# Assignment – 4

Write a program using pthreads to demonstrate the reader writer synchronization problem. Implement appropriate synchronization. Show the different results with and without synchronization

Code :

#include <stdio.h>

#include <pthread.h>

#include <unistd.h>

#define NUM\_READERS 5

#define NUM\_WRITERS 5

#define MAX\_EXECUTIONS 5

int shared\_variable = 0;

pthread\_mutex\_t mutex = PTHREAD\_MUTEX\_INITIALIZER; pthread\_cond\_t readers\_condition = PTHREAD\_COND\_INITIALIZER; pthread\_cond\_t writers\_condition = PTHREAD\_COND\_INITIALIZER;

int reader\_executions = 0; int writer\_executions = 0;

void\* reader(void\* arg) {

int id = \*((int\*)arg);

while (reader\_executions < MAX\_EXECUTIONS)

{

printf("Reader %d reads: %d\n", id, shared\_variable); reader\_executions++;

sleep(1);

}

return NULL;

}

void\* writer(void\* arg) {

int id = \*((int\*)arg);

while (writer\_executions < MAX\_EXECUTIONS)

{

shared\_variable++; printf("Writer %d writes: %d\n", id, shared\_variable); writer\_executions++;

sleep(1);

}

return NULL;

}

int main() {

pthread\_t readers[NUM\_READERS]; pthread\_t writers[NUM\_WRITERS]; int reader\_ids[NUM\_READERS]; int writer\_ids[NUM\_WRITERS];

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

reader\_ids[i] = i + 1;

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

}

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

writer\_ids[i] = i + 1;

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

}

for (int i = 0; i < NUM\_READERS; i++) { pthread\_join(readers[i], NULL);

}

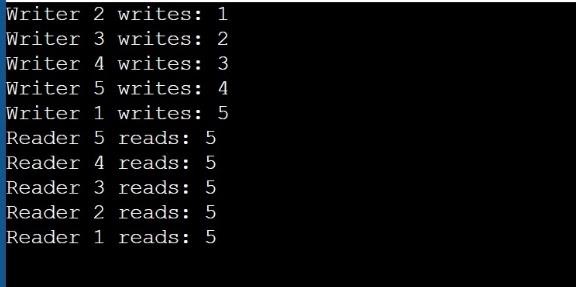
for (int i = 0; i < NUM\_WRITERS; i++) { pthread\_join(writers[i], NULL);

}

return 0;

}

Output :



Code :

#include <pthread.h>

#include <semaphore.h>

#include <stdio.h>

sem\_t wrt; pthread\_mutex\_t mutex; int balance = 1000; // Initial balance

void \*credit(void \*cno) { sem\_wait(&wrt);

balance += 100; // Credit operation

printf("Credit %d: New balance is %d\n", (\*((int \*)cno)), balance); sem\_post(&wrt);

}

void \*debit(void \*dno) { sem\_wait(&wrt);

balance -= 100; // Debit operation

printf("Debit %d: New balance is %d\n", (\*((int \*)dno)), balance); sem\_post(&wrt);

}

void \*reader(void \*rno) { pthread\_mutex\_lock(&mutex);

printf("Reader %d: Current balance is %d\n", (\*((int \*)rno)), balance);

pthread\_mutex\_unlock(&mutex);

}

int main() { pthread\_t readers[5], writers[5]; pthread\_mutex\_init(&mutex, NULL);

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

int rids[5] = {1, 2, 3, 4, 5}; // Reader IDs

int wids[5] = {1, 2, 3, 4, 5}; // Writer IDs

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

pthread\_create(&readers[i], NULL, reader, &rids[i]); pthread\_create(&writers[i], NULL, credit, &wids[i]);

}

for (int i = 0; i < 5; i++) { pthread\_join(readers[i], NULL); pthread\_join(writers[i], NULL);

}

pthread\_mutex\_destroy(&mutex); sem\_destroy(&wrt);

return 0; }

Output :

