

Guião aula08 - Gil Guedes 125031

1 - a - **getpid** - Retorna o identificador de processo (PID) do processo chamador

getppid - Retorna o PID do processo pai do processo chamador

fork - Cria um novo processo ao duplicar o processo chamador

(executei o código antes)

b - i - 5 linhas.

ii - O `printf(Antes do fork: PID = %d, PPID = %d\n', getpid(), getppid());` vai imprimir “Antes do fork: PID = 6315, PPID = 6114”.

O `printf(Quem sou eu?\nApós o fork: PID = %d, PPID = %d\n', getpid(), getppid());` vai imprimir “Quem sou eu? Após o fork: PID = 6316, PPID = 6315” para o processo Pai e depois para o processo Filho.

c - O processo pai de PPID=6114, é o PID da shell.

2 - a - O código é praticamente igual ao ‘fork1’, mas agora no print do após, ele imprime o valor do fork. Se fork=0 é o processo filho e se o fork=PID do filho é o processo pai.

b - Usando o valor de retorno do fork, sabe se que o processo pai é maior que zero, tendo o valor do PID do filho, e o processo filho tem sempre valor 0 no fork, é assim que os distinguimos.

3 - a - O `execl` substitui o processo atual por outro programa. Mas mantém o mesmo PID.

b - ii - ‘./child’

c - i - A instrução `printf("Porque é que eu não apareço?\n");` nunca é executada porque a função ‘`execl()`’ substitui completamente o processo atual por um novo.

ii - O PID mantém se e o PPID muda de um print para o outro.

4 - a - **wait** - espera por qualquer filho

waitpid - permite escolher qual filho esperar e definir o seu comportamento

b - i - A mudança fundamental é que o pai agora espera que o filho termine antes de prosseguir.

ii - A ordem de execução é Pai, Pai, Filho, Filho, Pai.

iii - O prompt da shell será posicionado no final, após todas as mensagens dos processos.

5 - a - O programa imprime o PID depois conta até 10 e atualiza o PID. O 'fflush' vai atualizar o contador nos 10 segundos de espera.

c - O CTRL-C para o processo.

6 - a - **sigaction** - configura a ação a executar quando um sinal é recebido

b - Não se pode parar o programa antes que o contador chegue a 50.

d - **kill** - envia um sinal a um processo para o terminar

e - Acontece a mesma coisa que na alínea c.