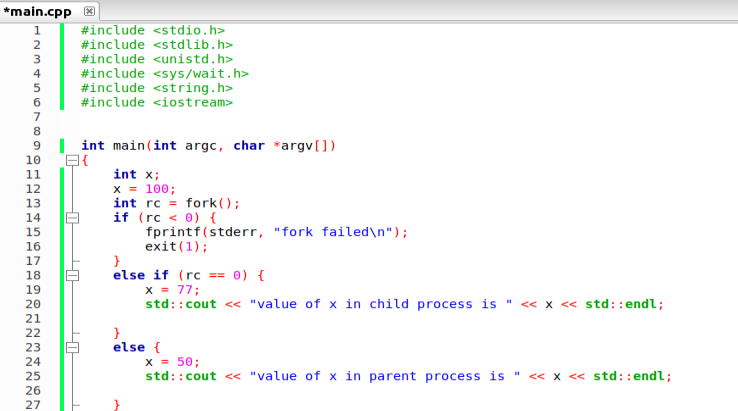
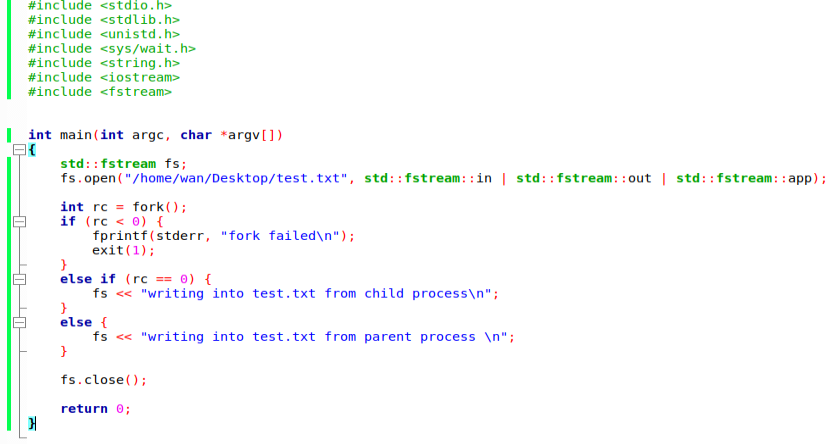
**Homework (Simulation) Wan Huzaifah bin Wan Azhar**

In this homework, you are to gain some familiarity with the process management APIs about which you just read. Don’t worry – it’s even more fun than it sounds! You’ll in general be much better off if you find as much time as you can to write some code, so why not start now?

**Answer:**



In the diagram, x is first initialized to 100. If x is not changed in the child and parent process, then the child will inherit the state of x before it was being forked, which is 100. If the parent and the child change the value of x, then both of the x will change respective to the number that it has changed, e,g, x for child is 77 and x for parent is 50.



Test.txt output:



The result shows that the parent and the child process can access the file descriptor returned by open(). As such, both processes can write into the file. Both process are trying to write at the same time but only one process can gain access to the file at a time. As such, parent child writes to the file before child process to the write (or vice-versa). Thus, this avoids problem of competing for resources.

1. Running child process first



Sleep() function will suspend execution of parent process for 0.3 seconds, which will ensure child process run first before parent process. This is not deterministic, however.

1. Exec command



There are many variants of the exec command because each variant have its own uses. For example, execl, execlp, execle takes list of string as argument while execv, execvp and execvpe accept vector of string as argument. Execlp and execvp will search for the file in the Path if the trailing character / is not specify while execle and execvpe are able to specify the environment of the exec function differs from its parent process.

1. Wait command

Wait() will return the pid of the process that is waiting for other process to finish. If wait() is in parent process, then after the child is finished, wait() will return the pid of the parent process. If wait() is specified in the child process, then nothing will happens as the child cannot wait for its parent while the parents will wait for its child. The output then is deterministic.

1. Waitpid command



Waitpid() will be useful for handling error if the child process exited improperly or if the user forcefully terminate the child process. Waitpid() is also useful if a process wants to wait for a specific child process to finish rather than one random child process.

1. Closing File Descriptor in child process

Closing STDOUT\_FILENO and displaying output using printf will not display the printf message in the standard output.

1. Redirect stdout from first child to second child

