UFCSPA – Informática Biomédica

Sistemas Operacionais – 19/02 – Prof. João Gluz

Lista de Exercícios 1 – Exercícios sobre Processos, Sinais e Pipes

(1) Criar um programa que se abre em 2 processos. O processo pai imprime "eu sou o pai" e seu *pid*. O filho imprime "eu sou o filho" e seu *pid*.

(2) Criar um programa que se abre em 2 processos e que possua pelo menos uma variável global, que deve ser inicializada antes da operação ***fork***(). Depois verifique se o valor da variável global também é herdado pelo filho, imprimindo o valor dessa variável em cada processo.

(3) Altere o programa do exercício (3) atribuindo valores distintos a variável global no processo pai e no processo filho. Depois verifique se a alteração da variável do processo pai é vista pelo processo filho e vice-versa.

(4) Faça um programa que cria um processo pai e um processo filho, depois faz o processo filho imprimir N números pares. A quantidade N de números pares a ser impressa é passada como parâmetro (argumento) de linha de comando do programa. O processo pai espera o processo filho terminar a execução e só então imprimir uma mensagem indicando que terminou sua execução.

(5) Faça um programa que implemente a primitiva ***system***() usando ***execl***() e ***fork***(), isto é, faça o código do processo filho ser substituído pelo código de um outro programa qualquer. O nome do programa que ira substituir o código do processo filho deve ser passado como parâmetro de linha de comando para o programa.

(6) Faça um programa que cria um processo filho e também um duto (*pipe*) nomeado através da função ***pipe***() e faz com que o processo pai leia do teclado e o processo filho imprima na tela. A comunicação entre os processos é através do duto.

(7) Faça um programa que cria um processo filho e também um duto nomeado através da função ***pipe***() e faz com que o processo filho leia do teclado e o processo pai imprima na tela. A comunicação entre os processos é através do duto.

(8) Faça um programa que cria um processo filho e se conecta ao filho através de um duto (pipe) que será utilizado para comunicação bidirecional. A linha de comando deste programa deve ter dois argumentos: o primeiro argumento define que operação aritmética será feita pelo processo filho: “-a” para adição, “-s” para subtração, “-m” para multiplicação e “-d” para divisão; o segundo argumento é um valor numérico ponto flutuante que será usado na operação aritmética. O processo pai lê valores do teclado e envia esses valores para o processo filho. O processo filho recebe os valores do processo pai pelo duto de comunicação pai para filho, executa a operação selecionada pelos argumentos da linha de comando e envia o resultado de volta ao pai pelo duto de comunicação filho para pai. Os processos pai e filho se encerram quando for lido o valor 0.

(9) Faça dois programas distintos para testar a comunicação por sinais: um programa mestre e um programa escravo. Cada programa será executado como um processo independente em um terminal distinto. Cada programa inicialmente imprime seu pid no terminal usando a função ***getpid***(). O programa mestre envia pode enviar os sinais SIGUSR1, SIGUSR2 e SIGTERM para o programa escravo através da função ***kill***() de acordo com a escolha do usuário. O programa escravo deve capturar os sinais enviados pelo programa mestre com o uso da função ***signal***(). Cada vez que um desses sinais for capturado deve ser impresso uma mensagem no terminal informando que essa captura ocorreu. Tanto o programa mestre quanto o escravo encerram a execução quando o usuário selecionar a opção de envio de SIGTERM. O programa mestre deve esperar o fim do programa escravo e depois informar na tela que isso ocorreu.