

## Laboratório 2: Manipulação de processos

### 1. Objetivos

- Compreender a estrutura de processos em Linux.
- Manipular processos em linguagem de programação.

### 2. Materiais

- Distribuição Linux.
- Ambiente de desenvolvimento.
- Comandos do sistema e bibliotecas de programação.

### 3. Descrição e Métodos

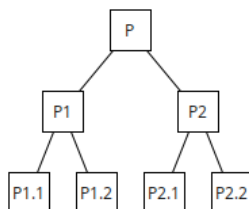
#### Parte 1: Manipulação de processos

1. Execute o comando **ps aux** e identifique três programas do sistema (*daemons*) e três programas do usuário, explicando os valores cada uma das colunas para um de cada tipo (sistema e usuário).
2. Há processos **zombies** executando em seu sistema operacional? Posso eliminá-los do sistema usando o comando **kill -SIGKILL pid\_zombie**? Justifique.
3. Quais os processos com maior utilização de CPU? Quais os processos com maior utilização de memória? Qual o processo do usuário está a mais tempo em execução?
4. Como eu faço para suspender um processo no Linux? Como eu faço para retomar a execução novamente?
5. O que aconteceria se um processo criasse recursivamente processos filhos indefinidamente? Implemente um programa em Linux que faça isso e apresente o resultado. (Sugestão: testar na máquina virtual).

#### Parte 2: Programação

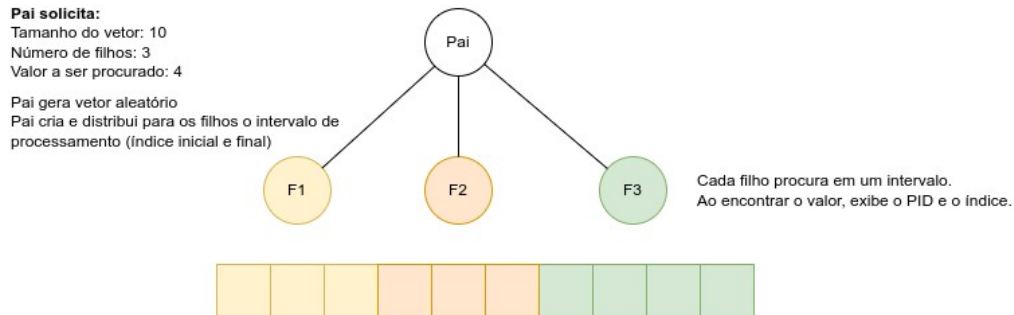
1. Faça um programa que crie uma hierarquia de processos com N níveis ( $1 + 2 + 4 + 8 + \dots + 2^{N-1}$ ) processos. Visualize a hierarquia usando um comando do sistema (**ps tree**).

Exemplo com  $N = 3$ :



2. Faça um programa que receba um comando Linux como parâmetro e execute como um filho do processo. O processo pai deve aguardar o término da execução do comando.

3. Faça um programa que receba um vetor e divida para N filhos partes iguais de processamento para localizar um item. Exibir o PID dos filhos que encontrarem o valor procurado.



4. Faça uma interface de shell simples que fornece um prompt ao usuário para executar comandos do shell do sistema. Se o comando for executado em segundo plano (&), a interface deve possibilitar a execução de outros comandos. Caso contrário, a interface deve esperar o retorno do comando e, em seguida, exibir o prompt novamente.

**Importante:** Fazer um *Makefile* para compilar em Linux/Unix. Use a linguagem de programação C ou C++. Não usem a função *system()*, exceto para executar o *pstree* na questão 1 (opcional).

As atividades da *Parte 1* devem ser entregues em um documento PDF descrevendo as respostas. As atividades da *Parte 2* devem ser entregues o código-fonte das soluções.

Compactar o documento da *Parte 1* e os arquivos da *Parte 2* em um único arquivo (tar.gz) e submeter no Moodle.