MC202D - Estruturas de Dados

1º Semestre de 2017

Professor: Rafael C. S. Schouery

Monitores: Guilherme Colucci Pereira (PED)

Marcelo Pinheiro Leite Benedito (PED)

Erik de Godoy Perillo (PAD)

Laboratório 5 - Salvando a Páscoa

1. O Problema

Todos os anos, é a mesma correria na fábrica de ovos de Páscoa do Sr. Coelho, afinal, não é nada fácil produzir milhões de ovos em um período tão curto de tempo (ele só pode começar depois do Carnaval). Com todas as matérias-primas em mãos, os ovos são fabricados em várias máquinas, paralelamente, que são operadas por duendes ajudantes, cedidos pelo Papai Noel. Os ovos possuem diferentes medidas, sabores e enfeites, então, apesar de todas as máquinas serem idênticas, os ovos possuem diferentes tempos de preparo.

No início do expediente, os duendes tomam suas posições nas máquinas e esperam as ordens do Sr. Coelho, que indicará quais ovos serão produzidos. O expediente é dividido em turnos discretos, que marcam o início da produção de ovos e seu tempo de produção. Por questões de logística, ele libera uma lista contendo, para cada ovo, o turno de **início da produção** (que pode iniciar nesse turno ou em um turno posterior) e seu **tempo de produção**. Alguns ovos são marcados como urgentes, pois devem começar sua produção imediatamente (assim que chegar seu turno de início).

Neste laboratório, usaremos uma estrutura de dados chamada **deque** (*double ended queue)*, uma generalização da fila, onde elementos podem ser inseridos e removidos de ambas as extremidades, comumente chamadas de frente e trás, ou cabeça e cauda. Portanto, temos quatro operações principais:

- *insere_frente(elemento)*
- *remove_frente()*
- *insere_tras(elemento)*
- *remove_tras()*

Para obtermos uma implementação eficiente desta estrutura, guardaremos os elementos em uma **lista duplamente encadeada**, porque as principais operações que devem ser implementadas são realizadas em tempo constante com essa escolha.

O seu trabalho é simular a produção de ovos do Sr. Coelho: cada máquina possuirá seu próprio **deque**, que será preenchido com os pedidos dos ovos. Assim que chegar seu turno de início, cada novo pedido será adicionado ao final do deque da máquina seguinte, circularmente, iniciando na primeira até a última, para então voltar à primeira e assim por diante, de forma a balancear as "filas" de cada máquina. Um pedido que está na primeira posição do deque de uma máquina está sendo produzido e só ficará pronto (e será removido) assim que passar sua quantidade de turnos de produção. Se um pedido é adicionado a uma máquina vazia, neste mesmo turno já é contado um turno de sua produção.

Às vezes, a estratégia de balanceamento do Sr. Coelho não funciona e algumas máquinas podem ficar vazias por terminar a produção dos seus ovos enquanto outras máquinas estão com vários pedidos em espera. Quando isso acontecer, a máquina vazia (ou seja, que tem seu deque vazio) deverá "roubar" pedidos que já estão alocados em outras máquinas (que estão no fim de seus deques) para começar a produzi-los. As máquinas vazias procuram, em ordem crescente de seus identificadores, máquinas com deques com mais de um pedido (o primeiro já está sendo produzido) **sempre** na ordem 0, 1, 2, …, m-1, ou seja, se em um turno ela roubou um pedido na máquina 2, quando for roubar outro, começa a procurar da máquina 0 novamente. No mesmo turno em que um pedido é roubado já é contado um turno de sua produção na nova máquina. Além disso, é roubado **apenas um** pedido por vez.

Quando chega um turno de início de um pedido urgente, ele será alocado na mesma ordem de um pedido normal, mas no início do deque daquela máquina, parando a produção do ovo atual que pode, inclusive, também ser um pedido

urgente (depois a produção continua levando em consideração o número de turnos que já passaram). Note que uma máquina pode roubar um pedido que já se iniciou mas parou para produzir um pedido urgente: nesse caso, também deve se levar em conta os turnos de produção já realizados.

A cada turno, as operações são feitas na seguinte ordem:

1. Verifica se o ovo está pronto, se estiver, imprima a saída;
2. Distribui os ovos que começam sua produção neste turno, levando em conta os métodos descritos e as urgências dos pedidos;
3. Se existirem máquinas vazias, estas procuram pedidos para roubar em máquinas com mais de um pedido em suas filas;
4. Por fim, incremente o número de turnos de produção de todos os pedidos que estão sendo produzidos.

Seu objetivo é informar ao Sr. Coelho, a cada turno, qual máquina acabou de produzir qual ovo.

2. Entrada

São dados dois números inteiros, **n** e **m**, representando o número de pedidos de ovos e máquinas, respectivamente. Em seguida, serão informados **n** triplas de números inteiros, um para cada pedido, que indicam o **tempo de início** em que o pedido estará pronto para começar a ser executado, seu **tempo de produção** e se este pedido é **urgente**: quando este valor é 1, o ovo é sinalizado como urgente, quando o valor é 0, o ovo é um pedido normal.

- Observações
    - Os pedidos são indexados de **0** até **n-1**;
    - As máquinas são indexadas de **0** até **m-1**;
    - O turno começa do **0**;
    - Os pedidos são fornecidos em ordem crescente de tempo de início.

Exemplo de entrada:

```
5 2
0 3 0
1 18 0
16 8 0
21 3 1
23 17 0
```

3. Saída

A cada turno em que terminar a produção de um ovo, você informará, em ordem crescente de identificador das máquinas, qual pedido foi feito em qual máquina, no formato: **Turno X: (id_máquina1, id_pedido1), (id_máquina2, id_pedido2)**. Você deve omitir os turnos que não têm o fim da produção de ovos.

Por exemplo, se temos três ovos com identificadores 0, 1 e 2 sendo produzidos e eles ficaram prontos nos turnos 3, 3 e 9 pelas máquinas 0, 1 e 0, respectivamente, então a saída deve ser a seguinte. Note que os turnos em que não houveram finalização de produção foram omitidos.

```
Turno 3: (0, 0), (1, 1)
Turno 9: (0, 2)
```

- Observações
    - Note que na saída há um espaço entre o identificador da máquina e o do pedido;
    - Note que, quando há mais de um pedido sendo finalizado no mesmo turno, eles são **separados** por vírgula.

Exemplo de saída:

```
Turno 3: (0, 0)
Turno 19: (1, 1)
Turno 24: (0, 2), (1, 3)
Turno 41: (0, 4)
```

4. Informações

- Este laboratório possui peso 3.
- A submissão da sua solução deverá conter múltiplos arquivos:
    - **lab5.c**: código cliente, contém a resolução do problema
    - **Deque.h**: interface da estrutura de dados
    - **Deque.c**: implementação da interface
- Você poderá utilizar o **Code::Blocks** para montar seu projeto ou utilizar o **Makefile** disponibilizado na página do laboratório:
    - Para compilar seu projeto, basta utilizar o comando ‘make’ em um terminal do Linux.
    - Veja mais instruções na página da disciplina