

Universidad de San Carlos de Guatemala
Facultad de Ingeniería
Curso: Sistemas Operativos 2
Aux: Derek Esquivel Díaz



Juan F. Urbina S. **2019060651**
Sección: A

Guatemala, Marzo 2024

HT2 (1 Hilo)

Código

```
#include <stdio.h>
#include <time.h>
#include <pthread.h>
#include <stdlib.h>
#include <unistd.h>
typedef struct {
    long long int inicio;
    long long int fin;
    long long int suma;
} hilos;

void *suma_hilo(void *arg) {
    hilos *hilo = (hilos *) arg;
    long long int total = 0;
    for (long long int i = hilo->inicio; i <= hilo->fin; i++) {
        total += i;
    }
    hilo->suma = total;
    return NULL;
}

int main() {
    long long int total = 0;
    clock_t start, end;
    double tiempo;
    pthread_t hilo[1];
    hilos hilo1;
    start = clock();
    //vamos con el hijo 1
    hilo1.inicio = 1;
    hilo1.fin = 100000;
    pthread_create(&hilo[0], NULL, suma_hilo, &hilo1);
    //esperar a finalizar hilos
    pthread_join(hilo[0], NULL);
    total = hilo1.suma;
    end = clock();
    tiempo = ((double) (end - start)) / CLOCKS_PER_SEC;
    printf("Tiempo de ejecución (1 Hilo): %f (s)\n", tiempo);
    printf("Total: %lld\n", total);

    return 0;
}
```

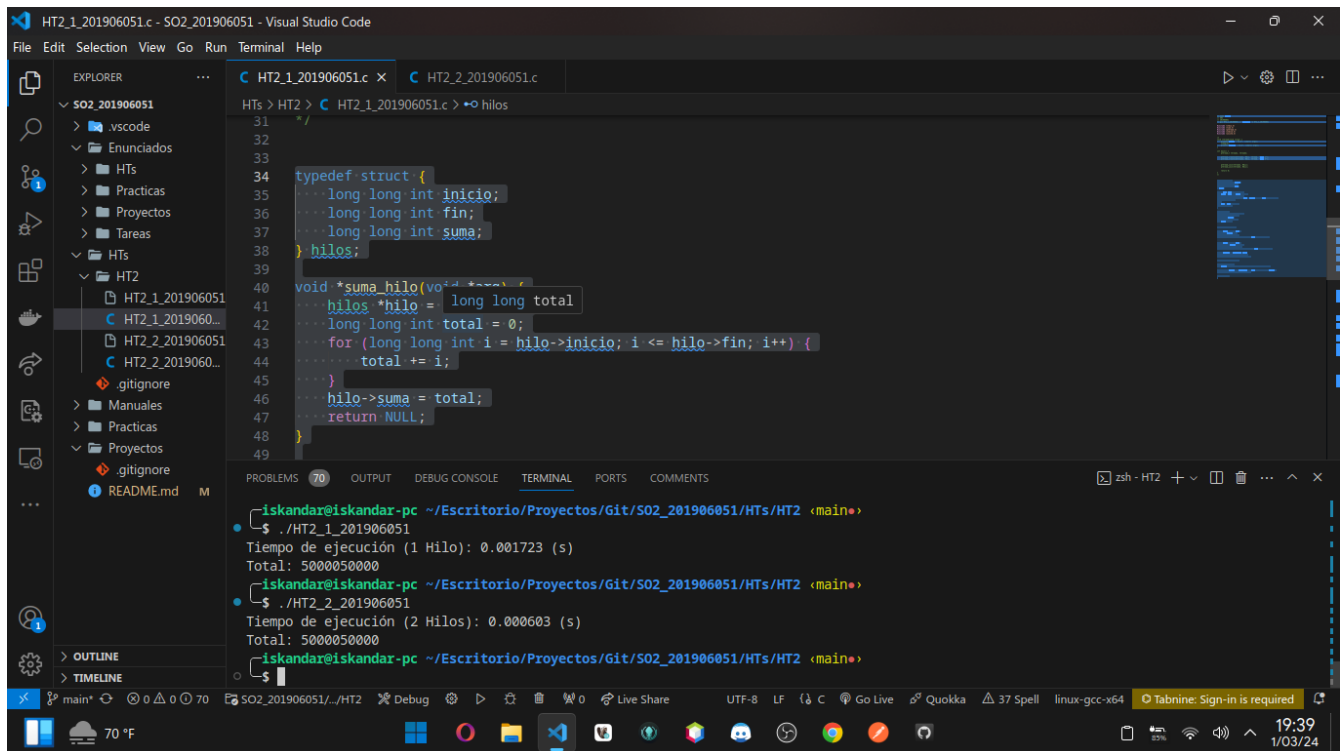
HT2 (2 Hilos)

Código

```
#include <stdio.h>
#include <stdlib.h>
#include <pthread.h>
#include <time.h>
#include <unistd.h>
typedef struct {
    long long int inicio;
    long long int fin;
    long long int suma;
} hilos;
void *suma_hilo(void *arg) {
    hilos *hilo = (hilos *) arg;
    long long int total = 0;
    for (long long int i = hilo->inicio; i <= hilo->fin; i++) {
        total += i;
    }
    hilo->suma = total;
    return NULL;
}
int main() {
    long long int total = 0;
    clock_t start, end;
    double tiempo;
    pthread_t hilo[2];
    hilos hilo1, hilo2;
    start = clock();
    //vamos con el hijo 1
    hilo1.inicio = 1;
    hilo1.fin = 50000;
    pthread_create(&hilo[0], NULL, suma_hilo, &hilo1);
    //vamos con el hijo 2
    hilo2.inicio = 50001;
    hilo2.fin = 100000;
    pthread_create(&hilo[1], NULL, suma_hilo, &hilo2);
    //esperar a finalizar hilos
    pthread_join(hilo[0], NULL);
    pthread_join(hilo[1], NULL);
    total = hilo1.suma + hilo2.suma;
    end = clock();
    tiempo = ((double) (end - start)) / CLOCKS_PER_SEC;
    printf("Tiempo de ejecución (2 Hilos): %f (s)\n", tiempo);
    printf("Total: %lld\n", total);

    return 0;
}
```

Salida en consola



The screenshot shows a Visual Studio Code editor with a C program in a file named `HT2_2_201906051.c`. The code defines a structure `hilo` with fields `inicio`, `fin`, and `suma`. It then implements a function `*suma_hilo` that calculates the sum of integers from `inicio` to `fin` using a loop. The `main` function calls `suma_hilo` twice, once for each thread.

```
31 /*
32 */
33
34 typedef struct {
35     long long int inicio;
36     long long int fin;
37     long long int suma;
38 } hilo;
39
40 void *suma_hilo(void *arg) {
41     hilo *hilo = (hilo *)arg;
42     long long int total = 0;
43     for (long long int i = hilo->inicio; i <= hilo->fin; i++) {
44         total += i;
45     }
46     hilo->suma = total;
47     return NULL;
48 }
49
```

The terminal output shows the execution results for two threads:

```
iskandar@iskandar-pc ~/Escritorio/Proyectos/Git/SO2_201906051/HTs/HT2 <main>
$ ./HT2_1_201906051
Tiempo de ejecución (1 Hilo): 0.001723 (s)
Total: 5000050000
iskandar@iskandar-pc ~/Escritorio/Proyectos/Git/SO2_201906051/HTs/HT2 <main>
$ ./HT2_2_201906051
Tiempo de ejecución (2 Hilos): 0.000603 (s)
Total: 5000050000
iskandar@iskandar-pc ~/Escritorio/Proyectos/Git/SO2_201906051/HTs/HT2 <main>
$
```

- **¿Cuánto tiempo tardo en ejecutarse en un solo hilo?**
 - El tiempo de ejecución en un hilo fue de 0.001723s (caso en el que no se usó ningún sleep para pausar o hacer pausas)
- **¿Cuánto tiempo tardo en ejecutarse en dos hilos?**
 - El tiempo de ejecución de dos hilos fue de 0.000603s (caso en el que no se usó ningún sleep para pausar o hacer pausas)
- **Analice y responda a que cree que se debe la diferencia en los tiempos de ejecución.**
 - El tiempo de diferencia para cada uno se debe a que el de 2 hilos hacía de manera simultánea la sumatoria de los datos dividiendo la carga, haciendo que fuese más rápido el proceso que el de 1 hilo, que hacia la sumatoria completa.
- **¿Es adecuado utilizar programación multihilo en este caso? ¿Por qué?**
 - Big depending de qué se querría, desde mi punto de vista no es del todo necesario (en este caso) ya que de por sí no lleva una cantidad exorbitante de tiempo el usar hilos, a decir verdad, que por una centésima de tiempo más que la otra es risible para este caso en específico; ya para procesos más complejos que se quieran llevar o hacer mayor cantidad de procesos en un menor tiempo posible o de manera simultánea, sería de tomarlo en cuenta para la optimización de tiempo y/o recursos.