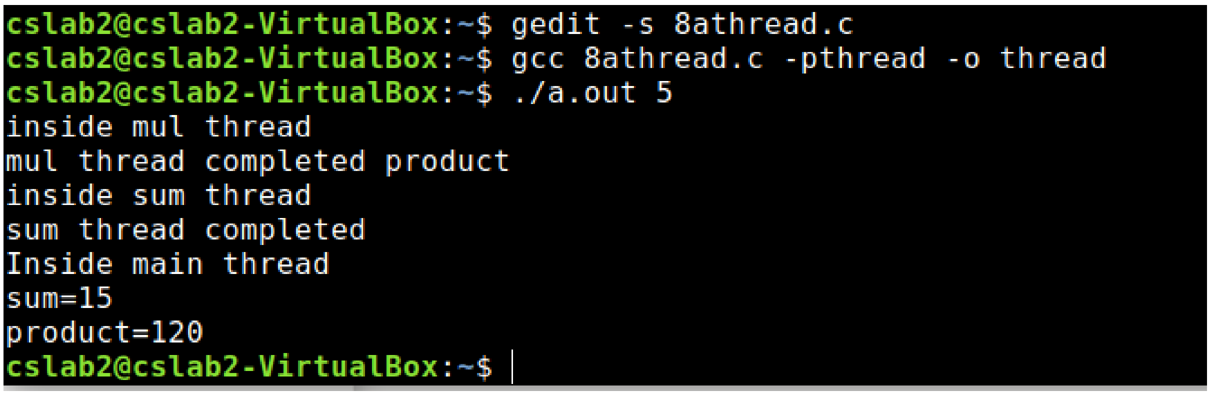
8. a. Write a program with two threads and a main thread. Schedule the task of calculating the natural sum upto ‘n’ terms and factorial of ‘n’ on these threads.
Note: The main thread should read ‘n’ from command line and pass it as parameter to remaining threads. Terminate the threads using system calls.

```
//gcc thread.c -pthread -o thread
#include<stdio.h>
#include<stdlib.h>
#include<pthread.h>
int sm=0, prod=1;
void *sum(void *parm)
{
int i, n;
n = atoi(parm);
printf("inside sum thread\n");

for(i=1; i<=n;i++)
{
sm+=i;
}
printf("sum thread completed\n");
}
void *fact(void *parm)
{
int i, n;
n = atoi(parm);
printf("inside mul thread\n");
for(i=2; i<=n;i++)
{
prod =prod *i;

}
printf("mul thread completed product\n");
}
void main(int argc, char * argv[])
{
pthread_t T1,T2;
pthread_attr_t attr;
pthread_attr_init(&attr);
pthread_create(&T1, &attr, sum, argv[1]);
pthread_create(&T2, &attr, fact, argv[1]);
pthread_join(T1,NULL);
pthread_join(T2,NULL);
printf("Inside main thread\n");
printf("sum=%d\n",sm);
printf("product=%d\n",prod);
}
```



```
cslab2@cslab2-VirtualBox:~$ gedit -s 8athread.c
cslab2@cslab2-VirtualBox:~$ gcc 8athread.c -pthread -o thread
cslab2@cslab2-VirtualBox:~$ ./a.out 5
inside mul thread
mul thread completed product
inside sum thread
sum thread completed
Inside main thread
sum=15
product=120
cslab2@cslab2-VirtualBox:~$ 
```