**Assignment No. 05**

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

1. **Without Synchronization…**

**Code :**

#include <pthread.h>

#include <stdio.h>

#include <stdlib.h>

#include <unistd.h>

int total\_amount = 2500;

void \*add\_money(void \*x) {

int cash = 700;

total\_amount += cash;

printf("Writer %d credited Rs- %d. New balance: Rs- %d\n", \*((int \*)x), cash, total\_amount);

}

void \*deduct\_money(void \*x) {

int cash = 400;

total\_amount -= cash;

printf("Writer %d debited Rs- %d. New balance: Rs- %d\n", \*((int \*)x), cash, total\_amount);

}

void \*show\_balance(void \*x) {

printf("Reader %d checked balance: Rs- %d\n", \*((int \*)x), total\_amount);

}

int main() {

pthread\_t readers[6], writers[3];

char actions[] = {'D', 'C', 'D'};

int ids[6] = {1, 2, 3, 4, 5, 6};

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

pthread\_create(&readers[i], NULL, show\_balance, &ids[i]);

}

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

if(actions[i] == 'C')

pthread\_create(&writers[i], NULL, add\_money, &ids[i]);

else

pthread\_create(&writers[i], NULL, deduct\_money, &ids[i]);

}

for(int i = 3; i < 6; i++) {

pthread\_create(&readers[i], NULL, show\_balance, &ids[i]);

}

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

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

}

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

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

}

for(int i = 3; i < 6; i++) {

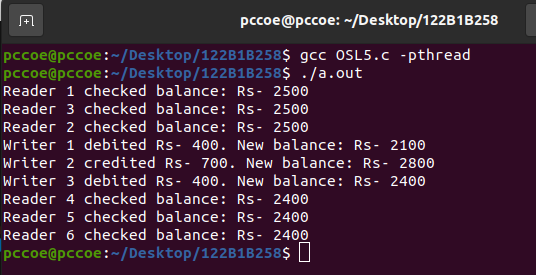
pthread\_join(readers[i], NULL);

}

return 0;

}

**Output :**

****

1. **With Synchronization…**

**Code :**

#include <pthread.h>

#include <semaphore.h>

#include <stdio.h>

sem\_t resource;

pthread\_mutex\_t lock;

int total\_amount = 2500;

int active\_readers = 0;

void \*add\_money(void \*x) {

int cash = 750;

sem\_wait(&resource);

total\_amount += cash;

printf("Writer %d credited Rs- %d. New balance: Rs- %d\n", \*((int \*)x), cash, total\_amount);

sem\_post(&resource);

}

void \*deduct\_money(void \*x) {

int cash = 250;

sem\_wait(&resource);

total\_amount -= cash;

printf("Writer %d debited Rs- %d. New balance: Rs- %d\n", \*((int \*)x), cash, total\_amount);

sem\_post(&resource);

}

void \*show\_balance(void \*x) {

pthread\_mutex\_lock(&lock);

active\_readers++;

if (active\_readers == 1) {

sem\_wait(&resource);

}

pthread\_mutex\_unlock(&lock);

printf("Reader %d checked balance: Rs- %d\n", \*((int \*)x), total\_amount);

pthread\_mutex\_lock(&lock);

active\_readers--;

if (active\_readers == 0) {

sem\_post(&resource);

}

pthread\_mutex\_unlock(&lock);

}

int main() {

pthread\_t readers[6], writers[3];

char actions[] = {'D', 'C', 'D'};

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

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

int ids[6] = {1, 2, 3, 4, 5, 6};

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

pthread\_create(&readers[i], NULL, show\_balance, &ids[i]);

}

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

if (actions[i] == 'C')

pthread\_create(&writers[i], NULL, add\_money, &ids[i]);

else

pthread\_create(&writers[i], NULL, deduct\_money, &ids[i]);

}

for (int i = 3; i < 6; i++) {

pthread\_create(&readers[i], NULL, show\_balance, &ids[i]);

}

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

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

}

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

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

}

for (int i = 3; i < 6; i++) {

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

}

pthread\_mutex\_destroy(&lock);

sem\_destroy(&resource);

return 0;

}

**Output :**

