DHRUV GARG 16 BCE 1190

OPENMP – SYNCHRONIZATION

LAB 2

Aim: To understand and implement the producer consumer problem using various synchronization constructs (like nowait, single, barrier and critical).

CODE AND OUTPUTS

1. PRODUCER CONSUMER PROBLEM

```c
#include <stdio.h>
#include <unistd.h>
#include <omp.h>
#include <time.h>

#define SIZE 4
#define NUM 26

char buffer[SIZE];
int in = 0;
int out = 0;
int count = 0;
int empty = 1;
int full = 0;
int i,j;
void put(char item)
{
    buffer[in] = item;
    in = (in + 1) % SIZE;
    count++;
    if (count == SIZE)
        full = 1;
    if (count == 1)
        empty = 0;
}

void producer(int tid)
{
    char item;
    while( i < NUM)
    {
        #pragma omp critical(CRIT)
        {
            item = 'A' + (i % 26);
            put(item);
            i++;
            printf("Thread: %d\tProducing\t%c\n",tid, item);
        }
        sleep(1);
```

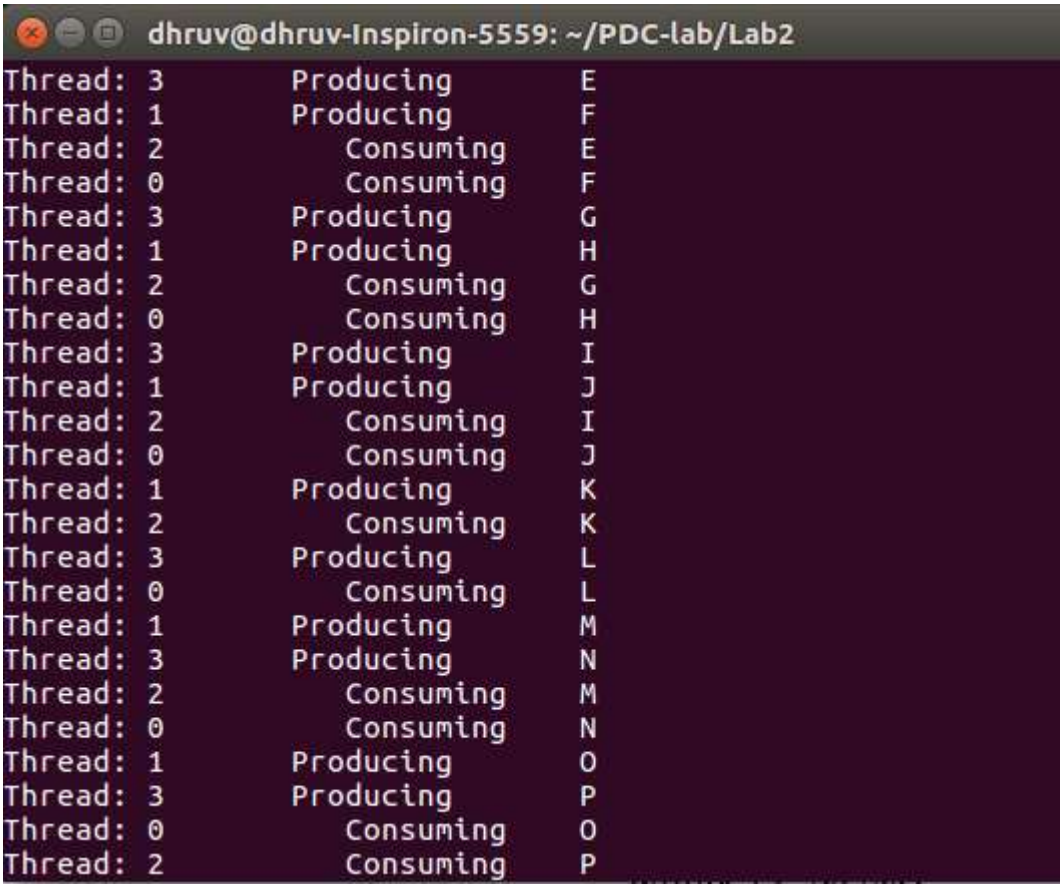
```
        }
        sleep(1);
    }
}

char get()
{
    char item;
    item = buffer[out];
    out = (out + 1) % SIZE;
    count--;
    if (count == 0)
        empty = 1;
    if (count == (SIZE-1))
        full = 0;
    return item;
}

void consumer(int tid)
{
    char item;
    while(j < NUM)
    {
        #pragma omp critical(CRIT)
        {
            j++;
            item = get();
            printf("Thread: %d\t   Consuming\t%c\n",tid, item);
        }
        sleep(1);
    }
}
```

```
int main()
{
    int tid;
    i=j=0;
    #pragma omp parallel firstprivate(i,j) private(tid)
    {
        tid=omp_get_thread_num();
        if(tid%2==1)
        {
            producer(tid);
        }
        else
        {
            consumer(tid);
        }
    }
}
```

OUTPUT



RESULT

Thus, the producer consumer problem was successfully implemented using the synchronization constructs in OpenMP.