**LAB 9 ASSIGNMENT**

**Name:** Birva Babaria

**Roll no.:** CE010

**ID:** 19CEUON064

**Aim:** Write the programs for the solution of Reader-Writer Problem.

**1). Write a program for ‘Readers have priority’.**

**CODE:**

//Readers having priority

#include<stdio.h>

#include<pthread.h>

#include<semaphore.h>

#include<unistd.h>

int count = 0;

int value = 0;

sem\_t x,wsem;

void \*reader(void \*arg)

{

while(1)

{

int n = (intptr\_t)(arg);

sem\_wait(&x);

count++;

if(count==1)

{

sem\_wait(&wsem);

}

sem\_post(&x);

//READUNIT START

printf("Reader %d is reading the value %d.\n",n,value);

//READUNIT END

sem\_wait(&x);

count--;

if(count==0)

{

sem\_post(&wsem);

}

sem\_post(&x);

sleep(1);

}

}

void \*writer()

{

while(1)

{

sem\_wait(&wsem);

//WRITEUNIT START

value++;

printf("Write is updating the value as %d.\n",value);

//WRITEUNIT END

sem\_post(&wsem);

sleep(1);

}

}

void main()

{

pthread\_t r[3],w;

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

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

for(int i=0;i<3;i++)

{

pthread\_create(&r[i],NULL,(void \*)reader,(void \*)(intptr\_t)i);

}

pthread\_create(&w,NULL,(void \*)writer,NULL);

pthread\_join(r[0],NULL);

pthread\_join(r[1],NULL);

pthread\_join(r[2],NULL);

pthread\_join(w,NULL);

sem\_destroy(&x);

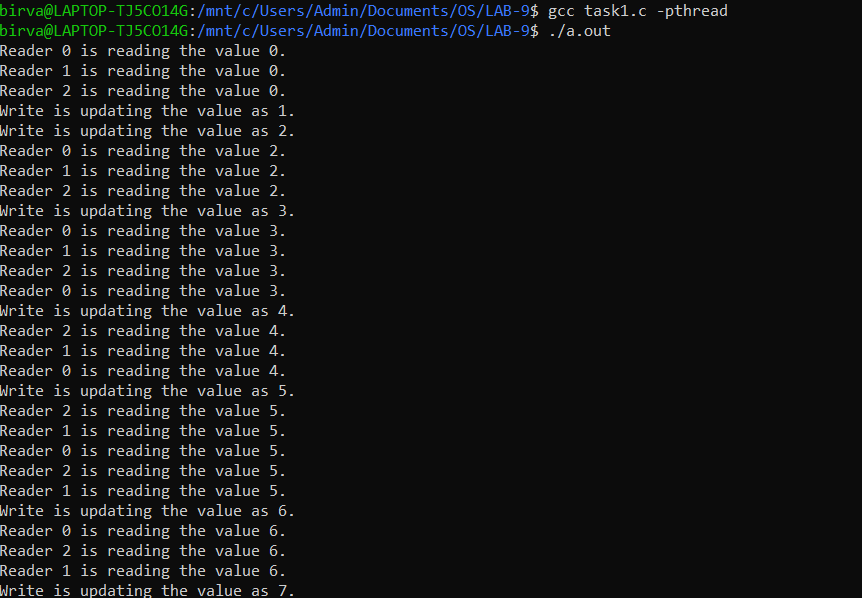
sem\_destroy(&wsem);

}

**OUTPUT:**

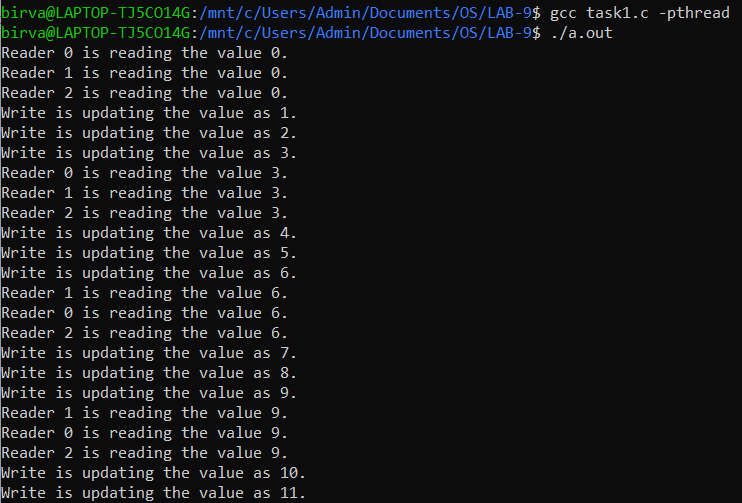
* Readers sleeping time: sleep(1)

Writers sleeping time: sleep(1)



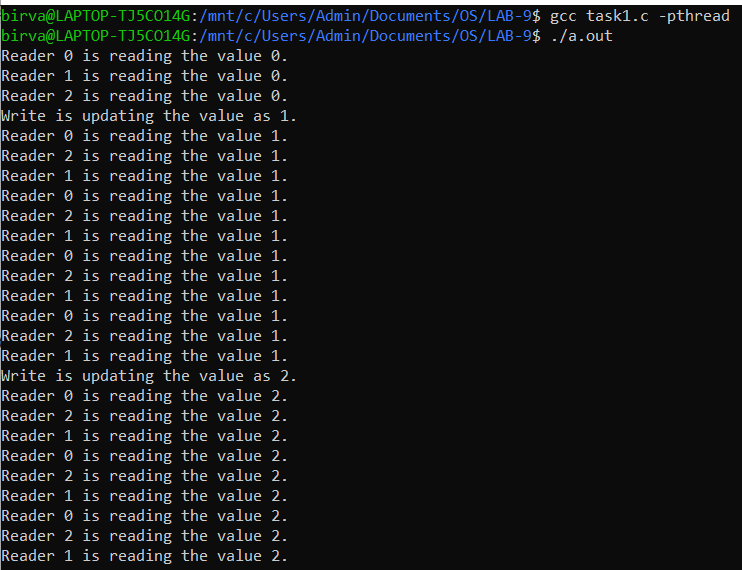
* Readers sleeping time: sleep(3)

Writers sleeping time: sleep(1)



* Readers sleeping time: sleep(1)

Writers sleeping time: sleep(5)



**2). Write a program for ‘Writers have priority’.**

**CODE:**

//Writers having priority

#include<stdio.h>

#include<pthread.h>

#include<semaphore.h>

#include<unistd.h>

#define max 3

sem\_t x, y, z, rsem, wsem;

int readcount, writecount;

void \*reader();

void \*writer();

int value = 1;

void \*reader(void \*arg)

{

while (1)

{

int n = (intptr\_t)arg;

sem\_wait(&z);

sem\_wait(&rsem);

sem\_wait(&x);

readcount++;

if (readcount == 1)

{

sem\_wait(&wsem);

}

sem\_post(&x);

sem\_post(&rsem);

sem\_post(&z);

printf("Reader %d is Reading the Value as %d.\n",n,value);

sem\_wait(&x);

readcount--;

if (readcount == 0)

{

sem\_post(&wsem);

}

sem\_post(&x);

sleep(1);

}

}

void \*writer(void \*arg)

{

while (1)

{

sem\_wait(&y);

writecount++;

if (writecount == 1)

{

sem\_wait(&rsem);

}

sem\_post(&y);

sem\_wait(&wsem);

value++;

int n=(intptr\_t)arg;

printf("Writer %d has updated the Value to %d.\n",n,value);

sem\_post(&wsem);

sem\_wait(&y);

writecount--;

if (writecount == 0)

{

sem\_post(&rsem);

}

sem\_post(&y);

sleep(1);

}

}

void main()

{

pthread\_t read[max], write[max];

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

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

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

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

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

for (int i = 0; i < max; i++)

{

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

}

for (int i=0; i<max; i++)

{

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

}

for (int i=0; i<max; i++)

{

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

}

for (int i=0; i<max; i++)

{

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

}

sem\_destroy(&x);

sem\_destroy(&y);

sem\_destroy(&z);

sem\_destroy(&rsem);

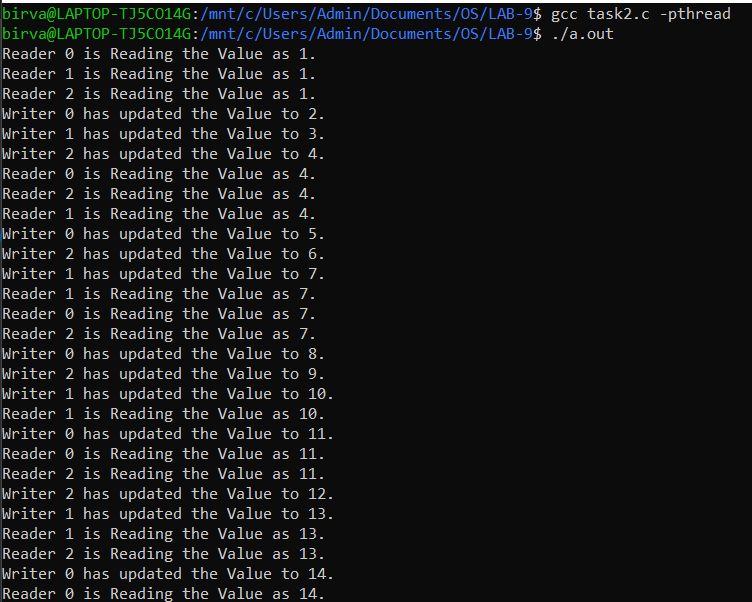
sem\_destroy(&wsem);

}

**OUTPUT:**

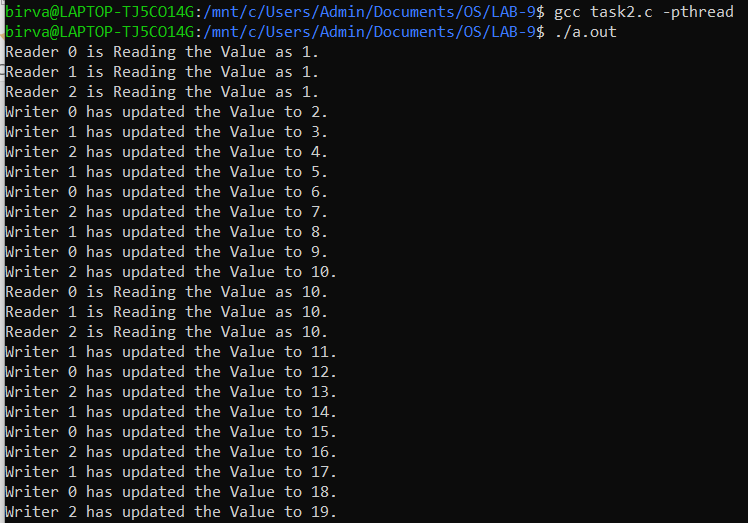
* Readers sleeping time: sleep(1)

Writers sleeping time: sleep(1)

****

* Readers sleeping time: sleep(3)

Writers sleeping time: sleep(1)



* Readers sleeping time: sleep(1)

Writers sleeping time: sleep(3)

