Algoritmos de escalonamento

O objetivo deste projeto é escrever um programa para simular o escalonamento de um conjunto de tarefas usando os algoritmos de escalonamento de processador mais conhecidos. O programa deverá suportar os seguintes algoritmos:

- FCFS (*First Come, First Served*)
- *Shortest Job First*
- ~~*Shortest Remaining Time First*~~
- Por prioridade, sem preempção
- Por prioridade, com preempção por prioridade
- *Round-Robin* com *quantum* = 2s, sem prioridade
- ~~*Round-Robin* com *quantum* = 5s, sem prioridade~~
- Round-robin com prioridade e envelhecimento (t_q=2, α=1)

No caso dos escalonamentos *round-robin*, o envelhecimento deve ocorrer a cada quantum e não há preempção por prioridade.

O programa deverá ler os dados dos processos da entrada padrão (*stdin*). Cada linha da entrada corresponde a um processo, com os seguintes dados fornecidos como inteiros separados por um ou mais espaços em branco:

- data de criação
- duração em segundos
- prioridade estática (escala de prioridades positiva)

Um exemplo de entrada para o simulador poderia ser:

```
0 5 2
0 2 3
1 4 1
3 3 4
```

Nesse exemplo de entrada, o processo P1 tem data de criação 0, sua execução dura 5 segundos e sua prioridade é 2. Esse formato de entrada deverá ser respeitado, pois o professor pode testar seu simulador com outros dados de entrada. Observe que essa listagem não precisa necessariamente estar ordenada por data de criação de cada processo.

Para cada algoritmo, o simulador deverá produzir as seguintes informações em sua saída padrão (*stdout*):

- tempo médio de vida
- tempo médio de espera
- número de trocas de contexto
- diagrama de tempo da execução

Para simplificar, o diagrama de tempo de cada execução pode ser gerado na vertical, de cima para baixo (uma linha por segundo), conforme mostra o exemplo a seguir:

```
tempo   P1  P2  P3  P4
 0- 1   ##  --
 1- 2   ##  --  --
 2- 3   --  ##  --
 3- 4   --  ##  --  --
 4- 5   --      ##  --
 5- 6   --      ##  --
 6- 7   ##      --  --
```

```
 7- 8   ##      --  --
 8- 9   --      --  ##
 9-10   --      --  ##
10-11   --      ##  --
11-12   --      ##  --
12-13   ##          --
13-14               ##
```

Deve ser entregue ao professor o código-fonte em C do simulador, devidamente comentado.

Sugestão de implementação (pseudo-código que deve ser ajustado para cada política):

```
início
  lê dados das tarefas da entrada padrão
  imprime cabeçalho do diagrama
  t = 0
  enquanto t < tmax

    se há uma tarefa rodando
      se a tarefa rodando chegou ao fim da execução
        migra a tarefa para o estado terminado
        libera o processador
      senão
        se a tarefa rodando chegou ao fim de seu quantum
        migra a tarefa para a fila de prontos
        libera o processador
      fim se
    fim se

    para cada tarefa i
      se a tarefa i inicia agora (em t)
        coloca a tarefa na fila de prontos
      fim se
    fim para

    se o processador estiver livre
      se houver tarefa na fila de prontas
        escolhe uma tarefa da fila de prontas
        migra essa tarefa para o estado "rodando"
      fim se
    fim se

    imprime linha do diagrama com o estado de cada tarefa

    incrementa o tempo (t++)
    incrementa contadores da tarefa corrente (tempo de vida e de quantum)

  fim enquanto

  calcula e imprime tempos médios
fim
```

Sugere-se definir para cada tarefa:

- identificador
- datas de inicio e de conclusão

- duração (tempo necessário no processador)
- prioridades estática e dinâmica
- estado atual (nova, pronta, rodando, terminada, ...)
- tempo já executado (total e no quantum atual)
- ... (outros campos podem ser necessários para algumas políticas)