Tarefa 14 - Fila de roteador

Publicado em 26/10/2021

Prazo de entrega recomendado: ***02/11/2021***

Você deve implementar uma fila de prioridade para balancear tráfego de um roteador.

Um importante componente de uma rede de computadores são os *roteadores*. Eles são responsáveis por encaminhar pacotes de dados que chegam até eles. Em cada roteador, existe uma fila de prioridades que controla a ordem em que os pacotes são processados e encaminhados. Essa fila serve para garantir a qualidade da rede nos momentos que ela estiver sobrecarregada. Mesmo que o roteador não consiga encaminhar todos os pacotes que recebeu, entregará pelo menos os mais importantes.

Você precisa implementar o *firmware* de um roteador para um novo protocolo de rede. Nesse protocolo, os pacotes chegam ao roteador com uma prioridade predefinida além de uma taxa de incremento. O roteador deve guardar os pacotes enquanto a rede estiver congestionada. Sempre que a rede se desobstrui, o roteador emite um *tick* e os k pacotes com prioridades mais altas são encaminhados.

Logo depois que os pacotes são enviados, a prioridade de cada pacote remanescente é aumentada de sua taxa de incremento. Por exemplo, se um pacote remanescente `A` tinha prioridade 5 e incremento 1 e outro pacote `B` tinha prioridade 2 e incremento 3, depois do primeiro *tick*, esses pacotes terão prioridades 6 e 5, respectivamente. Se depois do próximo *tick* esses pacotes ainda não forem encaminhados, então o pacote `A` passará a ter prioridade 7 e pacote `B` passará a ter prioridade 8, passando à frente na fila de prioridades.

Como o roteador tem memória limitada, ele só tem capacidade para armazenar até m pacotes na fila. Infelizmente, isso significa que, enquanto a rede está muito congestionada e a fila está cheia, os pacotes que chegam são descartados, independentemente de suas prioridades.

Entrada

A primeira linha contém a configuração do roteador com os números k e m. Cada linha seguinte contém três números inteiros e representa um pacote ou um *tick*. Um pacote tem a forma `<id> <prioridade> <incremento>`, em que `<id>` é um inteiro positivo identificando o pacote. Um *tick* é representado por uma linha `0 0 0`.

A entrada deve ser lida até o fim do arquivo *EOF*.

Exemplo de entrada

```
11 99

1 1 1
2 2 1
3 3 1
4 4 1
5 5 1
6 6 1
7 7 1
8 8 1
0 0 0

9 9 1
10 10 1
11 11 1
12 12 1
13 13 1
14 14 1
15 15 1
0 0 0
```

Saída

A cada *tick*, deve ser impressa uma linha com o número do *tick* e uma linha com os dados atualizados `<id> <prioridade> <incremento>` de cada pacote enviado. As colunas podem ser separadas por um caractere de tabulação ou espaço. Veja o exemplo abaixo.

Exemplo de saída

```
TICK 1
8   8   1
7   7   1
6   6   1
5   5   1
4   4   1
3   3   1
2   2   1
1   1   1

TICK 2
15  15  1
14  14  1
13  13  1
12  12  1
11  11  1
10  10  1
9   9   1
```

Dicas e sugestões

Para ler uma sequência de linhas até o final de arquivo, você pode verificar o retorno do `scanf`, como no exemplo.

```
while (scanf(...) != EOF){ ... }
```

Escreva o algoritmo antes de programar e teste cada parte do programa com calma: verifique se a leitura está correta, teste **separadamente** cada operação da fila de prioridade utilizada, etc. Sempre é útil criar também uma função para mostrar a estrutura de dados. Enquanto está testando o algoritmo, chame essa função em cada iteração do laço principal, ou melhor, aprenda a configurar um *breakpoint* no início de cada iteração e chame essa função a partir do `gdb`!

Critérios

Você deve implementar uma fila de prioridades usando um *max heap*. Sua implementação deve ser eficiente, evitando passar pelos elementos da fila desnecessariamente.

Correção

Esta tarefa será corrigida automaticamente sempre que você realizar um `git push`. Depois de terminada a tarefa, deve-se utilizar o botão na interface de notas para solicitar a correção de um monitor.

Turma AB: O peso desta tarefa será 4.

← Anterior Voltar Próximo →