CS 314: Lab Assignment 2

Hrishikesh Ravindra Karande 210010020

Task1:
Print the string “Hello World” on screen. Each character must be printed by a different process. The process that prints the i th letter must have been spawned by the process that printed the (i-1) th letter. Each process must first print its designated character, as well as its own process ID, second, sleep for a random number of seconds (from 1 to 4 seconds), and then, do anything else it must do to achieve the given task.

Q1] What is the minimum lines of C code with which you can achieve the above?
After following good coding practice to code in C language and including necessary header files the number of lines of code that I have written inorder to solve this Task is **29.** However C does not requires any line breaks(except for #include Header Files) so the minimum lines can be reduced as much as possible sacrificing the code readability.

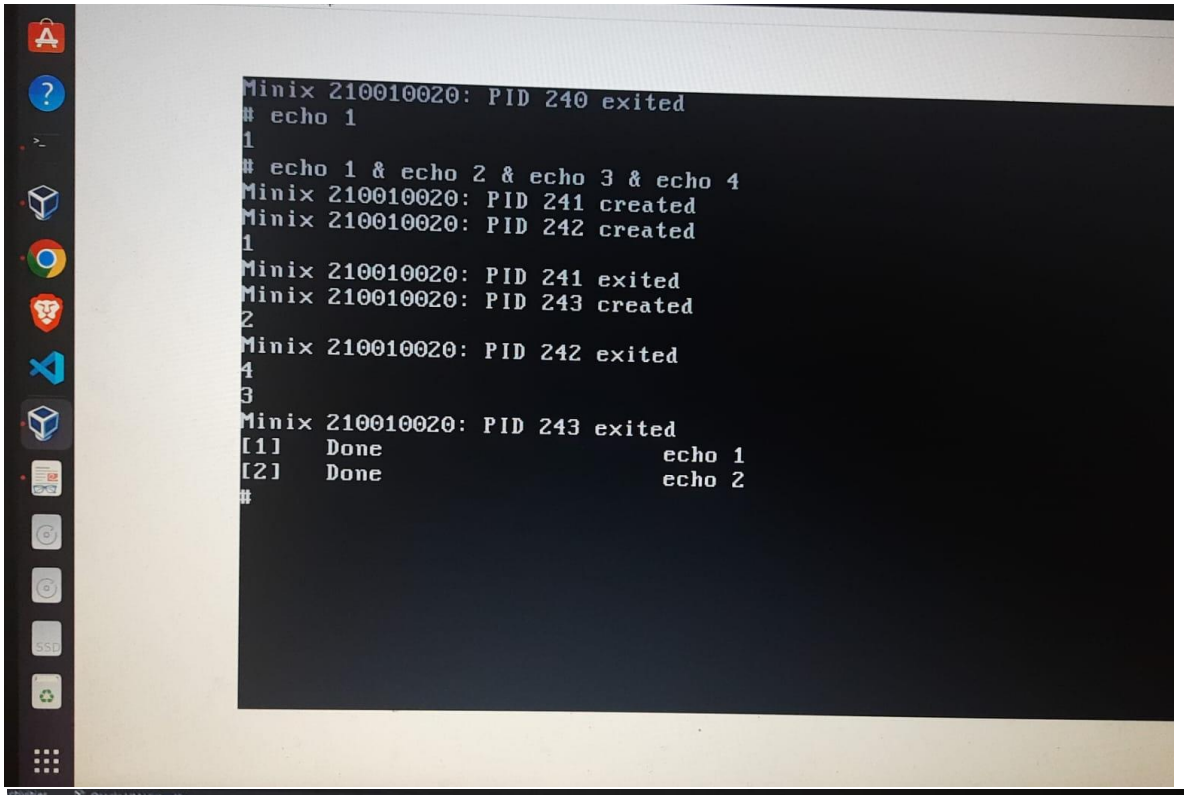
Task2:
Write a collection of programs twice, half, square such that they execute sequentially with the same process-id, and each program should also print its PID. (process id) The user should be able to invoke any combination of these programs, to achieve the required functionality

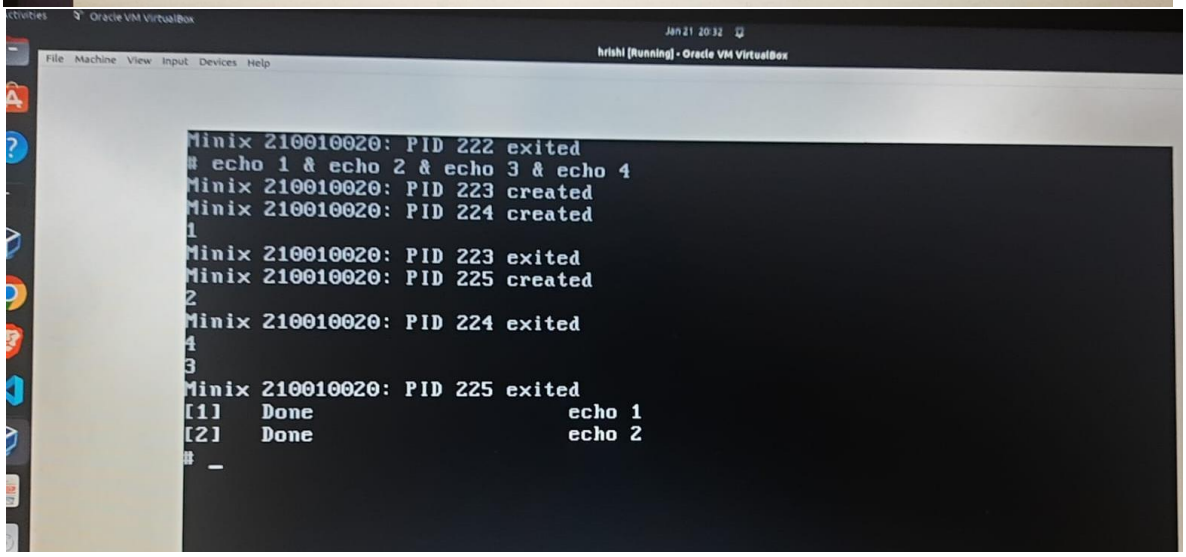
A] Using execvp(): next file name and arguments for next file are passed as arguments for this system call. The address space of new process is replaces the current process address space when this function is invoked.
Remarks: There are three files written half.c square.c twice.c and Makefile, to execute just enter ‘gmake compile’. Then you can run the required testcases as given in the sample inputs.

Task 3:

Modify the Minix3 source code such that:
- A message “Minix : PID created” is printed, whenever a process is created. (Let us follow the convention throughout this course that anything printed by the Operating System code will be prepended by the string “Minix: “.)
- A message “Minix : PID exited” is printed, whenever a process ends.

A] Files modified forkexit.c located at minix/servers/pm.
This file contains many functions related to creating and deleting them by fork() and waitpid()/exit() respectively. At the time of process creation new pid is forked and inserted in the mproc table. To print it we need to add print statement in do_fork() and while exiting add it in do_exit() function.





- ORDER OF PROCESS CREATION: FIRST TWO PROCESSES ARE CREATED TOGETHER ONE AFTER OTHER. WHEN THE FIRST ONE COMPLETES ITS EXECUTION THE SECOND PROCESS IS STILL PRESENT AND THE THIRD ONE IS CREATED.
- DURING EXECUTION THE LAST TWO PROCESS EXECUTE REVERSE ORDER THAN THEIR CREATION ORDER, AS CAN BE SEEN FROM THE SCREENSHOT ABOVE.
- DURING CREATION OF PROCESS THE PARENT IS CREATED FOLLOWED BY CHILD BUT DURING EXECUTION CHILD EXITS FIRST THEN THE PARENT EXITS, THIS IS AS EXPECTED.
- IF DURING EXITING THE PROCESS EXITS IN ORDER OF CREATION I.E THE PARENT TERMINATES BEFORE THE CHILD COMPLETES THEN IT BECOMES AN ORPHAN AND IS ADOPTED BY INIT PROCESS.