

**Ex.No:9**

## **Implementation of Threading**

**Date :**

**Aim:**

To write a C program to implement Threading & Synchronization

**Algorithm:**

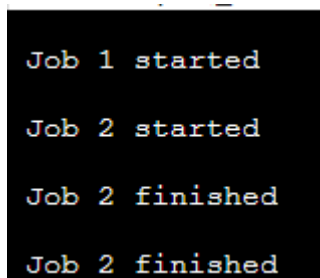
1. Start the Program
2. Initialize the process thread array.
3. Print the job started status.
4. Print the job finished status.
5. Start the main function
6. Check for the process creation if not print error message.
7. Stop the execution

**Program:**

```
#include<stdio.h>
#include<string.h>
#include<pthread.h>
#include<stdlib.h>
#include<unistd.h>
pthread_t tid[2];
int counter;
void* doSomething(void *arg)
{
    unsigned long i = 0;
    counter += 1;
    printf("\n Job %d started\n", counter);
    for(i=0; i<(0xFFFFFFFF);i++);
    printf("\n Job %d finished\n", counter);
    return NULL;
}
int main(void)
{
    int i = 0;
    int err;
    while(i < 2)
    {
        err = pthread_create(&(tid[i]), NULL, &doSomething, NULL);
```

```
if (err != 0)
printf("\ncan't create thread :[%s]", strerror(err));
i++;
}
pthread_join(tid[0], NULL);
pthread_join(tid[1], NULL);
return 0;
}
```

### Output:

A terminal window with a black background and white text. The output consists of four lines: 'Job 1 started', 'Job 2 started', 'Job 2 finished', and 'Job 2 finished'.

```
Job 1 started
Job 2 started
Job 2 finished
Job 2 finished
```

### Result: