Assignment 1 Report

PLAGIARISM STATEMENT:

I certify that this assignment/report is my own work, based on my personal study and/or research and that I have acknowledged all material and sources used in its preparation, whether they be books, articles, reports, lecture notes, and any other kind of document, electronic or personal communication. I also certify that this assignment/report has not previously been submitted for assessment in any other course, except where specific permission has been granted from all course instructors involved, or at any other time in this course, and that I have not copied in part or whole or otherwise plagiarised the work of other students and/or persons. I pledge to uphold the principles of honesty and responsibility at CSE@IITH. In addition, I understand my responsibility to report honour violations by other students if I become aware of it.

***Name*: Shreyas Jayant Havaldar**
***Roll No*: CS18BTECH11042**
***Date*: 3-12-2019**

***Signature*: S.J.H.**

QUESTION 1

APPROACH

I initially forked off 1 process from the parent. The parent on successful fork of child 1, proceeds to fork off another process, child 2. An error in any fork was handled using the *perror()* function.
Parent: Waits using *waitpid()* until both the children have terminated and then prints a message to denote it is exiting.
Child 1: Employs an infinite loop to keep printing 2 statements with the intermittent gap of 1s using *sleep()* function, until it is killed by Child 2.
Child 2: Prints a message and then sleeps off for 10s. Next, it kills Child 1 using *kill(pid1, SIGKILL)* and then sleeps for 10s again. Then it prints a message and terminates.
Note: The first statement may be either printed by Child1 or Child2, it is completely random, as it occurs at arbitrarily the same time.

SAMPLE OUTPUT

Child1 says hello and sleeps for 1s.. PID: 10390
Child2 says hi and sleeps for 10s! PID: 10391
After sleeping for 1s.. PID: 10390
Child1 says hello and sleeps for 1s.. PID: 10390
After sleeping for 1s.. PID: 10390
Child1 says hello and sleeps for 1s.. PID: 10390
After sleeping for 1s.. PID: 10390
Child1 says hello and sleeps for 1s.. PID: 10390
After sleeping for 1s.. PID: 10390
Child1 says hello and sleeps for 1s.. PID: 10390
After sleeping for 1s.. PID: 10390
Child1 says hello and sleeps for 1s.. PID: 10390
After sleeping for 1s.. PID: 10390
Child1 says hello and sleeps for 1s.. PID: 10390
After sleeping for 1s.. PID: 10390
Child1 says hello and sleeps for 1s.. PID: 10390
After sleeping for 1s.. PID: 10390
Child1 says hello and sleeps for 1s.. PID: 10390
After sleeping for 1s.. PID: 10390
Child1 says hello and sleeps for 1s.. PID: 10390
Child2 Killing Child1.. PID: 10391
After killing Child1.. Sleeping for 10s PID: 10391
After sleeping for 10s.. PID: 10391
Child terminates.. PID: 10391
Parent will now exit.. PID: 10389

QUESTION 2

APPROACH

We created a utility function *getNanos()* to calculate the time in nanoseconds. The *main()* function runs a loop for 20 times to perform 20 context switches and note down the time taken for them. *runner()* function employs *fork()* to create a child process. The child passes a signal to the parent using *kill()* and thus the time taken in context switching is measured. The signal handler SIGINT, which invokes *sigPass()*, is used while passing signal from child process. The parent then kills the child using *kill()* with signal handler SIGKILL. This is repeated 20 times.
A pretty consistent context switch time of about 30,000 ns = 30 microseconds is obtained which is verified by checking against online averages.
After this the *info()* function is invoked which prints the necessary system info. Libraries and files used include:

- sys/utsname.h
- /proc/cpuinfo
- /proc/meminfo

The files data was accessed using *fopen()* and storing the data in a FILE pointer. The necessary details were obtained from the respective files and then printed.

Sample output for my system is on the next page:

SAMPLE OUTPUT

```
Context Switch no: 1 || Time taken in nanoseconds: 37259 || Context switch complete
Context Switch no: 2 || Time taken in nanoseconds: 37450 || Context switch complete
Context Switch no: 3 || Time taken in nanoseconds: 39529 || Context switch complete
Context Switch no: 4 || Time taken in nanoseconds: 30838 || Context switch complete
Context Switch no: 5 || Time taken in nanoseconds: 42461 || Context switch complete
Context Switch no: 6 || Time taken in nanoseconds: 60974 || Context switch complete
Context Switch no: 7 || Time taken in nanoseconds: 31456 || Context switch complete
Context Switch no: 8 || Time taken in nanoseconds: 33390 || Context switch complete
Context Switch no: 9 || Time taken in nanoseconds: 33695 || Context switch complete
Context Switch no: 10 || Time taken in nanoseconds: 31734 || Context switch complete
Context Switch no: 11 || Time taken in nanoseconds: 29857 || Context switch complete
Context Switch no: 12 || Time taken in nanoseconds: 31452 || Context switch complete
Context Switch no: 13 || Time taken in nanoseconds: 30161 || Context switch complete
Context Switch no: 14 || Time taken in nanoseconds: 28714 || Context switch complete
Context Switch no: 15 || Time taken in nanoseconds: 28818 || Context switch complete
Context Switch no: 16 || Time taken in nanoseconds: 28248 || Context switch complete
Context Switch no: 17 || Time taken in nanoseconds: 28328 || Context switch complete
Context Switch no: 18 || Time taken in nanoseconds: 28399 || Context switch complete
Context Switch no: 19 || Time taken in nanoseconds: 28375 || Context switch complete
Context Switch no: 20 || Time taken in nanoseconds: 28178 || Context switch complete
Printing System Details:
 OS Type        : Linux
 OS Release     : 4.15.0-70-generic
 OS Version     : #79-Ubuntu SMP Tue Nov 12 10:36:11 UTC 2019
 CPU model name : Intel(R) Core(TM) i7-8550U CPU @ 1.80GHz
 CPU cache size : 8192 KB
 RAM Size in GB : 7.689438
```