CI-0122 Tarea 1 Sistemas Operativos
Rodrigo Vilchez Ulloa | B78292

---

1. `SumaUno.c`

```
rigovil@rodrigo:~/Escritorio/UCR/2020 - II Semestre/CI-0122/tareas cortas/Semana
 1$ gcc sumaUno.c -w
rigovil@rodrigo:~/Escritorio/UCR/2020 - II Semestre/CI-0122/tareas cortas/Semana
 1$ ./a.out
Serial version:      Valor acumulado por 100 procesos es  100000  en 0.550000 ms
Fork   version:      Valor acumulado por 100 procesos es  0  en 76.235000 ms
```

El programa de forma serial suma uno 1000 veces por cada proceso según la cantidad que haya sido indicada, el número de procesos solo es para establecer un límite pues todo se ejecuta de manera serial. Pero también hace el mismo cálculo de manera paralela, creando efectivamente un proceso por cada uno que haya sido indicado, y ese proceso va a sumarle uno 1000 veces a una variable que ha sido enviada como puntero, sin embargo persiste el mismo problema que el programa anterior pues el puntero no corresponde a la dirección de la variable original del proceso padre, entonces el valor solo se suma en la dirección de la copia de esa variable en el espacio de memoria de ese proceso.

2. `PiPorSeries.c`

```
rigovil@rodrigo:~/Escritorio/UCR/2020 - II Semestre/CI-0122/tarea
 1$ gcc PiPorSeries.c -w
rigovil@rodrigo:~/Escritorio/UCR/2020 - II Semestre/CI-0122/tarea
 1$ ./a.out
Creating process 78977: starting value 0, finish at 100000
Creating process 78978: starting value 100000, finish at 200000
Creating process 78979: starting value 200000, finish at 300000
Creating process 78980: starting value 300000, finish at 400000
Creating process 78981: starting value 400000, finish at 500000
Creating process 78982: starting value 500000, finish at 600000
Creating process 78983: starting value 600000, finish at 700000
Creating process 78984: starting value 700000, finish at 800000
Creating process 78985: starting value 800000, finish at 900000
Creating process 78986: starting value 900000, finish at 1000000
Valor calculado de Pi es  0  con 1000000 terminos
```

El programa lo que hace es crear 10 procesos (en realidad se crean más), pero solamente esos 10 son los encargados de calcular una parte para encontrar el resultado de la aproximación de pi, cada uno de estos procesos tiene un identificador (de 0 a 9) que utilizan para calcular su parte y la guardan en un array de acuerdo con ese id. Al final, el proceso padre suma todos los valores dentro de ese array y debería de dar un valor de aproximación a pi, el problema es que al utilizar fork, los procesos copian el espacio de memoria del proceso padre pero son espacios independientes, por lo que todos los procesos guardan su resultado parcial en un array que no es el mismo que el del proceso padre, dando así un valor de 0, que es el valor con el que se inicializa ese array.

3. Programa `pruebSem.c` funcionando con la clase `Semaforo`

```
rigovil@rodrigo:~/Escritorio/UCR/2020
 1$ g++ pruebaSem.cc Semaforo.cc -w
rigovil@rodrigo:~/Escritorio/UCR/2020
 1$ ./a.out
Voy primero
Voy segundo
Voy primero
Voy segundo
```

Instrucciones: compilar el programa utilizando el comando `make` y correrlo con el comando `./pruebaSem`.