

Name : Rajvardhan Reddy
Reg No : 180905093
Sec : B
Roll No : 19

OS LAB – 8 : Programs on Threads

P1) 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.

Code :

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

void *thread_code(void* param)
{
    int n = *((int*)param);
    int arr[n];
    arr[0] = 0;
    arr[1] = 1;
    for (int i = 2; i < n; i++)
        arr[i] = arr[i - 2] + arr[i - 1];
    int *sol = (int *)calloc(n, sizeof(int));
    memcpy(sol, arr, sizeof(int) * n);
    return sol;
}

void main()
{
    int n;
```

```

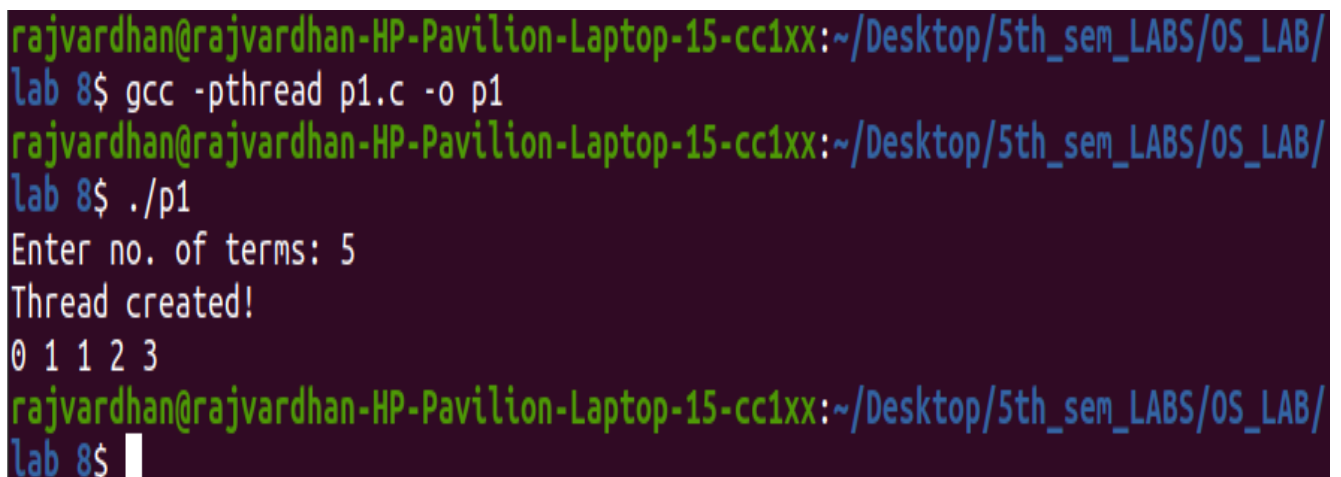
printf("Enter no. of terms: ");
scanf("%d", &n);
void *arr;

pthread_t thread;
pthread_create(&thread, 0, &thread_code, (void*)&n);
printf("Thread created!\n");
pthread_join(thread, &arr);

int *arr2 = arr;
for (int i = 0; i < n; i++)
    printf("%d ", (int)arr2[i]);
printf("\n");
}

```

Output :



```

rajvardhan@rajvardhan-HP-Pavilion-Laptop-15-cc1xx:~/Desktop/5th_sem_LABS/OS_LAB/
lab 8$ gcc -pthread p1.c -o p1
rajvardhan@rajvardhan-HP-Pavilion-Laptop-15-cc1xx:~/Desktop/5th_sem_LABS/OS_LAB/
lab 8$ ./p1
Enter no. of terms: 5
Thread created!
0 1 1 2 3
rajvardhan@rajvardhan-HP-Pavilion-Laptop-15-cc1xx:~/Desktop/5th_sem_LABS/OS_LAB/
lab 8$

```

P2) Write a multithreaded program that calculates the summation of non-negative integers in a separate thread and passes the result to the main thread.

Code :

```

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

void* thread_code(void* n)
{
    int sum = 0;
    int limit = *((int*)n);
    for (int i = 1; i <= limit; i++)

```

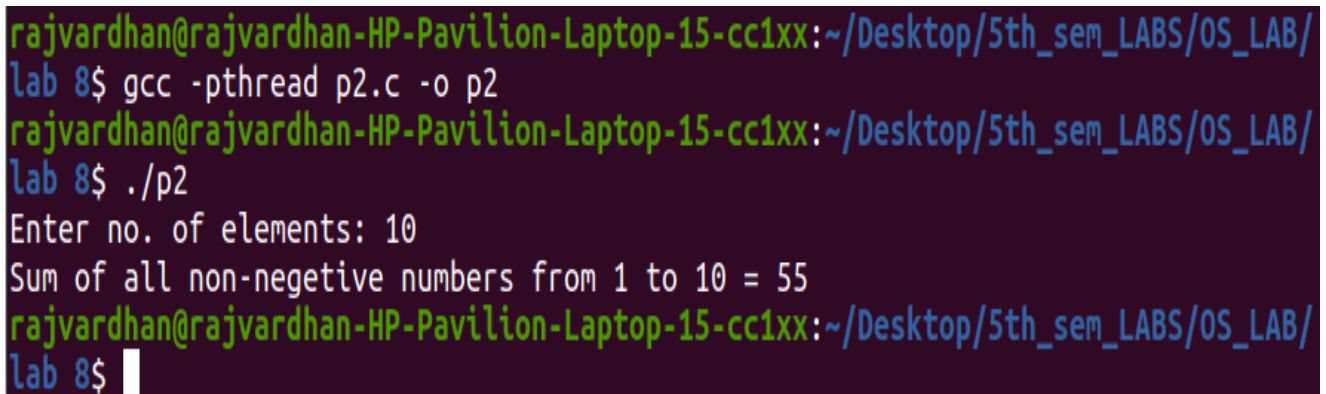
```

        sum += i;
        *((int*)n) = sum;
    }

void main()
{
    int n, num;
    printf("Enter no. of elements: ");
    scanf("%d", &n);
    num = n;
    pthread_t thread;
    pthread_create(&thread, 0, &thread_code, (void*)&n);
    pthread_join(thread, NULL);
    printf("Sum of all non-negative numbers from 1 to %d = %d\n", num, n);
}

```

Output :



```

rajvardhan@rajvardhan-HP-Pavilion-Laptop-15-cc1xx:~/Desktop/5th_sem_LABS/OS_LAB/
lab 8$ gcc -pthread p2.c -o p2
rajvardhan@rajvardhan-HP-Pavilion-Laptop-15-cc1xx:~/Desktop/5th_sem_LABS/OS_LAB/
lab 8$ ./p2
Enter no. of elements: 10
Sum of all non-negative numbers from 1 to 10 = 55
rajvardhan@rajvardhan-HP-Pavilion-Laptop-15-cc1xx:~/Desktop/5th_sem_LABS/OS_LAB/
lab 8$

```

P3) Write a multithreaded program for generating prime numbers from a given starting number to the given ending number.

Code :

```

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

void *prime(void* arr)
{
    int n1 = *((int*)arr);
    int n2 = *(((int*)(arr + sizeof(int)))));
    printf("Prime numbers: ");
    for (int i = n1; i <= n2; i++)
    {
        int flag = 0;
    }
}

```

```

        for (int j = 2; j <= i / 2; j++)
        {
            if ((i % j) == 0)
            {
                flag = 1;
                break;
            }
        }
        if (flag == 0)
            printf("%d ", i);
    }
    printf("\n");
}

int main()
{
    int arr[2];
    printf("Enter lower limit: ");
    scanf("%d", &arr[0]);
    printf("Enter upper limit: ");
    scanf("%d", &arr[1]);
    pthread_t thread;
    pthread_create(&thread, 0, &prime, (void*)arr);
    pthread_join(thread, NULL);
}

```

Output :

```

rajvardhan@rajvardhan-HP-Pavilion-Laptop-15-cc1xx:~/Desktop/5th_sem_LABS/OS_LAB/lab 8$ gcc -pthread p3.c -o p3
rajvardhan@rajvardhan-HP-Pavilion-Laptop-15-cc1xx:~/Desktop/5th_sem_LABS/OS_LAB/lab 8$ ./p3
Enter lower limit:
5
Enter upper limit:
100
Prime numbers: 5 7 11 13 17 19 23 29 31 37 41 43 47 53 59 61 67 71 73 79 83 89 97
rajvardhan@rajvardhan-HP-Pavilion-Laptop-15-cc1xx:~/Desktop/5th_sem_LABS/OS_LAB/lab 8$

```

P4) 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 must wait until both the threads are done.

Code :

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

void * even(void *brr)
{
    int *arr = (int*)brr;
    int size = arr[0];
    int sum = 0;
    for (int i = 1; i <= size; i++)
        if (arr[i] % 2 == 0)
            sum += arr[i];
    return (void *) sum;
}

void * odd(void *brr)
{
    int *arr = (int*)brr;
    int size = arr[0];
    int sum = 0;
    for (int i = 1; i <= size; i++)
        if (arr[i] % 2 != 0)
            sum += arr[i];
    return (void *) sum;
}

int main()
{
    int n, e, o;
    printf("Enter size of array: ");
    scanf("%d", &n);
    int arr[n + 1];
    arr[0] = n;
    printf("Enter elements:\n");
    for (int i = 1; i <= n; i++)
        scanf("%d", &arr[i]);
```

```

pthread_t t1, t2;
pthread_create(&t1, 0, &even, (void *)arr);
pthread_create(&t2, 0, &odd, (void *)arr);
pthread_join(t1, (void*) &e);
pthread_join(t2, (void*) &o);
printf("Sum of even numbers in the array is = %d\n", (int)e);
printf("Sum of odd numbers in the array is = %d\n", (int)o);
}

```

Output :

```

rajvardhan@rajvardhan-HP-Pavilion-Laptop-15-cc1xx:~/Desktop/5th_sem_LABS/OS_LAB/lab 8$ gcc -pthread p4.c -o p4
p4.c: In function 'even':
p4.c:14:9: warning: cast to pointer from integer of different size [-Wint-to-pointer-cast]
   14 |     return (void *) sum;
      |           ^
p4.c: In function 'odd':
p4.c:25:9: warning: cast to pointer from integer of different size [-Wint-to-pointer-cast]
   25 |     return (void *) sum;
      |           ^
rajvardhan@rajvardhan-HP-Pavilion-Laptop-15-cc1xx:~/Desktop/5th_sem_LABS/OS_LAB/lab 8$ ./p4
Enter size of array:
10
Enter elements:
1 2 3 4 5 6 7 8 9 10
Sum of even numbers in the array is = 30
Sum of odd numbers in the array is = 25
rajvardhan@rajvardhan-HP-Pavilion-Laptop-15-cc1xx:~/Desktop/5th_sem_LABS/OS_LAB/lab 8$

```