INSTITUTO FEDERAL
Paraíba

Sistemas Operacionais

Atividade Prática sobre Processos

OBS: Cada questão vale 1,0.

Neste roteiro iremos exercitar alguns conceitos estudados em sala de aula sobre processos e *threads*. Siga os passos abaixo atentando para as questões que precisam ser respondidas (destacadas em amarelo). Tente responder por conta própria, seu aprendizado só irá ocorrer assim!

Para esta prática você irá precisar de um ambiente Linux para executar o código, uma vez que faremos uso do padrão POSIX.

Algumas orientações:

- Elabore um documento (.doc, .docx, etc.) contendo as respostas para as questões e os PRINTS das telas**;**
- Ao elaborar o relatório desse roteiro, FAÇA PRINTS DA TELA SEMPRE QUE POSSÍVEL**.**

Roteiro

1. Faça *download* do arquivo para a prática
2. Abra um terminal
3. Descompacte o arquivo. No terminal digite (você precisa estar na pasta onde foi realizado o *download* do arquivo):

   tar xvzf process_exercise.tar.gz

4. Encontre o arquivo **processdemo.c** dentro da pasta descompactada. O programa usa a chamada ***fork()*** para criar um processo filho. Os processos pai e filho são executados separadamente e cada um chama a função ***adjustX()*** com parâmetros diferentes em cada processo
5. Examine o código-fonte e tente determinar o que ele faz. Você pode fazer isso usando o programa **gedit**. Neste caso, você poderia digitar no terminal:

   gedit processdemo.c
6. Faça um exercício de tentar prever o que o código vai fazer
7. Compile o código-fonte. No terminal digite:

   gcc processdemo.c -o processdemo
8. Rode o programa. No terminal digite:

   ./processdemo
9. Descreva a saída e explique por que ela é dessa forma.
10. Enquanto o programa roda, execute (em outro terminal) o seguinte comando para visualizar os processos existentes na sua máquina:

    ps xl
11. Qual o processo pai e qual o processo filho? (Dica, verifique a coluna PID e PPID. Se não souber o que é PID e PPID, procure no Google). Justifique sua resposta.
12. Use o comando "*kill -9 <PID do filho>*" para matar o processo filho. O que aconteceu?
13. Use o comando "*kill -9 <PID do pai>*" para matar o processo pai. O que aconteceu?
14. Rode o programa novamente. Identifique e mate o processo pai primeiro e em seguida o filho. O que aconteceu?
15. Faz diferença matar o pai ou o filho antes?
16. Compile o programa **threaddemo.c** com o comando:

    gcc threaddemo.c -lpthread -o threaddemo
17. Esse programa usa a biblioteca *POSIX threads library* e é muito similar a **processdemo.c**, mas usa *threads* em vez de processos

18. Rode o programa. O que ele faz? Qual a diferença dele para o programa **processdemo.c**?
19. Qual a diferença de velocidade de saída (medido em linhas por segundo) comparado a **processdemo**? Quem é mais rápido? Você tem uma ideia do porquê?
20. Use o comando ***ps xl*** para verificar que há apenas 1 processo **threaddemo**
21. Investigue o efeito de remover o *loop* infinito no fim do **main()**. O que acontece? Por quê?
22. Modifique o programa **threaddemo.c** para que a manipulação do contador que é feita pelas diferentes *threads* se assemelhe à manipulação realizada pelos processos em **processdemo.c**