Assignment 2

Question 1 Documentation

Program Logic / How the Program Works:

There are 3 C files for this question namely **S1**, **SR** and **ST**. The main program first calls the fork system call and produces the **S1** process, and then again in the parent process another fork is called giving us the **SR** process and another fork call from the parent process produces the **ST** process.

In the **S1** process body, I first store the process Id in a local variable and then register a signal for the **SIGTERM** signal using the sigaction function and declare a signal handler function for S1's **SIGTERM**. I then start an infinite loop so that, the process once started never ends without user interrupt.

In the main parent program the **SR** process which is forked calls the execl function on an executable for the **SR.c** file which is **E1**. The execl also passes on the pid of **S1** so that **SR** can call sigqueue on **S1** using its pid.
The **SR** process first sets a timer using the setitimer function and set an interval of **2.5 seconds** and then starts an infinite loop so that the process never ends without any user interrupt.

Then it also registers a signal handler for the **SIGALRM** signal. The signal handler uses the inline assembly function to produce a random integer. This random integer is sent to the **S1** process using the sigqueue function and **S1's SIGTERM** handler prints this integer in **S1**. The pointer for the integer to be sent is set using a **union sigval** which can be passed as an argument in **sigqueue**.

In the **ST** process, the execl is called upon the executable for **ST.c** which is **E2**. The execl also passes on the pid of **S1** so that **ST** can call sigqueue on **S1** using its pid.
The **ST** process first registers a signal handler for the **SIGALRM** signal.

This signal handler uses the rdtsc inline assembly command to get the **TimeStamp**(**CPU counter**) and then this timestamp counter is converted to seconds and converted into a **human-readable form (HH:MM:SS),** which is passed on to the S1's **SIGTERM** handler using shared memory, which then prints it. The shared memory between S1 and ST processes is used to store the time string which is used by S1 to print.

The ST's main function also set a timer using the setitimer function and sets the interval at **2 seconds** and then starts an infinite loop so that the process never ends without any user interrupt.

How to run the entire program:

1. Type **make** in the terminal once you are in the directory of this question.
2. Type **./a.out** to execute the file.

How to stop the execution of the program:

Press **Ctrl + C.**