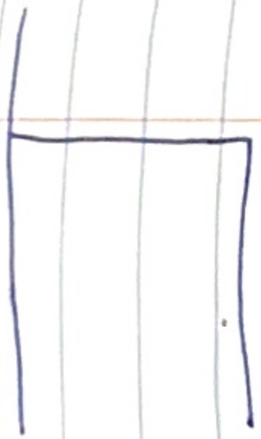
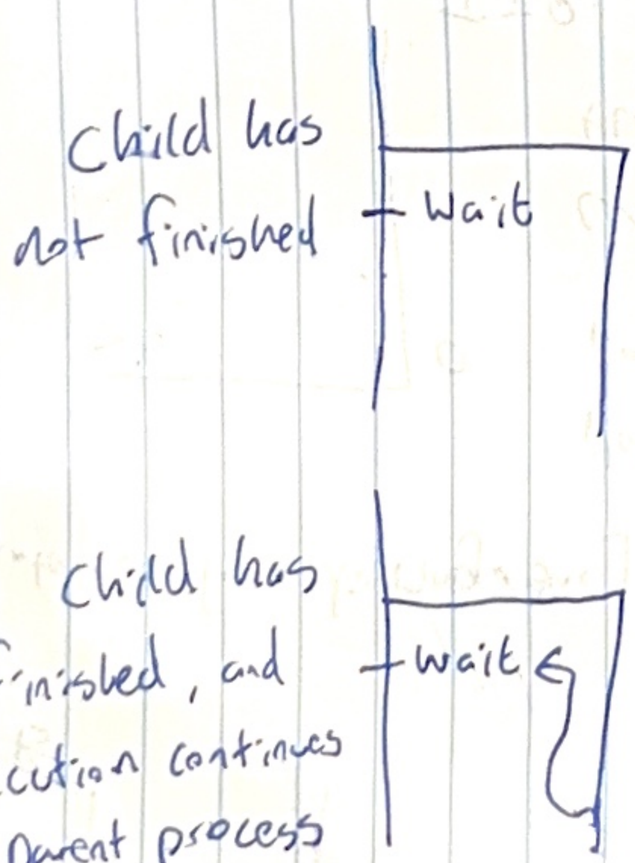
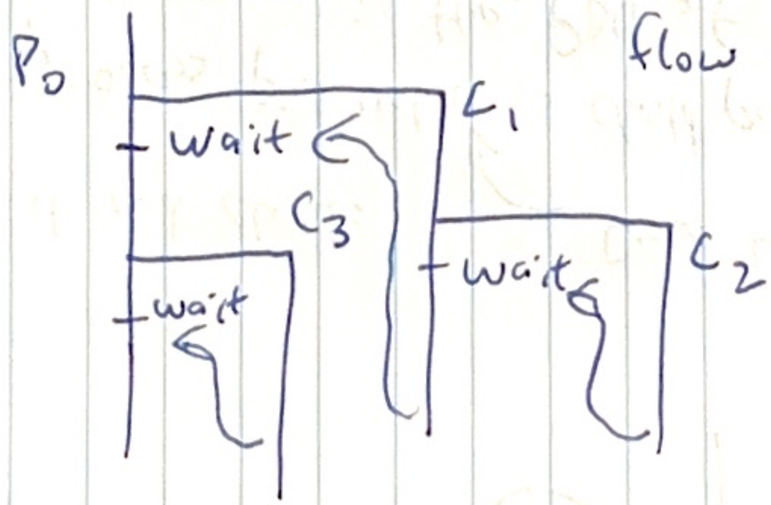


The left branch corresponds to the parent process in fork()
The right branch corresponds to the child process.

The notation for waiting in wait(NULL) or waitpid() is
, where the parent process must wait for the child process to terminate before moving on.



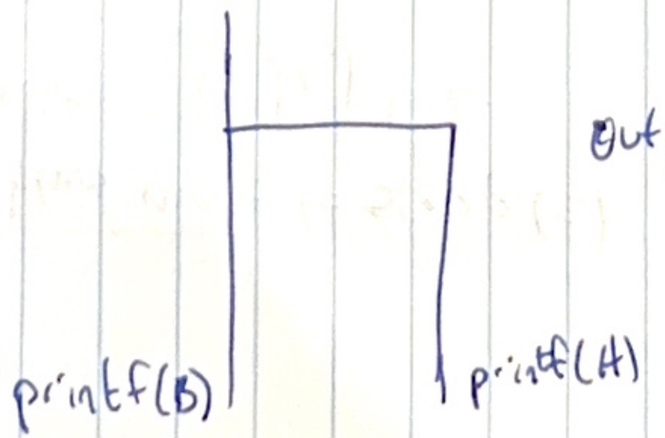
TIP: parent process wait only for <u>immediate</u> children



flow of execution is:

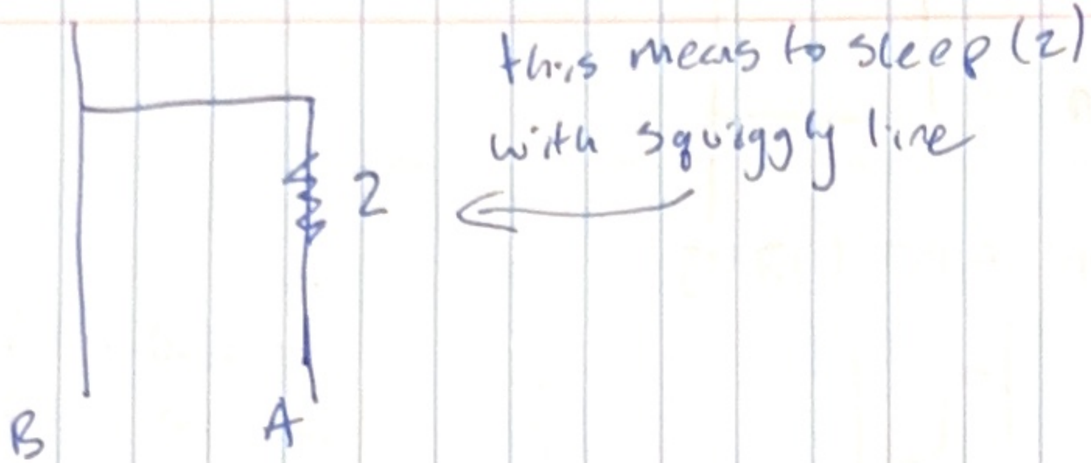
$C_2 \rightarrow C_1 \rightarrow P_0 \rightarrow C_3 \rightarrow P_0$

fork() calls also lead to NONDETERMINISTIC behavior without waits or sleep() calls



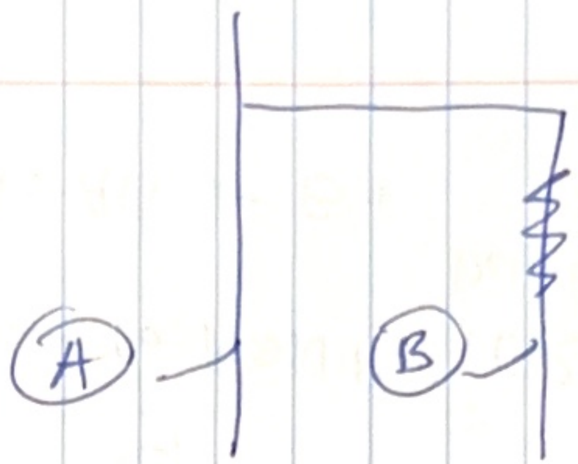
Output: AB or BA

notation for sleep()



this means to sleep(2) with squiggly line

notation for printf



Circled things are stuff printed to terminal

notation for distinguishing parent and child:

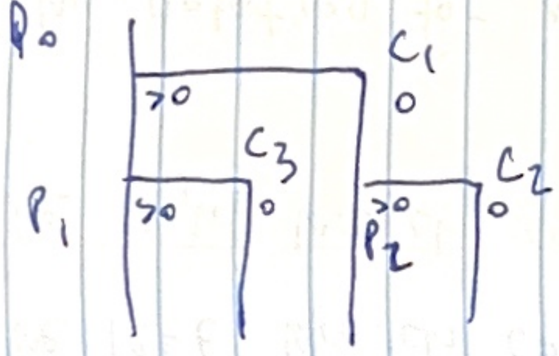
Parent process pid value is positive integer in the POV of the parent process

child process pid value is always zero in the POV of the child process

from fork() return value

is child pid number which is always a

TIP: when comparing pid vals from fork() e.g. if (fork() == fork())



the pid val from P_0 and P_1 are both >0 but are not equal since pid vals are incremented by one each time fork() is called, leading to 2 distinct positive integers.

WOW

>0
fork 0
forks=1
wait
waitpid 0

2.1 3 sec
2.2 6
2.3 0
2.4 3
2.5 3
2.6

fork 0
fork 0
sleep 0
fork 0
fork 0
fork 0
sleep 0

waitpid 0
waitpid 0
waitpid 0

0
fork 0
forks=1
1
sleep 0
sleep=1

>0
fork 0
fork=1
wait
waitpid 0

OW

0
fork 0
fork=1
1
sleep 0

>0
fork 0
wait
waitpid 0

0
fork 0
1
sleep 0