Primeiro Trabalho de Sistemas Operacionais – INF 1316 - 2025-1

O primeiro trabalho consiste em programar em linguagem C programas que implementem um **interpretador de comandos** e um **escalonador de programas**. O interpretador deverá ler a especificação do escalonamento de vários processos concorrentes e solicitar ao escalonador a execução dos mesmos de acordo com a especificação. Trata-se de processos Unix de programas C seus. O escalonador então inicia e passa a controlar a execução de todos os processos simultaneamente compatibilizando as políticas de escalonamento. Cada processo que poderá ter uma das seguintes diretrizes de escalonamento:

PRIORIDADE (prioridade = número de 0 a 7 sendo o menor número o de maior prioridade),
ROUND-ROBIN (tempo compartilhado, com fatia de tempo de **1 UT** – unidade de tempo),
REAL-TIME (nesse caso o interpretador deve indicar ao escalonador em qual momento deve ser iniciado e por quantas unidades de tempo deve executar).

Na política de escalonamento REAL-TIME cada processo deverá executar periodicamente (uma vez por minuto), iniciando sua execução, em determinado momento de tempo (I) e deve permanecer executando apenas durante um certo período de tempo (D). Tanto o momento de início como a duração da execução no minuto, são indicados pelos seus parâmetros. Por exemplo, P1 I=20 D=5, indica que P1 deve ser executado a cada minuto e 20 segundos e deve executar por 5 segundos, ou seja, do M:20 até o M:25 (onde M significa cada minuto).

Atenção: Para esse escalonamento REAL-TIME o seu escalonador deverá verificar se um novo processo iniciado não possui parâmetros de escalonamento conflitantes com os dos demais processos já em execução. Por exemplo, se seu sistema já está executando P1 I=0 D=10 e P2 I=20 D=5, então não será possível executar um processo P3 I=11 D=20, pois não haveria como permitir a execução simultânea de P2 e P3. Você também deverá verificar se I+D <= 60 segundos, pois o período de execução de um processo não deve ir além do início do próximo minuto.

No sistema Unix o escalonador executa como parte do núcleo (kernel) mas esse que você irá implementar no trabalho (Escalonador) é um simples processo que vai controlar e coordenar a execução dos processos lidos pelo interpretador de comandos. Ou seja, o seu escalonador vai gerenciar a ordem de execução dos programas utilizando os **sinais (SIGSTOP e SIGCONT)** para controlar quando devem ser suspensos e quando devem prosseguir com a sua execução.

O seu Interpretador deverá ser um processo Unix diferente do escalonador, e como eles precisam se comunicar, você deve usar algum tipo de comunicação inter-processo para passar os dados dos programas para o Escalonador.

O Escalonador deve ser um programa capaz de lidar com as 3 políticas acima (escalonador por PRIORIDADE, ROUND-ROBIN e REAL-TIME), ou seja, deverá implementar as estruturas de dados para tal gerenciamento combinado. Além disso, o seu escalonador deve considerar a seguinte **hierarquia de prioridades** entre as duas politicas:

REAL-TIME >> PRIORIDADE >> ROUND-ROBIN

O interpretador vai ler um comando (execução de 1 programa) por linha de um arquivo **exec.txt** (ASCII). A sintaxe dos comandos a ser analisada pelo interpretador é a seguinte:

Run <nome_programa> P=<prioridade> , processos com PRIORIDADE,
Run <nome_programa>, para o ROUND-ROBIN (processos com tempo compartilhado),
Run <nome_programa> I=<momento-início> D=<tempo-duração>, para REAL-TIME,
onde: momento-início é um inteiro no intervalo [0-59].

Os processos a serem escalonados / controlados devem ser CPU-bound (loops eternos, **exceto os de prioridade que são finitos e executam por 3 UT e os de tempo real que têm tempo de execução determinada**), estes processos devem ser criados por você. O interpretador irá ler de um arquivo de nome "**exec.txt**" quais são os programas a serem executados e deverá iniciá-los exatamente na ordem em que aparecem nesse arquivo, **com um intervalo de 1 UT entre cada um deles**. Ou seja, não pode-se ler toda a entrada e ordenar a lista antes de passar para o escalonador. Os programas são para serem passados um-a-um para o escalonador. A saída deve ser clara o suficiente para mostrar como e porque ocorre a preempção na execução dos processos. Esses arquivos serão objetos de avaliação. Sugestão: Mostrar as filas de prontos a cada preempção, indicar que processo parou de executar e que processo passou a executar.

O trabalho pode ser feito de forma individual ou em dupla. O trabalho será avaliado através de apresentação para o professor em sala de aula e deve ser enviado até a data/hora indicada na tarefa no site de EAD da disciplina. Cada dia de atraso acarreta um desconto de 1 ponto na nota máxima. Devem ser entregues os códigos fonte (interpretador, escalonador e programas de teste) e um relatório (em .pdf) indicando que programas serão executados em seu teste, instruções para compilação, a ordem de entrada para o escalonador e a ordem de execução determinada pelo escalonador (estas são informações da saída dos programas), juntamente com uma análise crítica sobre o que, de fato, ocorreu (se a ordem de execução dos programas foi a esperada, incluindo o que funciona e o que não funciona no seu trabalho). Essas explicações também serão objeto de avaliação. Dica: Faça uma linha do tempo com **120 UT´s** com a ordem de execução esperada e a obtida no teste que está indicado abaixo.

Teste de avaliação da execução do sistema:
Este interpretador/escalonador foi implementado em uma empresa de abastecimento de água que contém um sistema dividido em 6 processos, sendo:
P1 – Aquisição de dados da telemetria (leitura de equipamentos que medem as variáveis do processo em pontos do sistema de abastecimento, em um dado instante), que são: nível de um reservatório ou torre de abastecimento, pressão em uma adutora, vazão em uma adutora, temperatura ambiente, quantidade de cloro e turbidez (%) da água, em determinados pontos do abastecimento, entre outros.
P2 – Tratamento dos dados (conversão de dados binários em unidades de engenharia).
P3 – Armazenamento dos dados tratados em memória.
P4 – Exibição dos dados (em unidades de engenharia) e alarmes (se existirem).
P5 – Armazenamento dos dados em disco (histórico dos dados do abastecimento)
P6 – Exibição dos dados em forma gráfica (para análise de tendências).

Execução dos processos para o Teste (teste ainda outras situações):
Run P1 I=5 T=20
Run P2 I=30 T=5
Run P3 P=1
Run P4 P=3
Run P5
Run P6