   pthread_mutex_lock(&rw->lock);
   rw->writerActive = rw->fls;
   pthread_cond_broadcast(&rw->condVariable);
   pthread_mutex_unlock(&rw->lock);
}
```

XXX