ESTRUTURA DE DADOS - 04A - 2023.2

Descrição | Enviar | Editar | Visualizar envios

Buffering...

Data de entrega: sexta, 8 dez 2023, 23:59
Número máximo de arquivos: 3
Tipo de trabalho: Trabalho individual

Descrição

Quando estamos assistindo um vídeo na Internet que acaba sendo reproduzido mais rápido do que nossa conexão é capaz de baixar seu conteúdo, geralmente o reprodutor dá uma pausa e mostra a informação de que ele está aguardando um pouco mais do conteúdo estar disponível. Isso é o que chamamos de *buffering*.

O que o reprodutor está fazendo é na verdade aguardando que uma porção razoável de conteúdo seja baixado (não apenas qualquer quantidade) para evitar que a reprodução fique "engasgando" demais (executa e para muito frequentemente).

Para que isso aconteça o conteúdo disponível (mas não reproduzido ainda) do vídeo deve ser armazenado em um *buffer*, que deve ser uma estrutura *FIFO* para que os quadros do vídeo não sejam trocados de ordem. Além disso, para evitar demora associada a alocação e desalocação frequente dessa estrutura, ele é criado apenas uma vez como uma fila circular e preenchido / esvaziado sob demanda.

Nesta questão você deverá implementar a estrutura de Fila Circular e usá-la para simular essa situação. As entradas consistirão de "solicitações" feitas pela interface de rede e pelo reprodutor com relação aos conteúdos baixados ou reproduzidos, e serão descritas abaixo.

Operações

Vamos considerar que cada quadro a ser exibido pelo reprodutor de vídeo seja representado por exatamente *Q bytes* e que o número de quadros desenhados por segundo nesse vídeo é dado por *D* quadros. Consideraremos também que, uma vez em *buffering*, o reprodutor deverá coletar dados o suficiente para *T* segundos de reprodução ininterrupta antes de voltar a reproduzir o vídeo. Além disso, que o *buffer* utilizado tem capacidade máxima para *B bytes*.

As operações que serão simuladas, e o comportamento que elas devem produzir, são:

- Chegada de dados via rede, sendo informados quantos e quais "*bytes*" chegaram.
  - Esses *bytes* devem ser inseridos no *buffer*.
  - Caso o *buffer* já esteja cheio, os dados mais antigos devem ser sobrescritos.
    - Observe que, para evitar corrupção de dados, devemos descartar sempre os *bytes* em grupos, cada grupo representando um quadro.
- Solicitação de um quadro pelo reprodutor.
  - Caso haja conteúdo suficiente no *buffer* para compor um quadro, os *bytes* correspondentes devem ser consumidos.
  - Caso não haja, deve-se aguardar e realizar o *buffering*.
    - Daí em diante, deve-se continuar aguardando até que o conteúdo mínimo seja armazenado.

Observações

- Você deverá implementar também a leitura e escrita de dados (entrada e saída).
- Nem todos os casos de testes serão exibidos.
- **Implemente suas próprias versões das estruturas necessárias.**

Dados

As operações serão informadas em sequência, cada uma com todas as informações necessárias. A cada operação processada você deverá fornecer a saída correspondente ao seu processamento. Os dados da simulação e seu formato, além das saídas esperadas, são descritos nas subseções a seguir.

Esses dados serão fornecidos através da entrada padrão (lidas com `Scanner`, por exemplo) e as saídas deverão ser dadas através da saída padrão (escritas com `System.out.printf`, por exemplo).

Entrada

A entrada consiste de um inteiro positivo *O*, representando a quantidade de operações que serão simuladas, seguido dos valores de *Q*, *D*, *T* e *B*. A partir daí, seguirão *O* linhas cada uma contendo a operação a ser simulada e as informações associadas.

As operações serão dadas como:

- o caractere "`d`", representando a operação de recebimento de dados, seguido do número de bytes *N* que foram recebidos e uma *string S* com *N* caracteres; ou
- o caractere "`q`", representando a operação de solicitação de um quadro.

Os intervalos de valores de cada uma das informações são:

- 1 <= *O* <= 500
- 1 <= *Q* <= 16
- 1 <= *D* <= 30
- 1 <= *T* <= 5
- 1 <= *B* <= 4096
- 1 <= *N* <= 512

Por fim, na *string S* há apenas caracteres numéricos (dígitos entre `0` e `9`), com cada um deles representando o conteúdo de um *byte*.

Saída

Em resposta ao processamento de cada operação, você deverá escrever as seguintes saídas, uma por linha:

- considerando a operação de recebimento de dados:
  - `OK`, caso todos os bytes possam ser guardados sem sobrescrita; ou
  - `SOBRESCRITA K`, substituindo *K* pela quantidade de *bytes* que tenham sido sobrescritos em razão dessa operação.
- considerando a operação de solicitação de um quadro:
  - `BUFFERING`, caso o processo de *buffering* tenha iniciado nesta operação ou caso já havia iniciado mas ainda não há dados suficientes para continuar; ou
  - `QUADRO ...`, substituindo os três pontos pelos caracteres correspondentes a um quadro, caso haja dados o suficiente.

Exemplos

Exemplo 1

Entrada:

```
14 4 1 2 512
q
d 15 012345678901234
q
q
d 10 5678901234
d 1 5
q
q
q
q
q
d 3 678
q
```

Saída:

```
BUFFERING
OK
QUADRO 0123
QUADRO 4567
OK
OK
QUADRO 8901
QUADRO 2345
QUADRO 6789
QUADRO 0123
BUFFERING
OK
BUFFERING
```

Exemplo 2

Entrada:

```
9 2 2 5 32
d 10 0123456789
q
d 15 012345678901234
q
d 10 5678901234
d 10 5678901234
q
q
q
```

Saída:

```
OK
QUADRO 01
OK
QUADRO 23
OK
SOBRESCRITA 10
QUADRO 45
QUADRO 67
QUADRO 89
```

Exemplo 3

Entrada:

```
10 4 2 3 32
q
d 4 0123
d 4 4567
q
d 4 8901
q
d 12 234567890123
q
q
q
```

Saída:

```
BUFFERING
OK
OK
BUFFERING
OK
BUFFERING
OK
QUADRO 0123
QUADRO 4567
QUADRO 8901
```