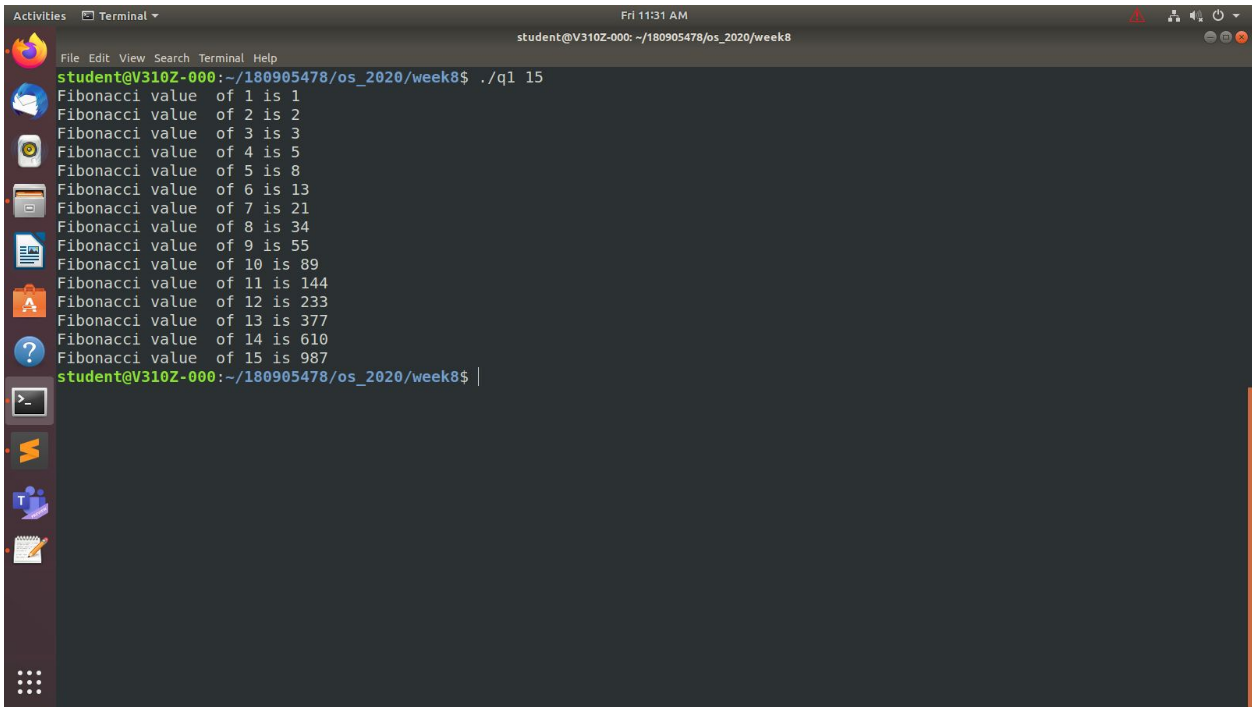
OS LAB8
NAME :SAGNIK CHATTERJEE
ROLL NO: 61
SEC : B
REG :180905478

Q1

```
#include <pthread.h>
#include <stdio.h>
#include <stdlib.h>

int prevsum;//shred value by the threads
void *runner(void *param)
{
   prevsum = fibonacci((int)param);
   pthread_exit(0);
}

int fibonacci (int x)
{
         if (x <= 1) {
         return 1;
         }
         return fibonacci(x-1) + fibonacci(x-2);
}
int main(int argc, char *argv[])
{
   int count, i;
   pthread_attr_t attr;

   if (argc != 2) {
         fprintf(stderr,"usage: pthreads <integer value>\n");
         exit(1);
   }

   count = atoi(argv[1]);

   if (count < 1) {
         fprintf(stderr,"%d must be>= 1\n", count);
         exit(1);
   }

   pthread_attr_init(&attr);
```

```
    for(i=1;i<=count;i++){
            pthread_t thread;
            pthread_create(&thread,&attr,runner,(void*)i);
            pthread_join(thread,NULL);
            printf("Fibonacci value  of %d is %d\n", i, prevsum);
    }
}
```

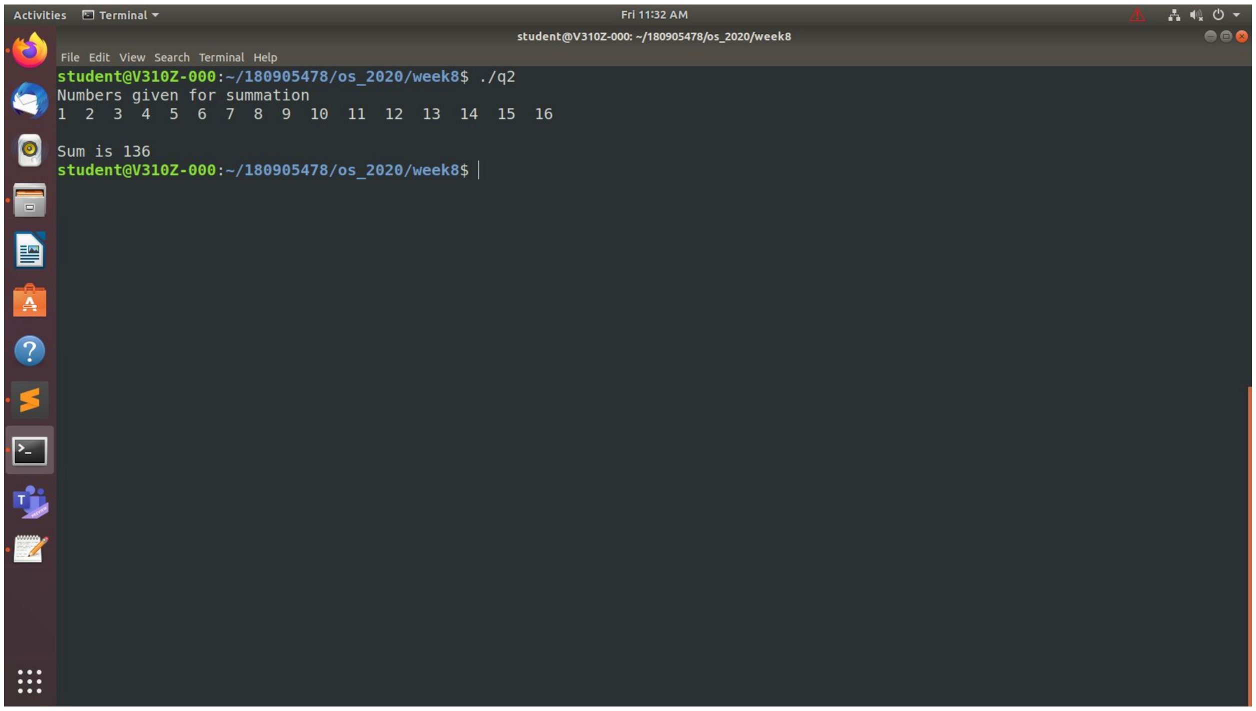


Activities
Terminal
Fri 11:31 AM
student@V310Z-000: ~/180905478/os_2020/week8
File Edit View Search Terminal Help

```
student@V310Z-000:~/180905478/os_2020/week8$ ./q1 15
Fibonacci value  of 1 is 1
Fibonacci value  of 2 is 2
Fibonacci value  of 3 is 3
Fibonacci value  of 4 is 5
Fibonacci value  of 5 is 8
Fibonacci value  of 6 is 13
Fibonacci value  of 7 is 21
Fibonacci value  of 8 is 34
Fibonacci value  of 9 is 55
Fibonacci value  of 10 is 89
Fibonacci value  of 11 is 144
Fibonacci value  of 12 is 233
Fibonacci value  of 13 is 377
Fibonacci value  of 14 is 610
Fibonacci value  of 15 is 987
student@V310Z-000:~/180905478/os_2020/week8$
```

Q2

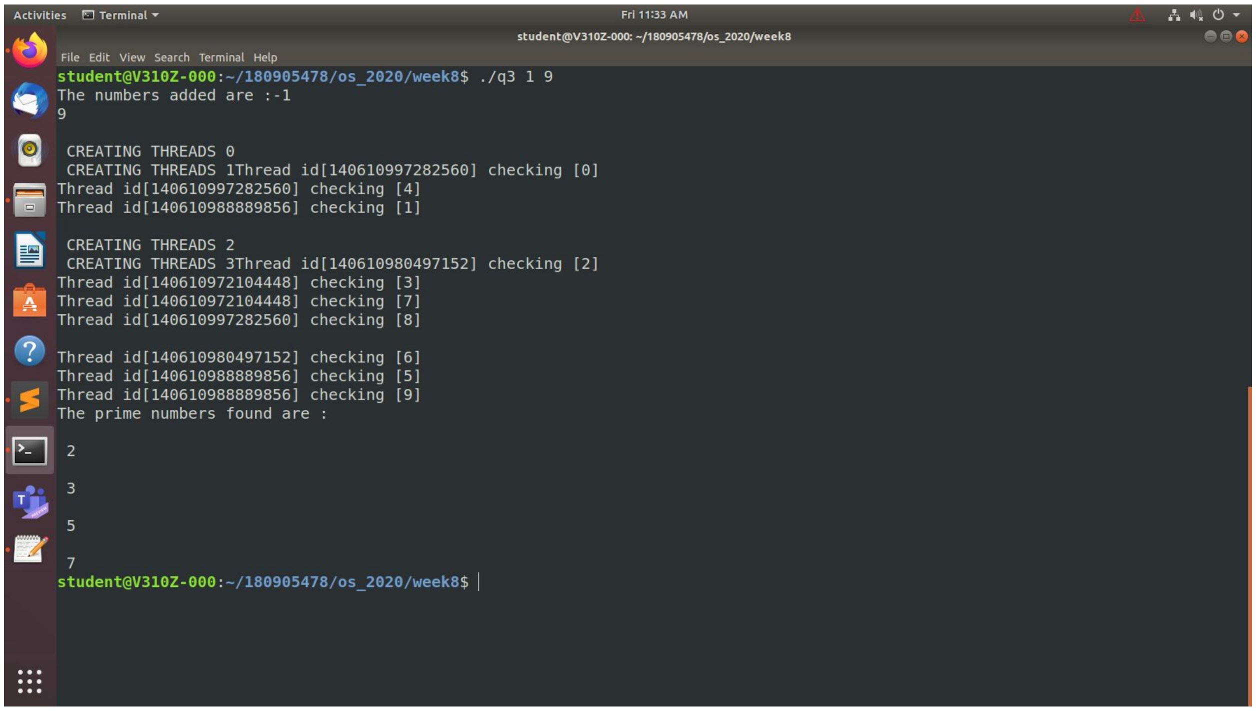
```
#include <stdio.h>
#include <pthread.h>
#include <stdlib.h>

#define max_threads 4
#define size 16

int arr[] = {1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16 };
int sum[]={0,0,0,0};
int part=0;

void * sum_array(){
    int thread_part =part++;
    //using 4 threads so diviing into 4 parts
    for(int i=thread_part * (size/4);i<(thread_part+1)*(size/4);i++){
            sum[thread_part]+=arr[i];
    }
}


int main(){
    printf("Numbers given for summation \n");
    for(int i=0;i<16;i++){
            printf("%d  ",arr[i]);
    }
    printf("\n");

    pthread_t threads[max_threads];

    //create the 4 threads
    for(int i=0;i<max_threads;i++){
            pthread_create(&threads[i],NULL,sum_array,(void*)NULL);
    }

    //wait for all threads to complete before joining
    for(int i=0;i<max_threads;i++){
            pthread_join(threads[i],NULL);
    }

    //adding sum of all 4 parts
    int total_sum=0;
    for(int i=0;i<max_threads;i++){
            total_sum+=sum[i];
    }
    printf("\nSum is %d\n",total_sum);
    return 0;

}
```


Activities
Terminal
Fri 11:32 AM
student@V310Z-000: ~/180905478/os_2020/week8
File Edit View Search Terminal Help

```
student@V310Z-000:~/180905478/os_2020/week8$ ./q2
Numbers given for summation
1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16

Sum is 136
student@V310Z-000:~/180905478/os_2020/week8$
```

Q3

```
#include<stdio.h>
#include<pthread.h>

#define N 10
#define MAX_THREADS 4

int prime_arr[N]={0};

void *printprime(void *ptr)
{
        int  j,flag;
        int i=(int)(long long int)ptr;
        while(i<N)
        {
        printf("Thread id[%ld] checking [%d]\n",pthread_self(),i);
        flag=0;
        for(j=2;j<=i/2;j++)
        {
        if(i%j==0)
        {
                flag=1;
                break;
        }
        }
        if(flag==0 && (i>1))
        {
        prime_arr[i]=1;
        }
        i+=MAX_THREADS;
  }
}

int main(int argc ,char **argv)
{
        pthread_t tid[MAX_THREADS]={{0}};
        printf("The numbers added are :-");
        int a =atoi(argv[1]);
        int b =atoi(argv[2]);
        printf("%d\n",a);
        printf("%d\n",b);
        int count=0;
        for(count=0;count<MAX_THREADS;count++)
        {
        printf("\r\n CREATING THREADS %d",count);
        pthread_create(&tid[count],NULL,printprime,(void*)count);
        }
        printf("\n");
        for(count=0;count<MAX_THREADS;count++)
        {
        pthread_join(tid[count],NULL);
        }

        int c=0;
        printf("The prime numbers found are :\n");
        for(count=a;count<=b;count++)
        if(prime_arr[count]==1)
        printf("\n %d \n",count);

        return 0;
 }
```

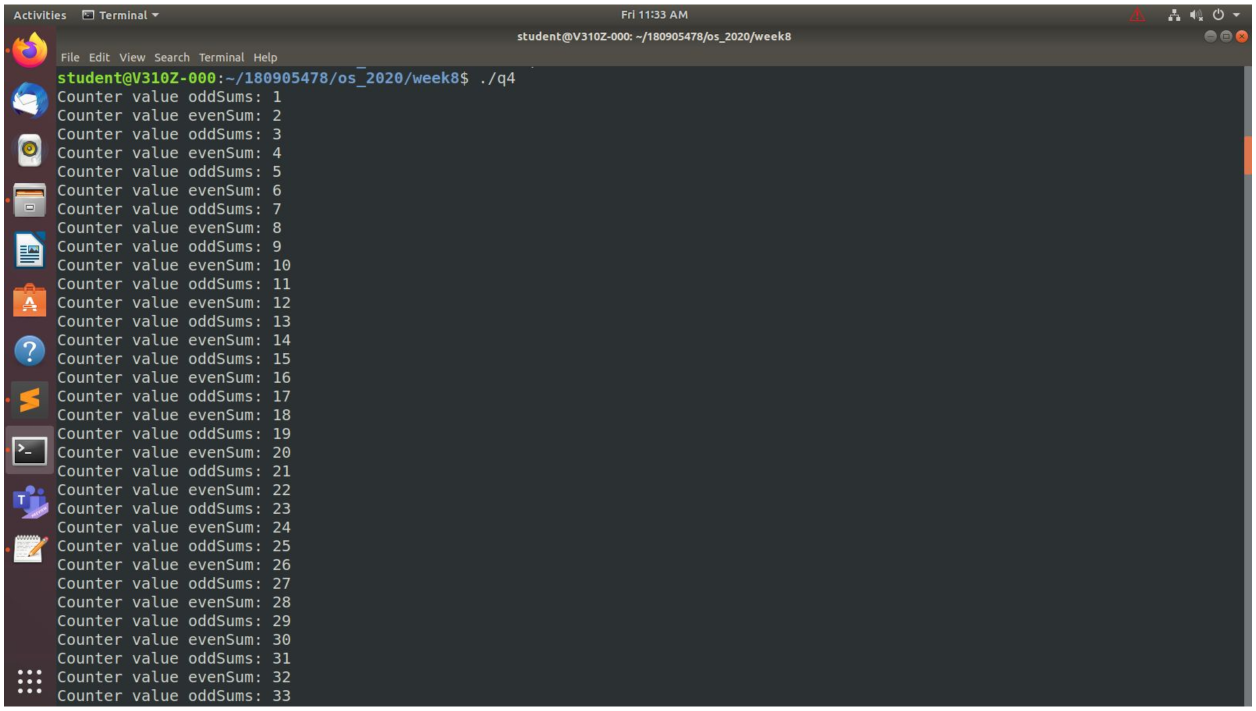


```
student@V310Z-000:~/180905478/os_2020/week8$ ./q3 1 9
The numbers added are :-1
9

 CREATING THREADS 0
 CREATING THREADS 1Thread id[140610997282560] checking [0]
Thread id[140610997282560] checking [4]
Thread id[140610988889856] checking [1]

 CREATING THREADS 2
 CREATING THREADS 3Thread id[140610980497152] checking [2]
Thread id[140610972104448] checking [3]
Thread id[140610972104448] checking [7]
Thread id[140610997282560] checking [8]

Thread id[140610980497152] checking [6]
Thread id[140610988889856] checking [5]
Thread id[140610988889856] checking [9]
The prime numbers found are :

 2

 3

 5

 7
student@V310Z-000:~/180905478/os_2020/week8$
```

Q4

```
#include <stdio.h>
#include <stdlib.h>
#include <pthread.h>

pthread_mutex_t count_mutex = PTHREAD_MUTEX_INITIALIZER;
pthread_cond_t condition_var = PTHREAD_COND_INITIALIZER;

int count = 0;

//checking for even and odd nums till this range
#define COUNT_DONE 500


// print odd numbers
void *oddNums(void* args)
{
   for(;;) {
          // Lock mutex and then wait for signal to relase mutex
          pthread_mutex_lock( &count_mutex );
          if ( count % 2 != 0 ) {
                   pthread_cond_wait( &condition_var, &count_mutex );
          }
          count++;
          printf("Counter value oddSums: %d\n",count);
```

```
            pthread_cond_signal( &condition_var );
            if ( count >= COUNT_DONE ) {
                        pthread_mutex_unlock( &count_mutex );
                        return(NULL);
            }
            pthread_mutex_unlock( &count_mutex );
    }
}

// print even numbers
void *evenNums(void* args)
{
    for(;;) {
            // Lock mutex and then wait for signal to release mutex
            pthread_mutex_lock( &count_mutex );
            if ( count % 2 == 0 ) {
                        pthread_cond_wait( &condition_var, &count_mutex );
            }

            count++;

            printf("Counter value evenSum: %d\n",count);
            pthread_cond_signal( &condition_var );
            if( count >= COUNT_DONE ) {
                        pthread_mutex_unlock( &count_mutex );
                        return(NULL);
            }
            pthread_mutex_unlock( &count_mutex );
    }
}

int main()
{
    pthread_t thread1, thread2;
    pthread_create(&thread1, NULL, oddNums, NULL);
    pthread_create(&thread2, NULL, evenNums, NULL);
    pthread_join(thread1, NULL);
    pthread_join(thread2, NULL);

    return 0;
}


//in the screenshot only shown till 33
```



```
student@V310Z-000:~/180905478/os_2020/week8$ ./q4
Counter value oddSums: 1
Counter value evenSum: 2
Counter value oddSums: 3
Counter value evenSum: 4
Counter value oddSums: 5
Counter value evenSum: 6
Counter value oddSums: 7
Counter value evenSum: 8
Counter value oddSums: 9
Counter value evenSum: 10
Counter value oddSums: 11
Counter value evenSum: 12
Counter value oddSums: 13
Counter value evenSum: 14
Counter value oddSums: 15
Counter value evenSum: 16
Counter value oddSums: 17
Counter value evenSum: 18
Counter value oddSums: 19
Counter value evenSum: 20
Counter value oddSums: 21
Counter value evenSum: 22
Counter value oddSums: 23
Counter value evenSum: 24
Counter value oddSums: 25
Counter value evenSum: 26
Counter value oddSums: 27
Counter value evenSum: 28
Counter value oddSums: 29
Counter value evenSum: 30
Counter value oddSums: 31
Counter value evenSum: 32
Counter value oddSums: 33
```