Lab6 – Threads

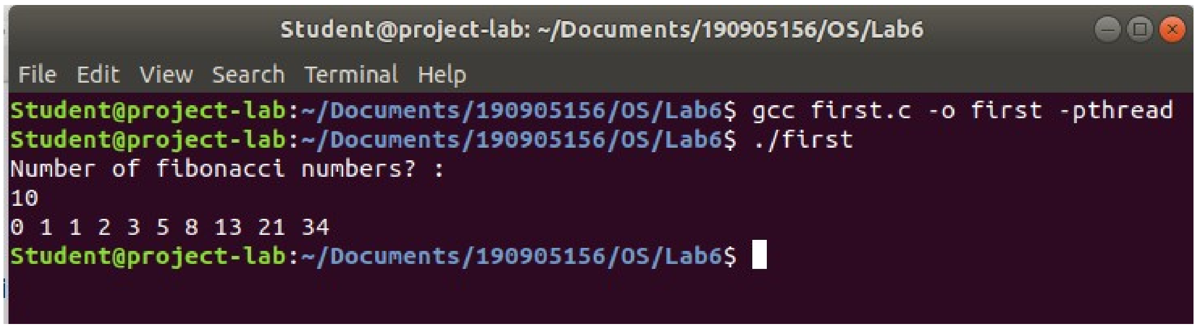
Name: Rhea Adhikari
Reg No: 190905156
Roll No: 23
Batch: D-1

Q1)
Write a multithreaded program that generates the Fibonacci series.

The program should work as follows:

The user will enter on the command line the number of Fibonacci numbers that the program is to generate. The program then will create a separate thread that will generate the Fibonacci numbers, placing the sequence in data that is shared by the threads (an array is probably the most convenient data structure). When the thread finishes execution, the parent will output the sequence generated by the child thread. Because the parent thread cannot begin outputting the Fibonacci sequence until the child thread finishes, this will require having the parent thread wait for the child thread to finish.

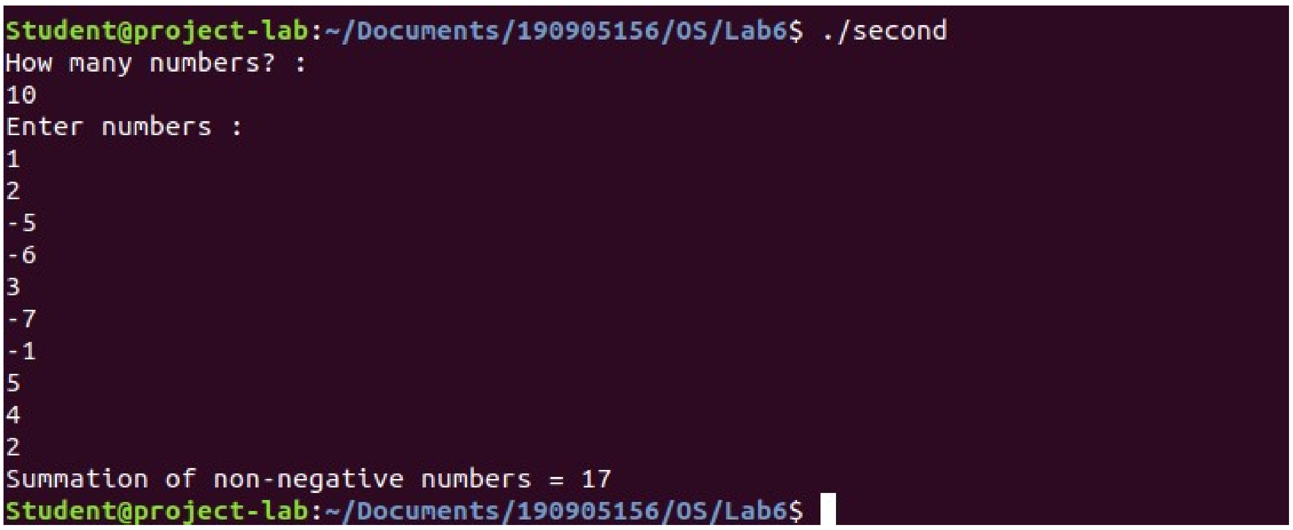
```
#include<stdio.h>
#include<stdlib.h>
#include<pthread.h>

void* generateFib(void* param)
{
  int* arr = (int*)param;
  int n = arr[0];
  arr[1] = 0;
  arr[2] = 1;

  for(int i = 3;i <= n;i++)
  {
    arr[i] = arr[i-1] + arr[i-2];
  }
  return NULL;
}

int main(int argc, char const *argv[])
{
  int n;
  printf("Number of fibonacci numbers? : \n");
  scanf("%d",&n);

  int* arr = (int*)malloc((n+1)*sizeof(int));
  arr[0] = n;

  pthread_t thread;
  pthread_create(&thread,0,&generateFib,(void*)arr);
  pthread_join(thread,0);

  for(int i = 1;i <= n;i++)
    printf("%d ",arr[i]);
  printf("\n");

  return 0;
}
```



```
Student@project-lab:~/Documents/190905156/OS/Lab6$ gcc first.c -o first -pthread
Student@project-lab:~/Documents/190905156/OS/Lab6$ ./first
Number of fibonacci numbers? :
10
0 1 1 2 3 5 8 13 21 34
Student@project-lab:~/Documents/190905156/OS/Lab6$
```

Q2)
Write a multithreaded program that calculates the summation of non-negative integers in a separate thread and passes the result to the main thread.

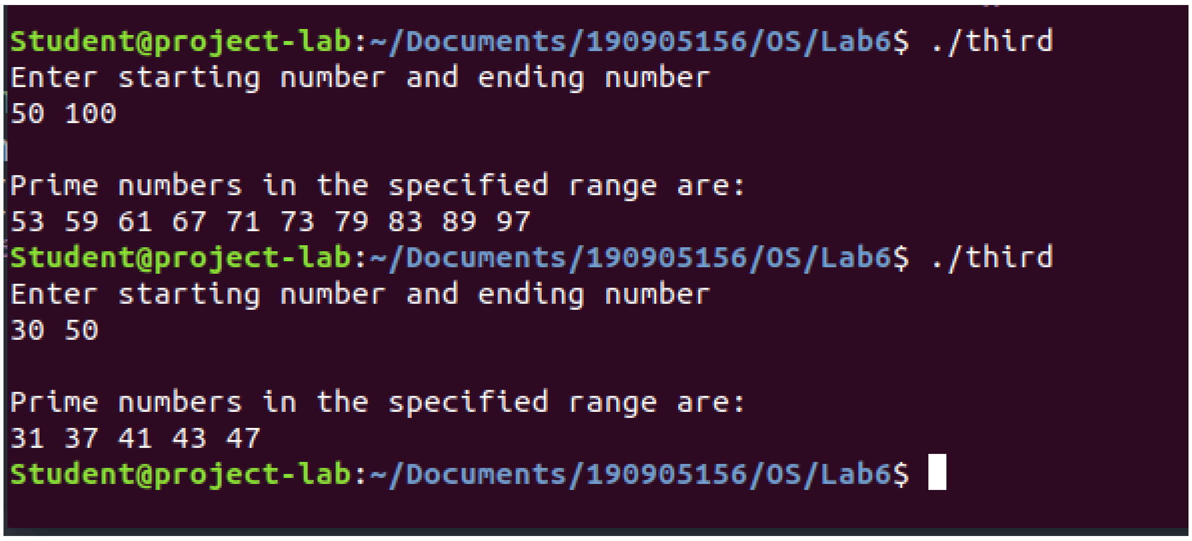
```
#include<stdio.h>
#include<stdlib.h>
#include<pthread.h>

void* summ(void* param)
{
int* arr = (int*)param;
int sum = 0;
int n = arr[0];

for(int i = 1;i <= n;i++)
{
if(arr[i] > 0)
sum += arr[i];
}

return (void*)sum;
}

int main(int argc, char const *argv[])
{
int n;

printf("How many numbers? : \n");
scanf("%d",&n);

int* arr = (int*)malloc((n+1)*sizeof(int));

arr[0] = n;

printf("Enter numbers : \n");

for(int i= 1;i <= n;i++)
{
scanf("%d",&arr[i]);
}

int answer = 0;
pthread_t thread;
pthread_create(&thread,0,&summ,(void*)arr);
pthread_join(thread,(void**)&answer);
printf("Summation of non-negative numbers = %d\n",answer);

return 0;
}
```



```
Student@project-lab:~/Documents/190905156/OS/Lab6$ ./second
How many numbers? :
10
Enter numbers :
1
2
-5
-6
3
-7
-1
5
4
2
Summation of non-negative numbers = 17
Student@project-lab:~/Documents/190905156/OS/Lab6$
```

Q3)
Write a multithreaded program for generating prime numbers from a given starting number to the given ending number.

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

#define N 100
#define MX_THRDS 4

int prime_arr[N]={0};

void *displayPrime(void *ptr)
{
   int  j,flag;
   int i=(int)(long long int)ptr;
   while(i<N)
   {
      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+=MX_THRDS;
  }
}

int main()
{
   pthread_t tid[MX_THRDS]={{0}};
   int count=0;
   printf("Enter starting number and ending number\n");
   int st,en;
   scanf("%d %d",&st,&en);

   for(count=0;count<MX_THRDS;count++)
   {
      pthread_create(&tid[count],NULL,displayPrime,(void*)count);
   }
   printf("\n");
   for(count=0;count<MX_THRDS;count++)
   {
      pthread_join(tid[count],NULL);
   }

   int c=0;
   printf("Prime numbers in the specified range are:\n");
   for(count=st;count<en;count++)
      if(prime_arr[count]==1)
         printf("%d ",count);
printf("\n");
return 0;
}
```

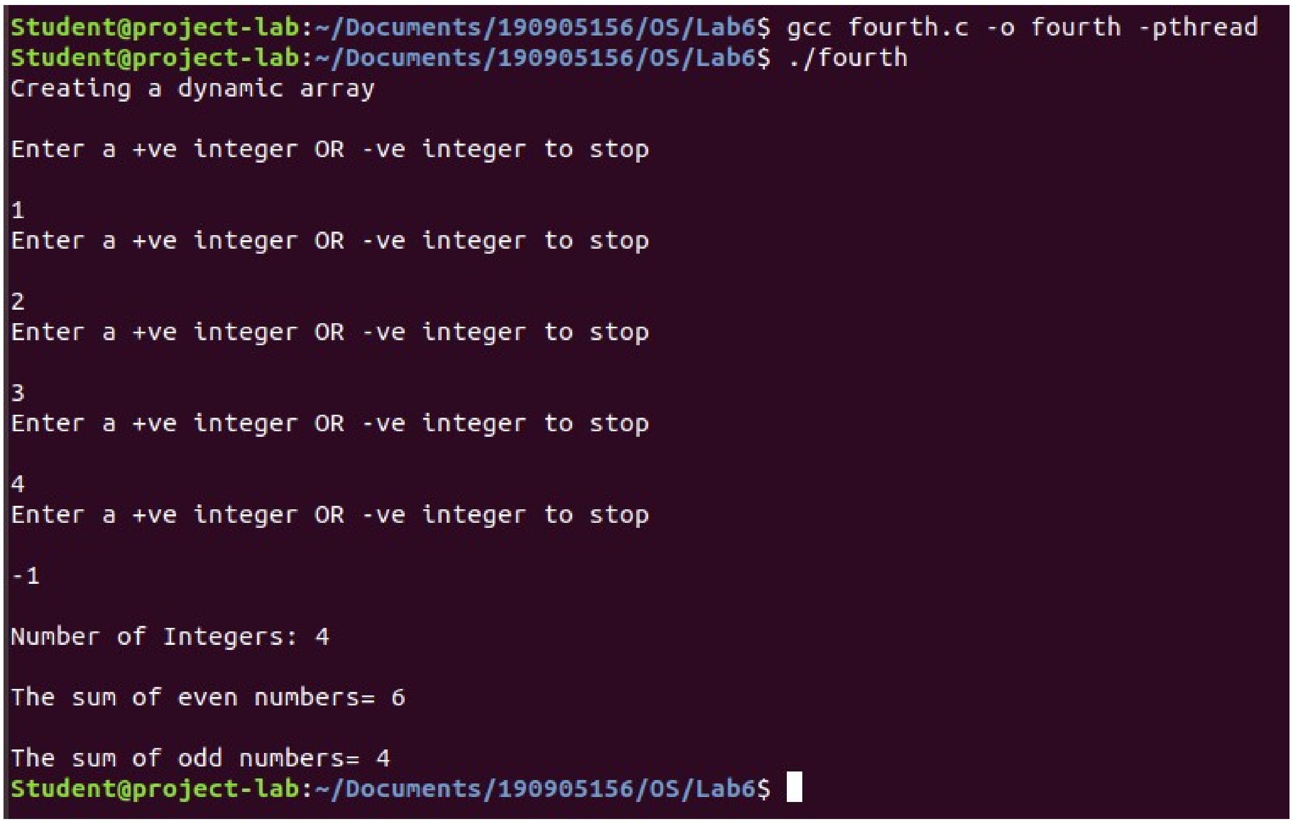


```
Student@project-lab:~/Documents/190905156/OS/Lab6$ ./third
Enter starting number and ending number
50 100

Prime numbers in the specified range are:
53 59 61 67 71 73 79 83 89 97
Student@project-lab:~/Documents/190905156/OS/Lab6$ ./third
Enter starting number and ending number
30 50

Prime numbers in the specified range are:
31 37 41 43 47
Student@project-lab:~/Documents/190905156/OS/Lab6$
```

Q4)
Write a multithreaded program that performs the sum of even numbers and odd numbers in an input array. Create a separate thread to perform the sum of even numbers and odd numbers. The parent thread has to wait until both the threads are done.

```
#include<stdio.h>
#include<stdlib.h>
#include<pthread.h>
#include <errno.h>
#include <ctype.h>
#include <unistd.h>
```

```
#define handle_error_en(en, msg)



/*do
{
errno = en;
perror(msg);
exit(0);}while(0)*/



volatile int running_threads = 0;



pthread_t thread[1];



int numOfElements;



struct Results
{
int sum;
}Results,Results2;



void *findsum(void *array_ptr)
{
int i; /*counter*/
int *elements=(int*)array_ptr;



for(i=0;i<numOfElements;i++)
{
if(elements[i]%2==0)
Results.sum+=elements[i];
else
Results2.sum+=elements[i];
}
running_threads -= 1;
return NULL;
}



int getArrayInput(int n,int *array_ptr)
{
int input;
int numberOfElements = 0;
printf("Creating a dynamic array\n\n");
for(;;)
{
printf("Enter a +ve integer OR -ve integer to stop\n\n");
if (scanf("%d",&input)!= 1)
{
printf("Not an integer\n");
exit(0);
}
if (input>=0)
{
if (numberOfElements==n)
{
n+=1;
array_ptr = realloc(array_ptr, n * sizeof(int));
}
array_ptr[numberOfElements++]=input;
}
else
{
printf("\nNumber of Integers: %d\n", numberOfElements);
break;
}
}
return numberOfElements;
}



void createThreads(int *array_ptr)
{
int s;
s = pthread_create(&thread[2], NULL, findsum, (void *)array_ptr);
if (s != 0)
handle_error_en(s, "pthread_create");
running_threads += 1;
}



int main()
{
int n = 1;
int *array_ptr = malloc(n*sizeof(int));
numOfElements = getArrayInput(n, array_ptr);
createThreads(array_ptr);
while(running_threads>0)
sleep(1);
printf("\nThe sum of even numbers= %d\n",Results.sum);
printf("\nThe sum of odd numbers= %d\n",Results2.sum);
return(0);
}
```



```
Student@project-lab:~/Documents/190905156/OS/Lab6$ gcc fourth.c -o fourth -pthread
Student@project-lab:~/Documents/190905156/OS/Lab6$ ./fourth
Creating a dynamic array

Enter a +ve integer OR -ve integer to stop

1
Enter a +ve integer OR -ve integer to stop

2
Enter a +ve integer OR -ve integer to stop

3
Enter a +ve integer OR -ve integer to stop

4
Enter a +ve integer OR -ve integer to stop

-1

Number of Integers: 4

The sum of even numbers= 6

The sum of odd numbers= 4
Student@project-lab:~/Documents/190905156/OS/Lab6$ 
```