[ED237] Round-Robin

Neste problema deverá submeter uma classe **ED237** contendo um programa completo para resolver o problema (ou seja, com o método main).
Pode assumir que no Mooshak **terá acesso a todas as classes base dadas nas aulas, incluindo as de listas, pilhas e filas** (não precisa de as incluir na submissão).
Pode fazer download de todas as classes num arquivo zip ou ver as classes uma a uma.

[PROBLEMAS PARA DOWNLOAD] Para precaver uma possível intermitência na ligação de internet, podem e devem fazer download de todos os problemas em:
`https://mooshak.dcc.fc.up.pt/~edados/teste_parte1/NUM_MECANOGRAFICO.zip` (onde `NUM_MECANOGRAFICO` deve ser substituido pelo vosso número mecanográfico)

Problema

Suponha que tem uma fila de processos para serem executados por um processador e que usa o seguinte algoritmo de *scheduling* com uma estratégia *round-robin:*

1. Pega no primeiro processo da fila e executa-o durante um máximo de *T* segundos
2. Se o processo ainda não terminou, é enviado para o final da fila passando a faltar menos *T* segundos para ele terminar
3. Volta ao primeiro ponto, continuando a processar sempre o primeiro processo da fila até todos os processos terem terminado.

Imagine por exemplo que *T=5* e que tinha a seguinte fila, onde o número indica o tempo restante. O processador iria passar por 7 iterações antes de terminar:

Tempo actual: 0 segundos (0 iterações do processador)

emacs 9	firefox 3	bash 12	dia 5

O processador começa por executar *emacs* durante 5 segundos. Ficam ainda a faltar 4 segundos e esse processo é agora colocado no final da fila:

Tempo actual: 5 segundos (1 iteração do processador)

firefox 3	bash 12	dia 5	emacs 4

Como *firefox* tem menos tempo do que 5 segundos, é executado durante os 3 segundos que precisa e termina. O algoritmo continua a ser executado até terminarem todos os processos:

Tempo actual: 8 segundos (2 iterações do processador) [termina "firefox"]

bash 12	dia 5	emacs 4

Tempo actual: 13 segundos (3 iterações do processador)

dia 5	emacs 4	bash 7

Tempo actual: 18 segundos (4 iterações do processador) [termina "dia"]

emacs 4	bash 7

Tempo actual: 22 segundos (5 iterações do processador) [termina "emacs"]

bash 7

Tempo actual: 27 segundos (6 iterações do processador)

bash 2

Tempo actual: 29 segundos (7 iterações do processador) [termina "bash"]
Fila vazia

A sua tarefa é escrever um método para simular este processo, escrevendo para o ecrã cada vez que termina um processo uma linha no formato `NOME_PROCESSO a b`, onde *a* é o tempo quando o processo terminou e *b* é o número de iterações do processador quando tal aconteceu. Por exemplo, se fosse dada a fila anterior e com *T=5*, o output devia ser:

```
firefox 8 2
dia 18 4
emacs 22 5
bash 29 7
```

Dicas: *(é livre para fazer fazer como quiser, mas é sugerido fazer da seguinte maneira)*:

- Crie uma classe *Process* para conter um processo, com dois atributos: *name* e *time* (tempo restante)
- Para representar os processos, use uma fila (*MyQueue<Process>*), ou uma lista circular (*CircularLinkedList<Process>*).
- Precisa de continuar a processar enquanto a fila (ou lista) não estiver vazia, dando sempre tempo de execução ao primeiro processo da fila (ou lista)
- Cuidado com os casos onde o processo tem menos tempo restante do que *T*

Input

A primeira linha contém um inteiro **T**, o tempo de execução máximo por iteração. A segunda linha contém um inteiro **N**, o número de processos da fila. Seguem-se **N** linhas com os processos no formato `NOME_PROCESSO TEMPO_NECESSÁRIO`. O nome é constituído unicamente por letras minúsculas e o tempo é um inteiro.

Output

O output deve conter **N** linhas, descrevendo os processos pela ordem em que foram terminando no formato `NOME_PROCESSO TEMPO_TERMINAÇÃO NUM_ITERAÇÕES`.

Exemplo de Input/Output

Input	Output
5 4 emacs 9 firefox 3 bash 12 dia 5	firefox 8 2 dia 18 4 emacs 22 5 bash 29 7

Teste Prático de Estruturas de Dados (CC1007)
8 de Junho de 2020
