Filas Circulares Encadeadas

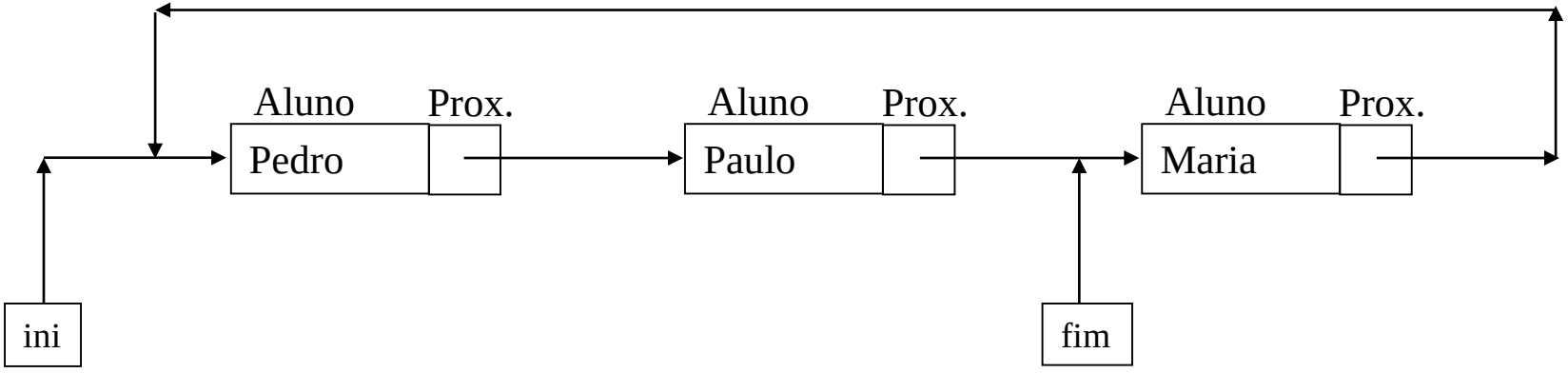
Filas simples partem do princípio que é possível processar 100% de um pedido antes de seguir para o próximo elemento da fila.

Existem situações onde o processamento pode atender outras requisições enquanto a liberação de um recurso está sendo aguardada.

Filas Circulares Encadeadas

É mais fácil representar uma fila como uma sequência circular do que como uma sequência linear.

Como uma sequência linear encadeada, uma fila é especificada por duas referências, uma para o início da fila e outra para seu final.

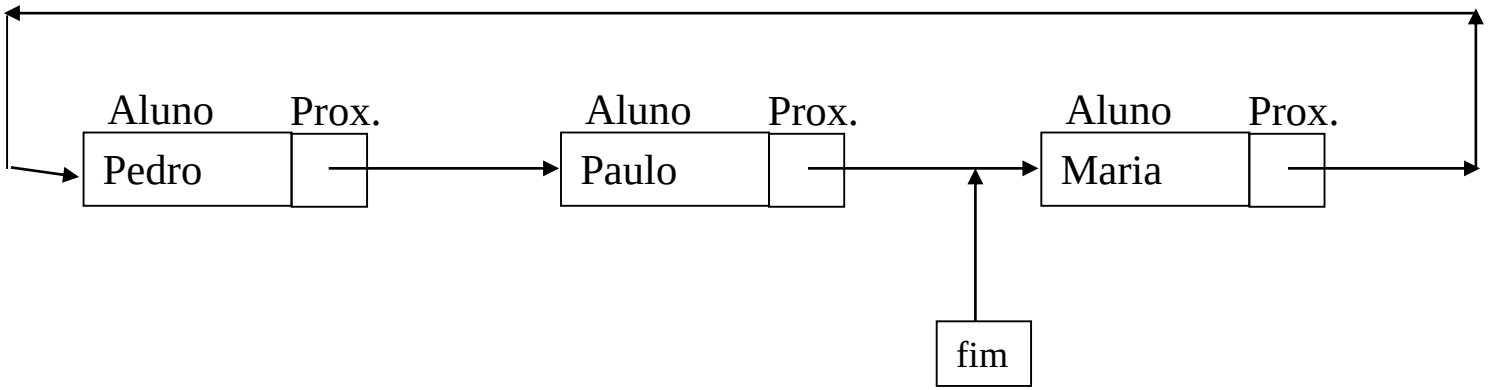


Filas Circulares Encadeadas

Entretanto, usando uma fila circular, uma fila pode ser especificada por uma única referência para essa fila, ou seja, uma referência para o final da fila e o nó seguinte (*prox*) é seu início.

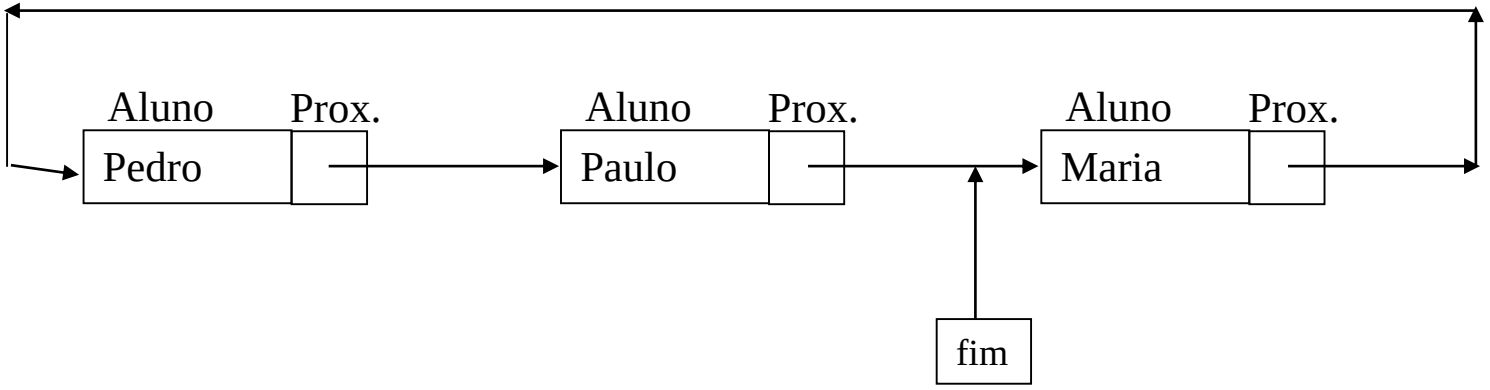
Suponha que seja feita uma pequena mudança na estrutura de uma fila, de modo que o campo próximo no último nó contenha uma referência de volta para o primeiro nó no lugar de *null* (esse tipo de fila é chamado de fila circular). Assim, a partir de qualquer ponto dessa fila, será possível atingir qualquer outro ponto na fila.

Se começar em determinado nó e atravessar a fila inteira, terminará, em última análise, no ponto inicial.



Filas Circulares Encadeadas

Observe que uma fila circular não tem um "primeiro" ou um "último" nó natural. Precisa-se, portanto, estabelecer um primeiro e um último nó por convenção. Uma convenção útil é permitir que uma referência externa para a fila circular aponte para o último nó, e que o nó seguinte se torne o primeiro nó.



Filas Circulares Encadeadas

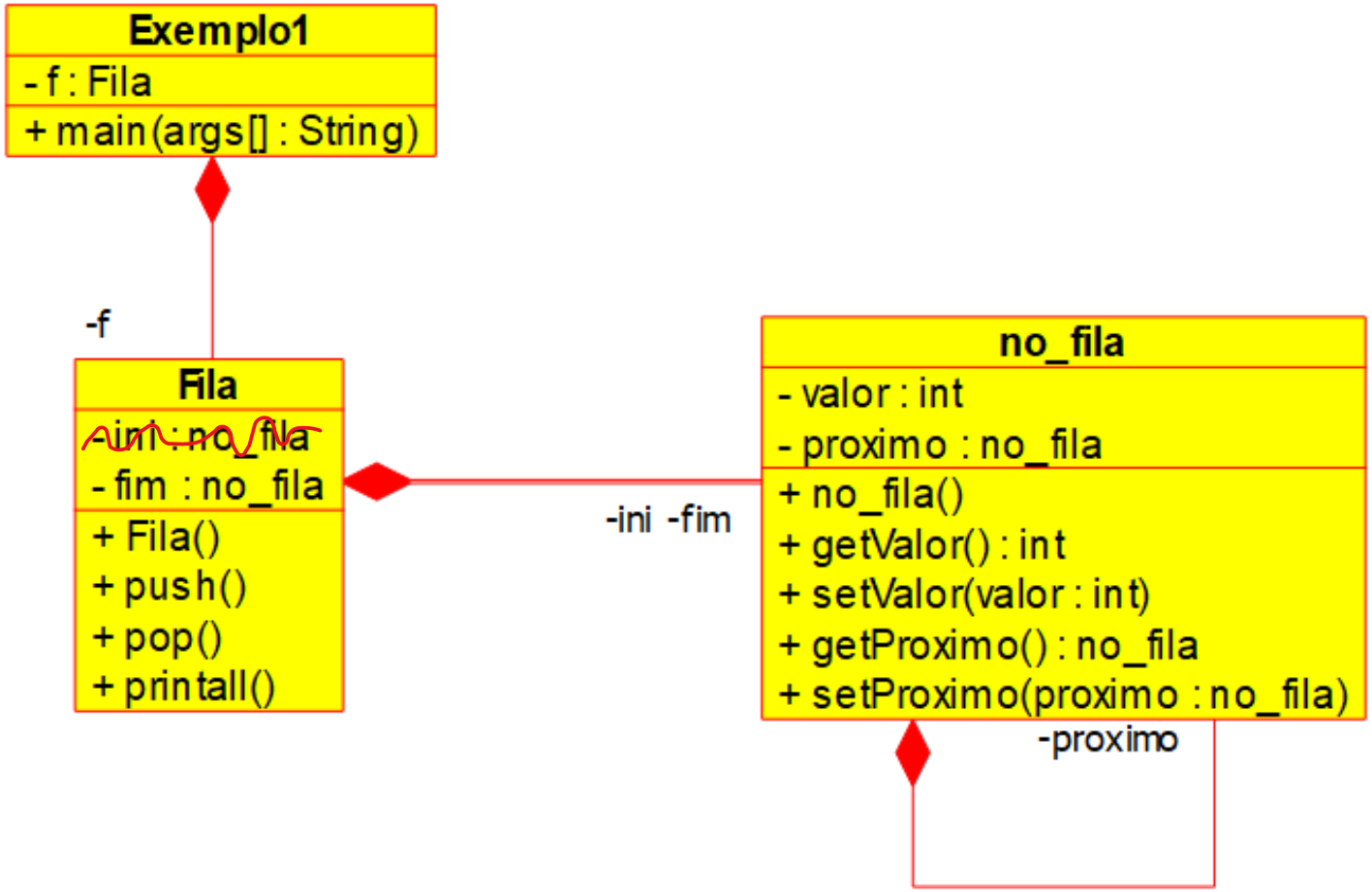
Um exemplo de fila circular é o gerenciamento de processos num sistema operacional. Cada processo ganha um tempo da CPU e em seguida volta para o final da fila.

Filas Circulares Encadeadas

Uma fila circular possui três operações básicas:

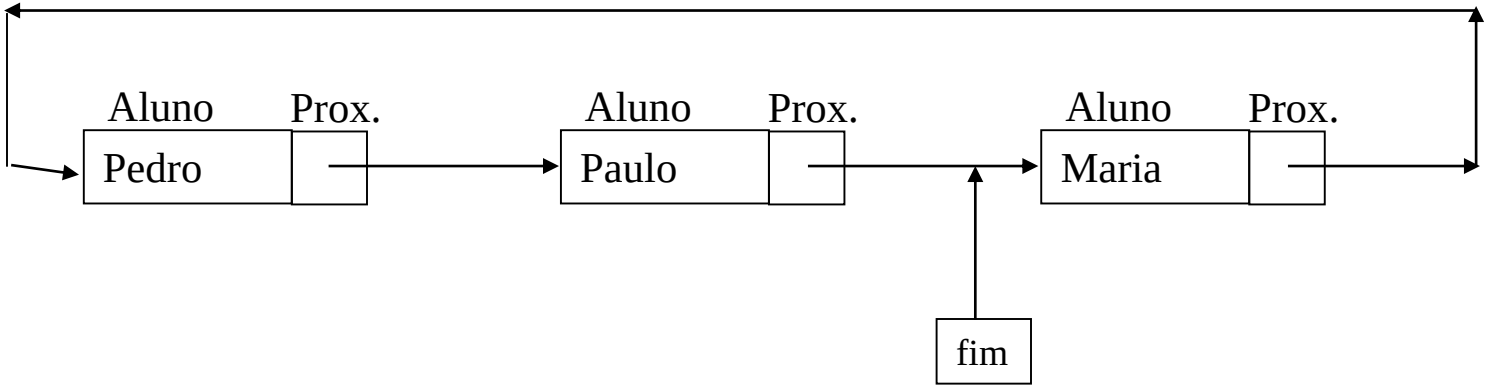
- Inserir um elemento no final da fila (*Push*)
- Retirar o primeiro elemento da fila (*Pop*)
- Ler o primeiro elemento sem retirar (*Front*)

Filas Circulares Encadeadas



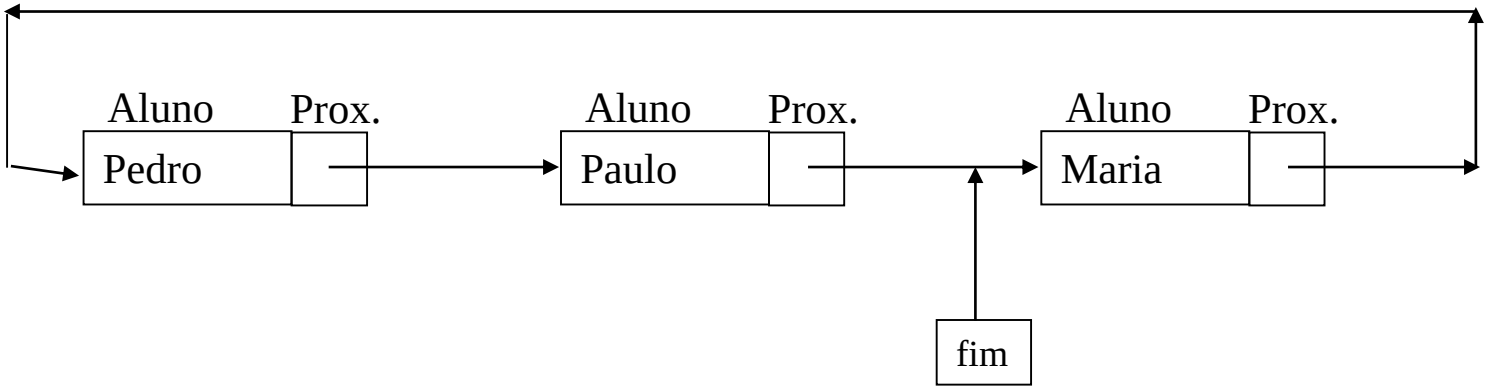
Exercício 01: Leitura Circular Encadeada

Construa um programa que insere nomes em uma fila circular.



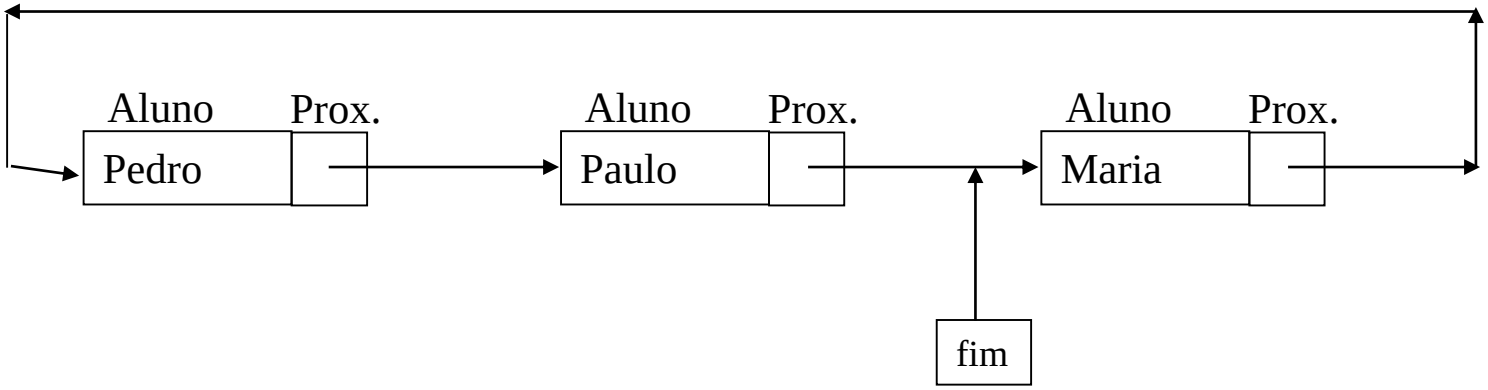
Exercício 02: Leitura/Impressão Circular Encadeada

Construa um programa que insere nomes em uma fila circular e que **permita a impressão do "primeiro elemento"** da fila.



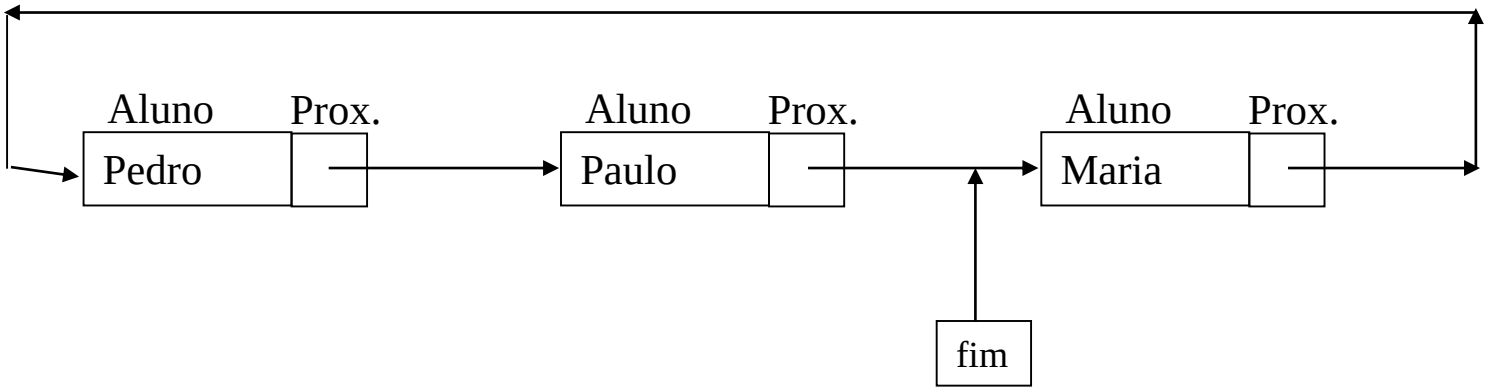
Exercício 03: Leitura/Impressão Circular Encadeada

Construa um programa que insere nomes em uma fila circular e imprime a cada inserção, o "primeiro" e o "**último**" nome da fila.



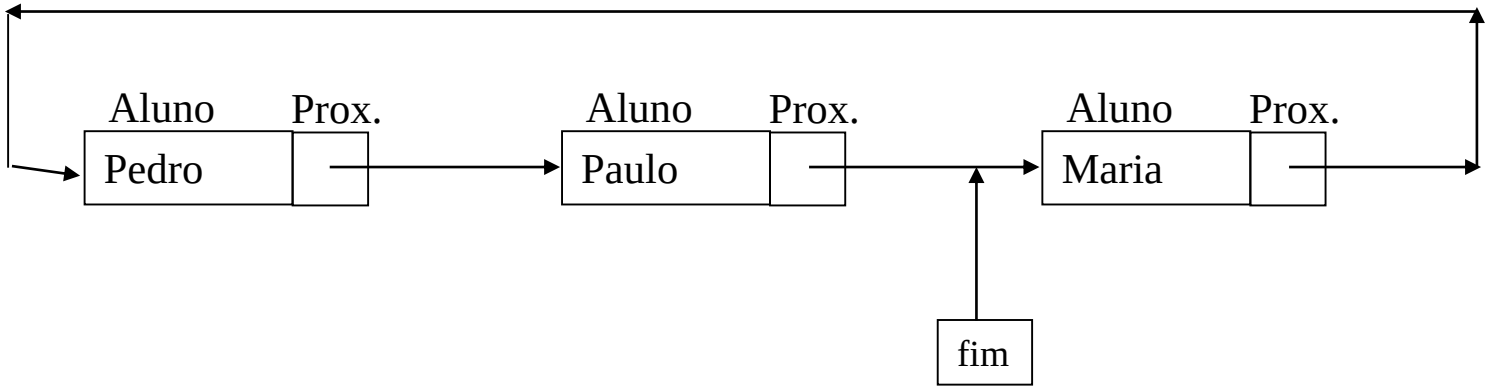
Exercício 04: Leitura/Impressão Circular Encadeada

Construa um programa que insere nomes em uma fila circular e imprime a cada inserção, a **fila completa**, depois o "primeiro" e "último" elemento da fila.



Exercício 05: Exclusão Circular Encadeada

Construa um programa que insere nomes em uma fila circular e imprime a cada inserção, a fila completa, o “primeiro” e o “último” elemento da fila. E, permitir também, a **exclusão** de um nome em uma fila circular.



Exercício 06: Fila Circular Encadeada

Em um cassino, existe uma máquina de caça níqueis que dará 1 milhão de dólares.
Porém, é permitido que os jogadores apostem apenas uma única vez e depois voltem para o final da fila. Em cada jogada, os jogadores gastam 5 dólares.
Faça um programa que simule o caça-níquel e a cada requisição subtraia 5 dólares do participante. Quando o valor em dólares for menor que 5 o participante deve ser retirado da fila.
Utilize uma fila circular para armazenar os dados.
Sabe-se que desde o início do seu funcionamento naquela noite, que haverá um premiado em uma determinada jogada.
Quem ganhou 1 milhão de dólares?

Exercício 07 (Bônus): Fila Circular Encadeada

O PROBLEMA DE JOSEPHUS:

Examina-se um problema que pode ser solucionado de maneira simples usando uma fila encadeada circular. Ele é conhecido como problema de Josephus e postula um grupo de soldados circundado por uma força inimiga esmagadora. Não há esperanças de vitória sem a chegada de reforços, mas existe somente um cavalo disponível para escapar. Os soldados entram num acordo para determinar qual deles deverá escapar e trazer ajuda. Eles formam um círculo e um número *n* é fornecido. Um de seus nomes é fornecido também. Começando pelo soldado cujo nome foi fornecido, eles começam a contar ao longo do círculo no sentido permitido por uma estrutura fila. Quando a contagem alcança *n*, esse soldado é retirado do círculo, e a contagem reinicia com o soldado seguinte. O processo continua de maneira que, toda vez que *n* é alcançado, outro soldado sai do círculo. Todo soldado retirado do círculo não entra mais na contagem. O último soldado que restar deverá montar no cavalo e escapar.

Considerando um número *n*, a ordem dos soldados no círculo e o soldado a partir do qual começa a contagem, o problema consiste em determinar a sequência na qual os soldados são eliminados do círculo e identificar o soldado que escapará.

Exercício 07 (Bônus): *Continuação*

A entrada para o programa é o nome dos soldados, que será o sequenciamento do círculo, começando pelo soldado a partir do qual a contagem deve ser iniciada. Ao final, o programa deve imprimir os nomes na sequência em que são eliminados e o nome do soldado que escapará.

Por exemplo, suponha que n = 3, o soldado a partir do qual começa a contagem seja o A, e que existam cinco soldados, chamados A, B, C, D e E. Contamos três soldados a partir de A para que C seja eliminado primeiro. Em seguida, começamos em D e contamos D, E e novamente A para que A seja eliminado a seguir. Depois, contamos B, D e E (C já foi eliminado) e, finalmente, B, D e B, de modo que D seja o soldado a escapar.

Uma fila encadeada circular, na qual cada nó representa um soldado, é uma estrutura de dados natural para usar na solução deste problema. É possível alcançar qualquer nó a partir de qualquer outro, percorrendo o círculo. Para representar a remoção de um soldado do círculo, um nó é eliminado da fila circular. Por último, quando só restar um nó na lista, o resultado será determinado.

Elabore um programa em Java para o problema de Josephus.

Exercício Bônus 08

Crie um menu com as seguintes opções encadeadas:

1) Implementação por fila circular de pilhas
2) Implementação por pilha de filas circulares
3) Implementação por fila circular de filas circulares

Escreva rotinas para implementar as operações corretas para cada uma das estruturas de dados. Considere que cada item (nó) consiste no armazenamento de um inteiro.

Inclusão na Fila Circular Encadeada

```
public void push(){
    no_fila temp_no = new no_fila();
     if(temp_no != null){
         temp_no.nome = JOptionPane.showInputDialog("Nome: ");
         if (fim != null){
           temp_no.proximo = fim.proximo;
           fim.proximo = temp_no;
           fim = temp_no;
         }else{
           temp_no.proximo = temp_no;
           fim = temp_no;
         }
         printall();
     }
}
```

Impressão na Fila Circular Encadeada

```
public void printall(){
     if (fim == null) {
       JOptionPane.showMessageDialog(null,"Fila vazia...");
       return;
     }
     no_fila temp_no = fim.proximo;
     String str = "Fila:\n";
     do{
        str += temp_no.nome + "\n";
        temp_no = temp_no.proximo;

     }while(temp_no != fim.proximo);
     JOptionPane.showMessageDialog(null,str);
     ...(null,"Primeiro: " + fim.proximo.nome);
     ...(null,"Ultimo: " + fim.nome);
}
```

Exclusão na Fila Circular Encadeada

```
public void pop(){
     if (fim == null)
        JOptionPane.showMessageDialog(null,"Fila vazia...");
     else{
        ...(null,"Valor removido: " + fim.proximo.nome);
        if (fim.proximo != fim){
           fim.proximo = fim.proximo.proximo;
        }else
             fim = null;
        printall();
     }
}
```