praveenkumar.R
192324012

Experiment-19:Design a C program to implement process synchronization using mutex locks.

Aim:

The aim of this program is to demonstrate process synchronization using mutex locks. This program will have multiple threads accessing a shared resource, and mutex locks will be used to ensure mutual exclusion, preventing race conditions and ensuring that only one thread can access the shared resource at a time.

Procedure:

1. Initialize a mutex lock to synchronize access to the shared resource.
2. Create multiple threads that will perform operations on the shared resource.
3. Use the mutex lock to ensure that only one thread can access the resource at a time.
4. The threads will increment or modify the shared resource, demonstrating the mutual exclusion mechanism.
5. Each thread will wait for the mutex lock before accessing the shared resource and release the lock after the operation.

Code Implementation:

```
#include <stdio.h>

#include <pthread.h>

pthread_mutex_t lock;

int shared_resource = 0;

void* thread_function(void* param) {

   pthread_mutex_lock(&lock);

   shared_resource++;

   printf("Shared Resource: %d\n", shared_resource);

   pthread_mutex_unlock(&lock);

   return NULL;

}
```

```
int main() {
    pthread_t threads[5];
    pthread_mutex_init(&lock, NULL);
    for(int i = 0; i < 5; i++) {
        pthread_create(&threads[i], NULL, thread_function, NULL);
    }
    for(int i = 0; i < 5; i++) {
        pthread_join(threads[i], NULL);
    }
    pthread_mutex_destroy(&lock);
    return 0;
}
```

Output: