Estrutura de Dados – 1º semestre de 2020

Professor Mestre Fabio Pereira da Silva

Fila

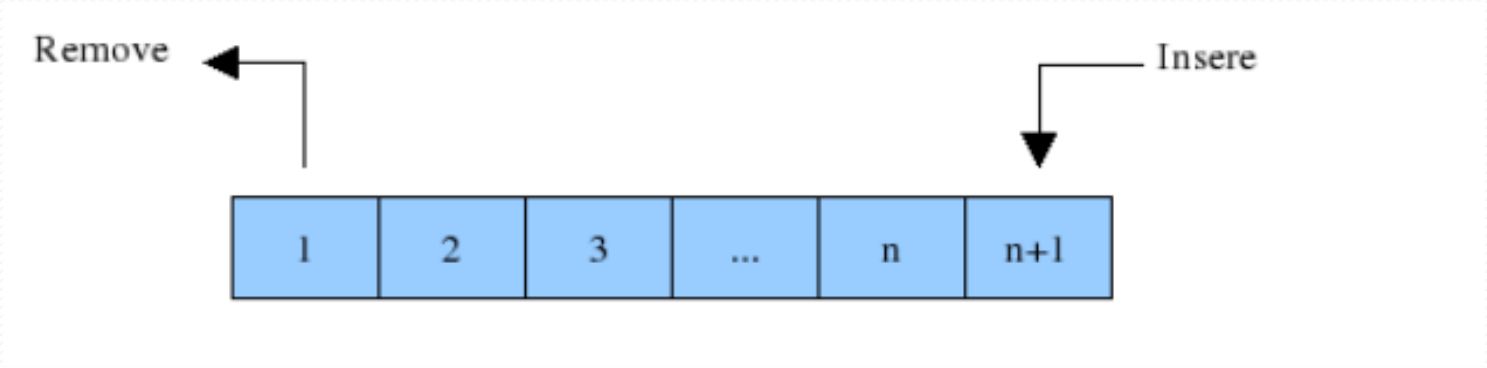
- Tipo abstrato de dados, em que o primeiro elemento inserido é o primeiro elemento retirado.
  - (FIFO – *First in First Out*)
- Aplicação: Sistemas operacionais: processamento, impressão de arquivos.
- Exemplos: Fila bancária, caixa lotérica, fila do cinema.

Fila

- São listas em que as operações de remoção e inserção ocorrem sempre em locais específicos
- A inserção é feita sempre no final da lista
- A remoção é feita sempre no início da lista
- **Em função disso uma fila assume a condição FIFO**

Fila

- São estruturas de dados do tipo FIFO (first-in first-out), onde o primeiro elemento a ser inserido, será o primeiro a ser retirado, ou seja, adiciona-se itens no fim e remove-se do início.



Fila

- Uma fila é caracterizada por ser uma linha de espera que cresce somando elementos ao seu final e que diminui tomando elementos da sua frente.
- Em uma extremidade os nós são somente adicionados, enquanto que na outra extremidade da fila os nós são apenas removidos.

Fila

- A estrutura de fila é análoga ao conceito que temos de filas em geral. O primeiro a chegar é sempre o primeiro a sair, e a entrada de novos elementos sempre se dá no fim da fila.
- Em computação vemos este conceito sendo implementado em filas de impressão.
- Assim como as pilhas, uma fila também pode ser implementada por meio de um vetor ou de uma lista encadeada.

Fila

- Operações:
- Adicionar elemento (método enqueue)
- Remover elemento (método dequeue)
- Verificar se a fila está vazia
- Verificar se a fila está cheia

Fila

- As filas são frequentemente usadas em simulações, uma vez que existe uma *teoria das filas* bem desenvolvida e matematicamente sofisticada na qual vários cenários são analisados e modelos que usam filas são construídos.

Fila

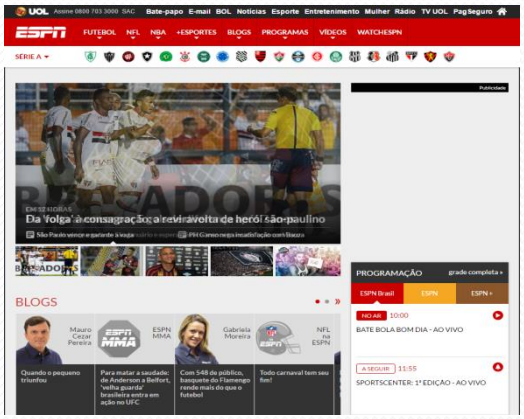
- O acesso aos elementos da fila é realizado através das posições "entrada" e "saída" – as demais posições não são visíveis.
  - Estrutura do tipo FIFO (*first in, first out*).
  - Acesso: através de dois apontadores.
  - Manipulação: apenas a entrada e a saída são visíveis.

Leitura – posição "saída"	**Escrita – posição "entrada"**
• elemento ← f(*j*)	• f(*i*) ← elemento
• atualiza índice de saída *j*	• atualiza índice de entrada *i*

  - Fila cheia: índice de entrada *i* > índice máximo da fila.
  - Fila vazia: índice de entrada *i* = índice de saída *j*.

Fila

- Considere um algoritmo que armazena as páginas que você acessa na Internet de tal modo que seja possível retornar à primeira página acessada refazendo o mesmo caminho. Como os dados devem ser organizados?



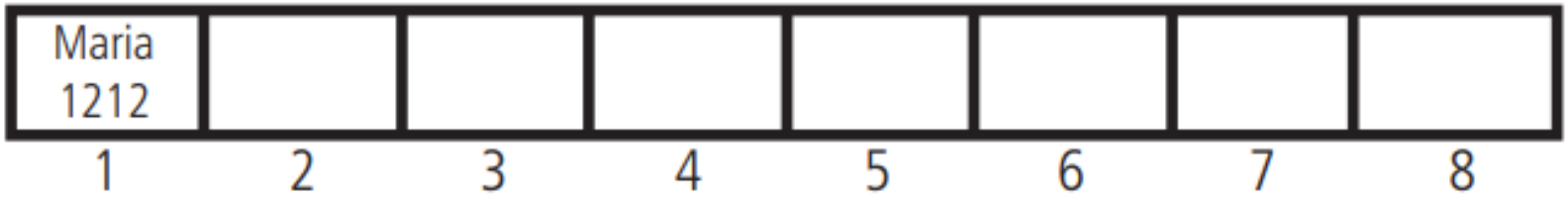
(1)

(2)

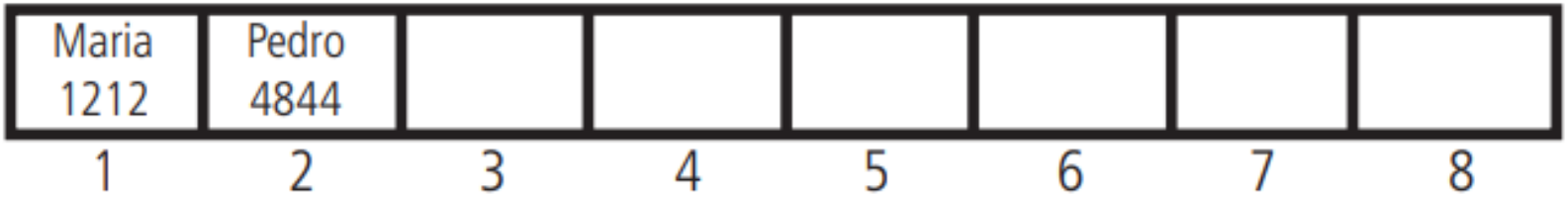
Devo retornar à página mais recentemente visitada.

Fila

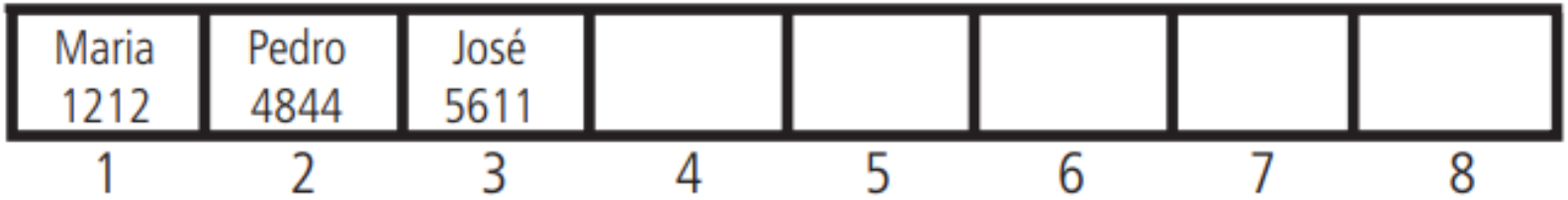
Inserção da aluna de matrícula 1212 e nome Maria



Inserção da aluna de matrícula 4844 e nome Pedro

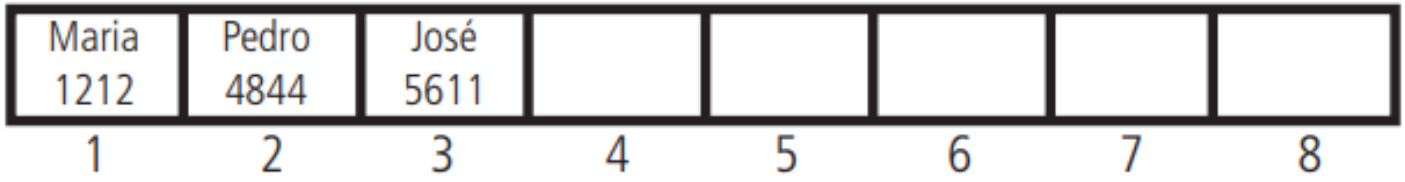


Inserção da aluna de matrícula 5611 e nome José

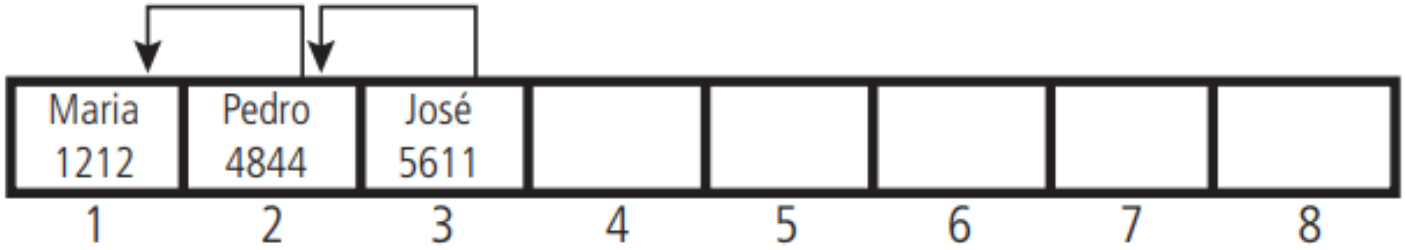


Fila

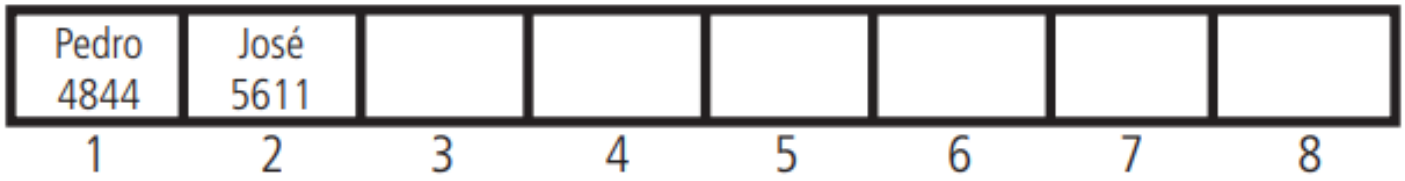
Fila com 3 elementos



Retirada do primeiro da fila



Fila depois da remoção



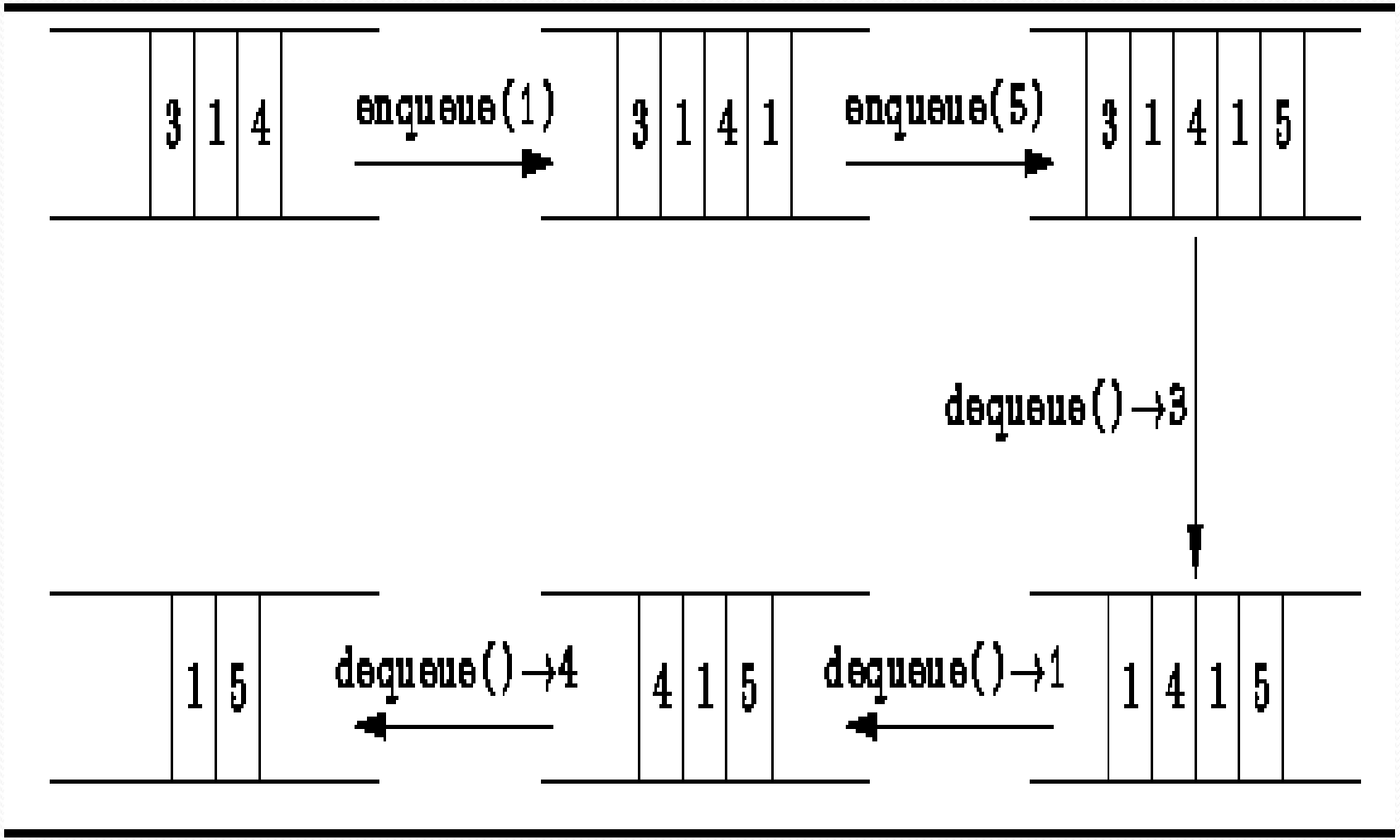
Fila

- Filas de impressão:
  - Impressoras tem uma fila, caso vários documentos sejam impressos, por um ou mais usuários, os primeiros documentos impressos serão de quem enviar primeiro;
- Filas de processos:
  - Vários programas podem estar sendo executados pelo sistema operacional. O mesmo tem uma fila que indica a ordem de qual será executado primeiro;
- Filas de tarefas:
  - Um programa pode ter um conjunto de dados para processar. Estes dados podem estar dispostos em uma fila, onde o que foi inserido primeiro, será atendido primeiro.

Fila

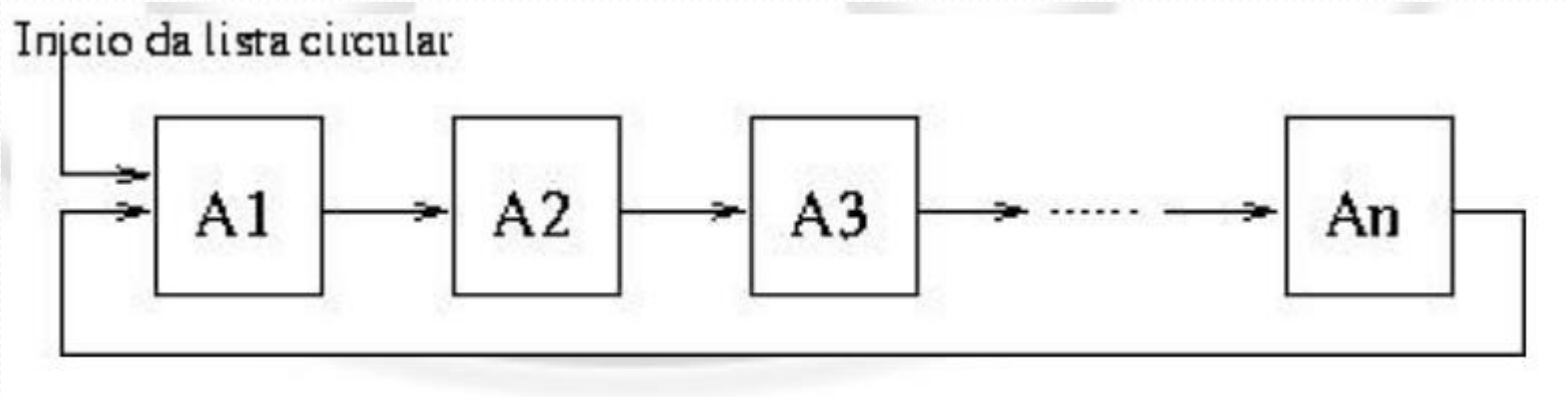
- Fila de Prioridades:
  - Cada item tem uma prioridade. Elementos mais prioritários podem ser atendidos antes, mesmo não estando no inicio da fila;
- Fila Circular:
  - Neste tipo de fila os elementos nem sempre são removidos ao serem atendidos, mas voltam ao fim da fila para serem atendidos novamente mais tarde.

Fila de prioridades



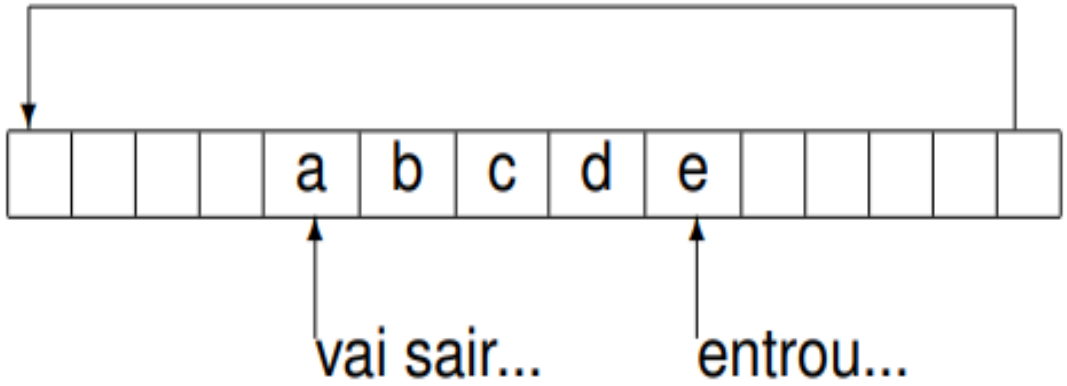
Listas circulares

- Outro tipo especial de lista é formado pelas listas circulares
- Nelas existe uma ligação entre entre o último e o primeiro elemento da lista, fechando portanto um ciclo
- Listas circulares são muito usadas na implementação de buffers para entrada de dados

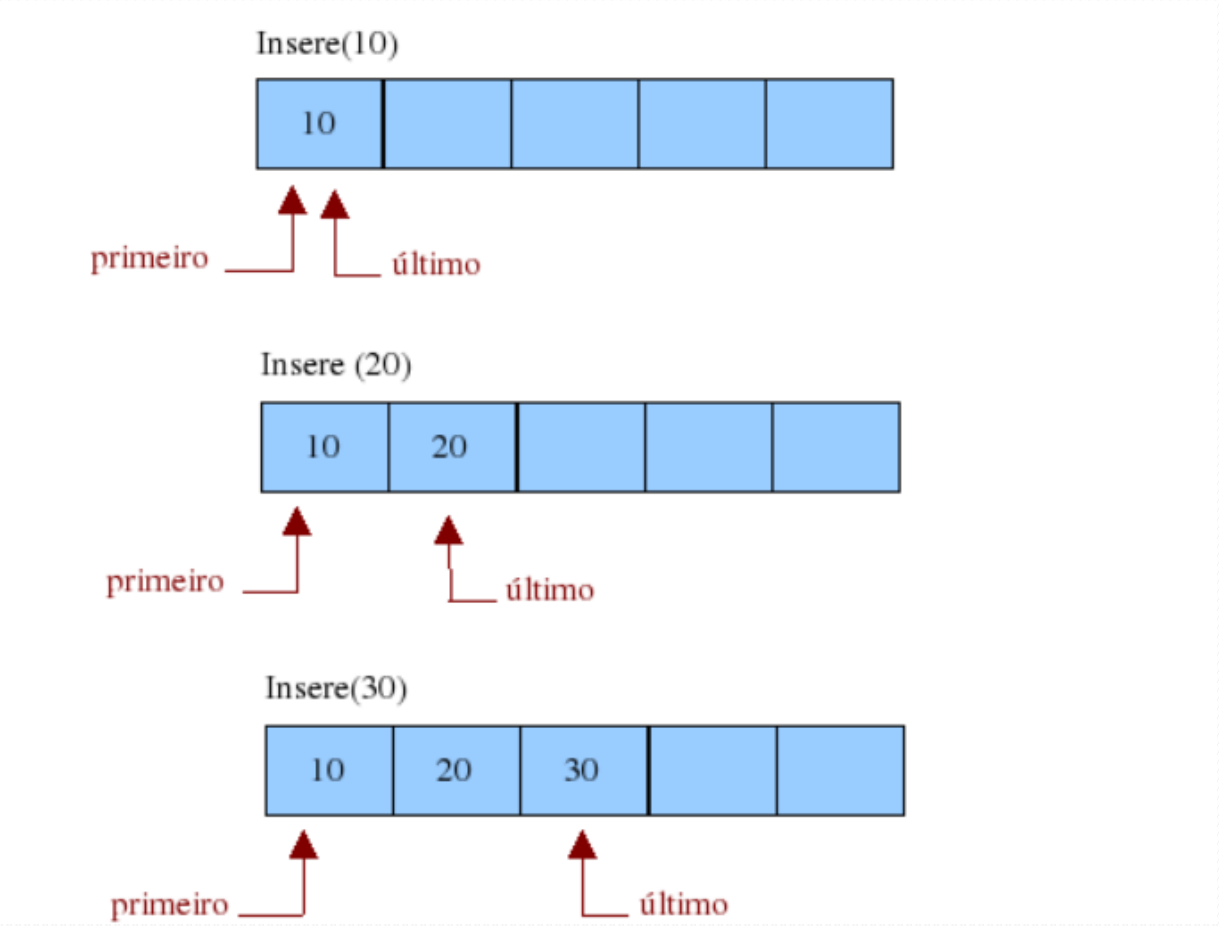


Fila circular

- A implementação mais comum de uma fila é por “arranjo circular”.

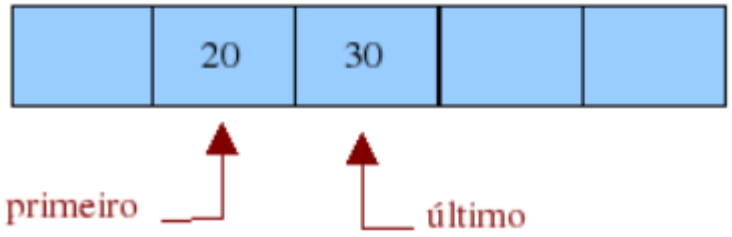


Fila circular

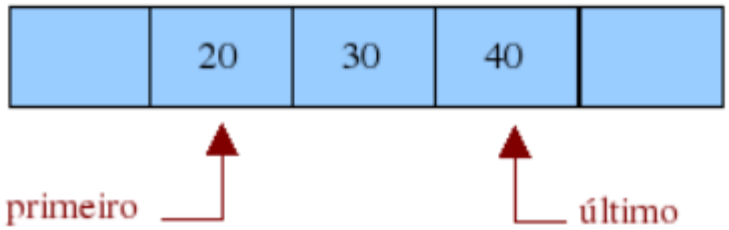


Fila circular

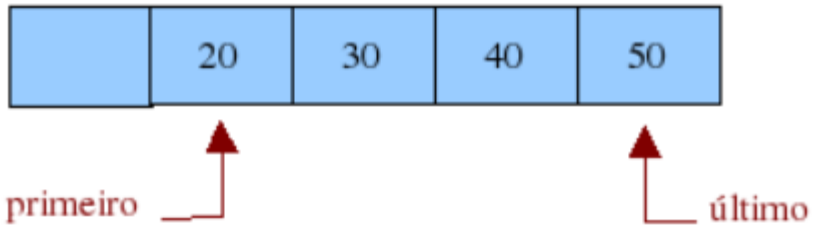
Remove()



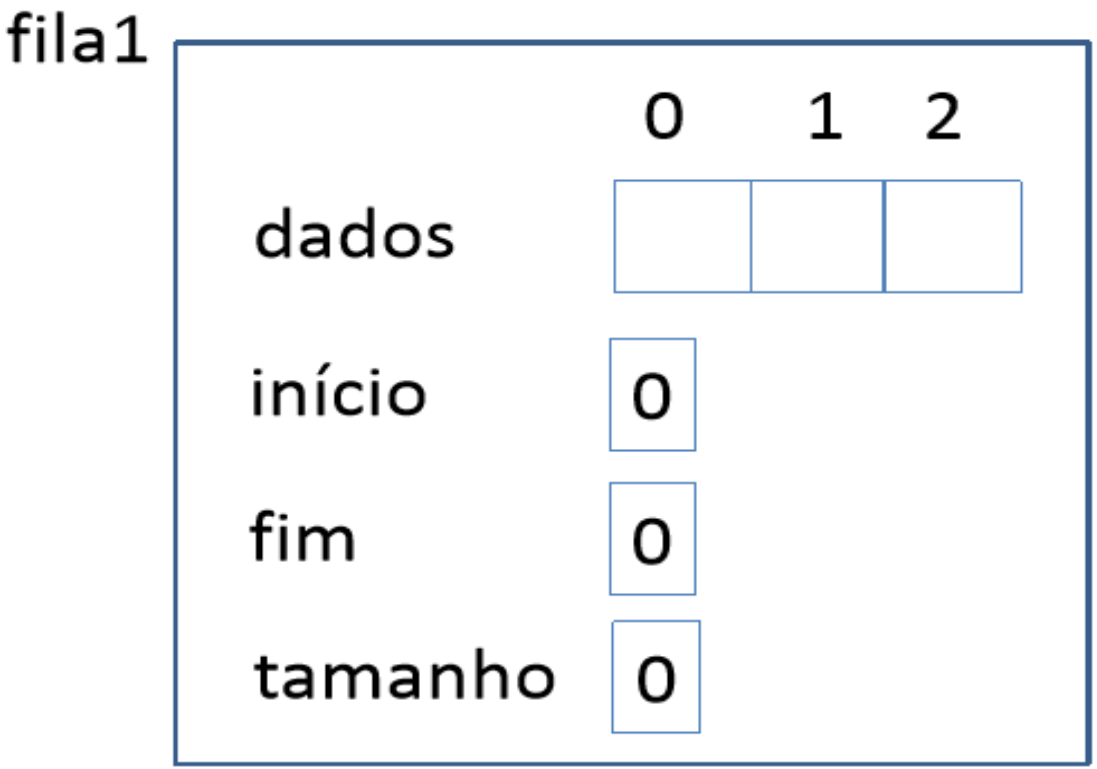
Insere (40)



Insere (50)



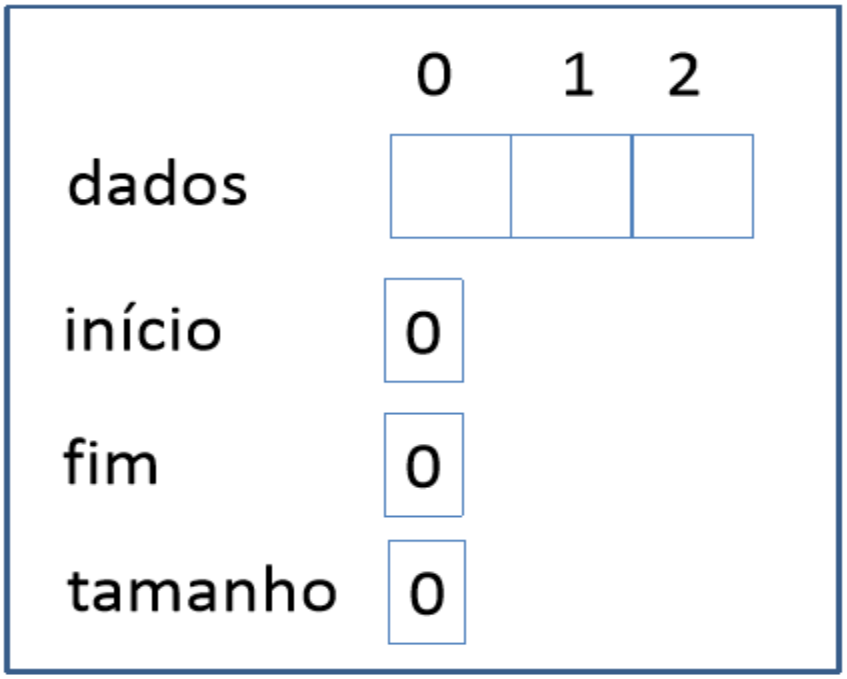
Fila circular



Fila circular -

adiciona (90)

fila1

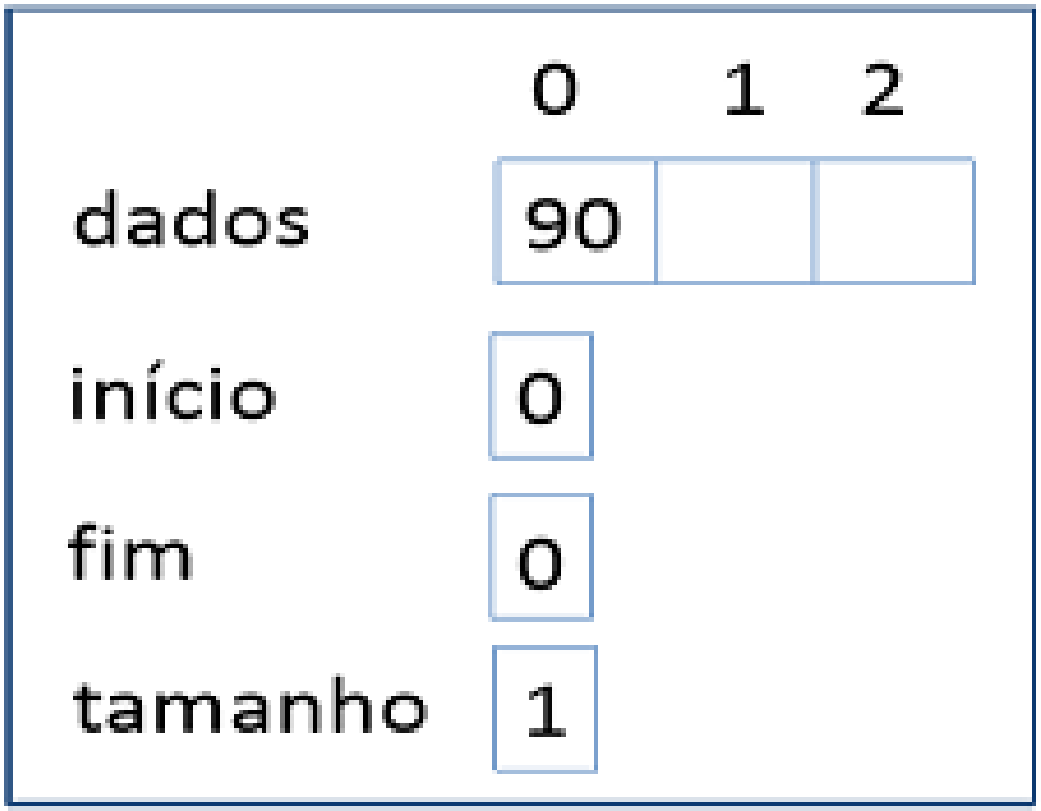


Estado atual da fila1. A fila está cheia? _____

Fila circular

Como a fila **não** está cheia:

fila1

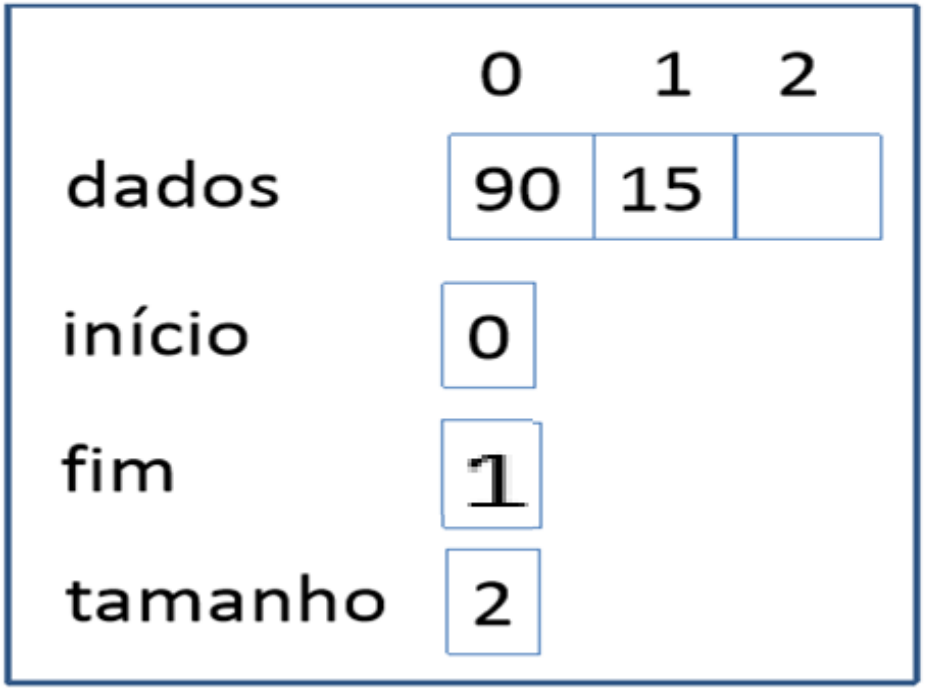


Fila circular

adiciona (15)

Como a fila **não** está cheia:

fila1

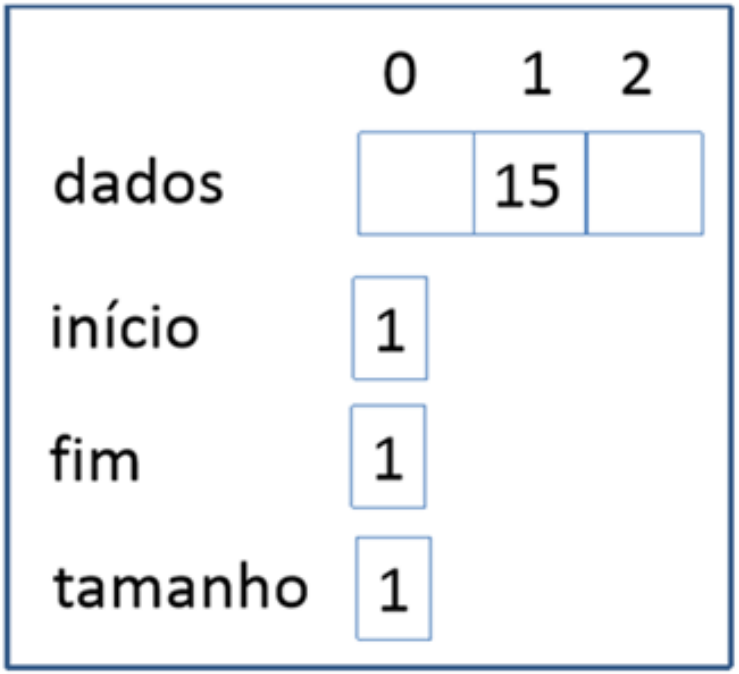


Fila circular

remove ()

Como a fila **não** está vazia:

fila1



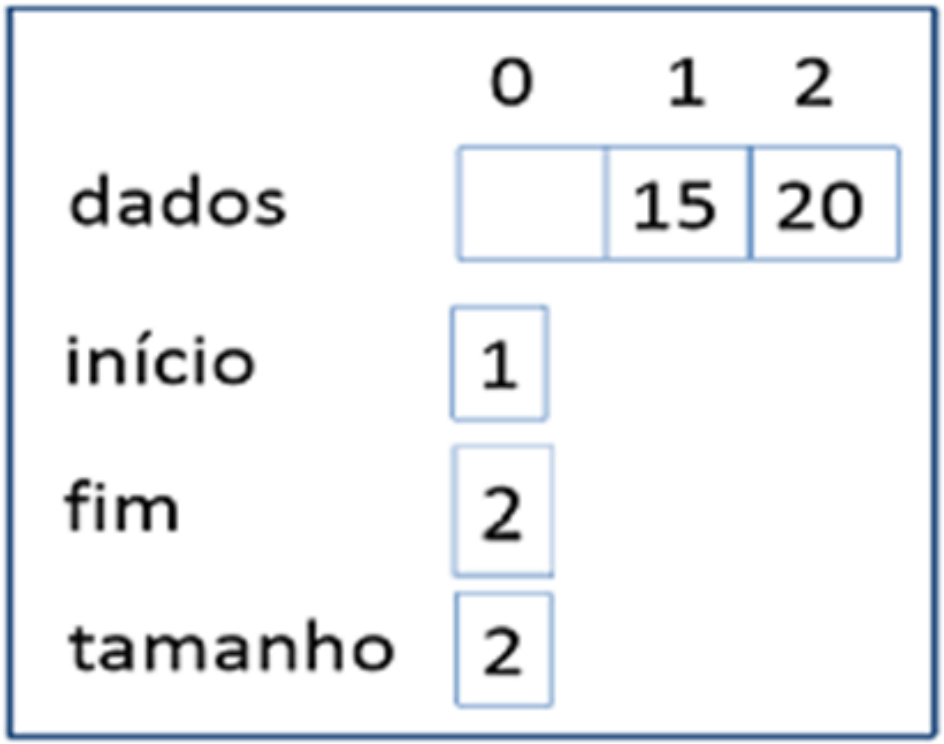
Retornar: **90**

Fila circular

adiciona (20)

Como a fila **não** está cheia:

fila1

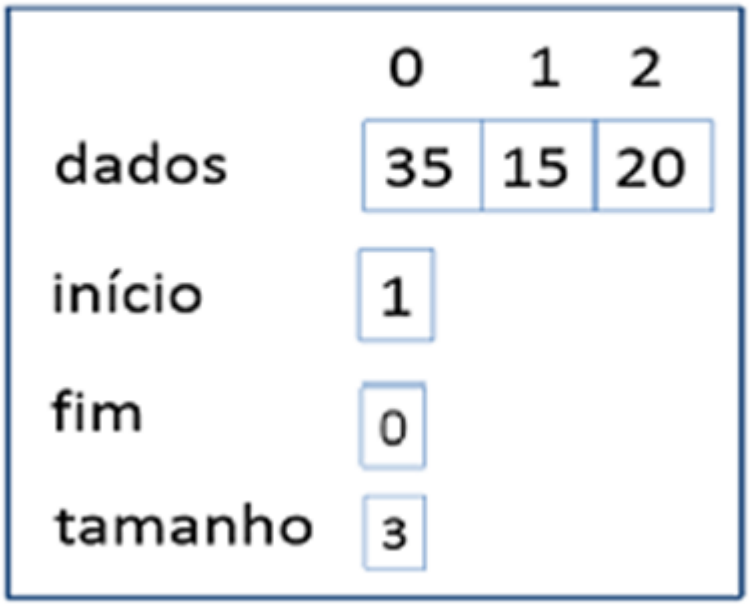


Fila circular

adiciona (35)

Como a fila **não** está cheia:

fila1

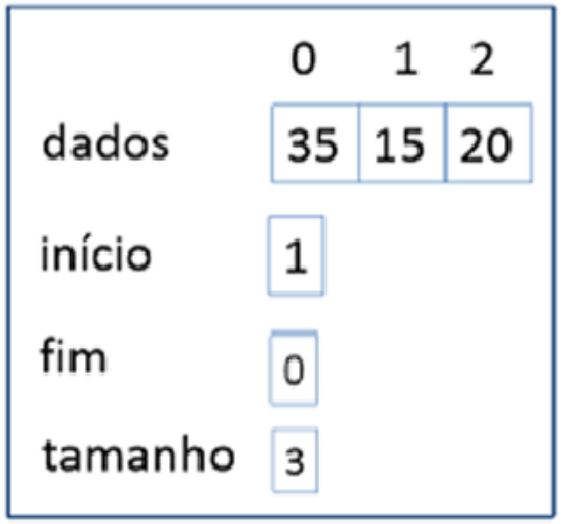


Fila circular

adiciona (51)

Como a fila **está cheia**:

fila1



Exibir:

ERRO! Fila Cheia!

Remoção na Fila de Prioridades

```
public Usuario remove(){
    Usuario r=null;
    if (tamanho>=1){
        r=dados[0];
        for (int i=0;i<tamanho-1;i++){
            dados[i]=dados[i+1];
        }
        tamanho--;
    }
    else{
        JOptionPane.showMessageDialog(null, "Fila vázia");
    }
    return r;
}
```

Pilha

- Uma pilha é uma estrutura de dados em que o acesso é restrito ao elemento mais recente na pilha.
- Uma pilha é uma estrutura de dados que pode ser acessada somente por uma de suas extremidades para armazenar e recuperar dados.
- Por essa razão, uma pilha é chamada de estrutura *LIFO* (*last in first out*).

Pilha

- São listas em que as operações de remoção e inserção ocorrem sempre em locais específicos
- A inserção é feita sempre no início da lista
- A remoção é feita sempre no início da lista
- **Em função disso uma fila assume a condição LIFO**

Pilha

- Dada uma pilha P = (a(1), a(2), ..., a(n)), dizemos que a(1) é o elemento da base da pilha; a(n) é o elemento topo da pilha; e a(i+1) está acima de a(i).
- Em uma pilha "ideal", operações básicas devem ocorrer em O(1), independentemente do tamanho N da pilha (ou seja, em tempo constante).

Pilha

- O conceito de pilha é usado em muitos softwares de sistemas incluindo compiladores e interpretadores.
- Como exemplo de sua utilização, *A maioria dos compiladores C usa pilha quando passa argumentos para funções).*
- As duas operações básicas – armazenar e recuperar – são implementadas por funções tradicionalmente chamadas de *push e pop*, respectivamente.
- A função *push()* coloca um item na pilha e a função *pop()* recupera um item da pilha.
- A região de memória a ser utilizada como pilha pode ser um vetor, ou uma área alocada dinamicamente.

Pilha

- Operações:
- Adicionar elemento (método push)
- Remover elemento (método pop)
- Verificar se a pilha está vazia
- Verificar se a pilha está cheia

Pilha

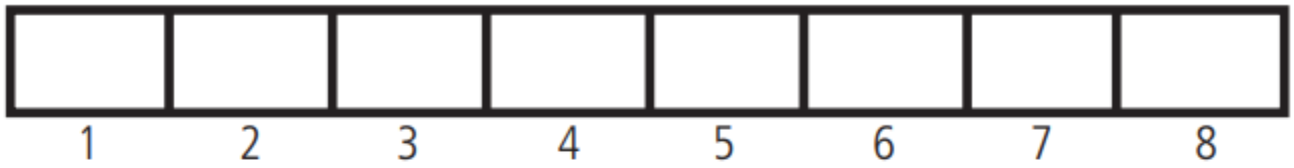
- Na implementação de pilha, em apenas uma das extremidades, chamada de topo, é realizada a manipulação dos elementos, em oposição a outra extremidade, chamada de base.
- Todas as operações em uma pilha podem ser imaginadas como as que ocorre numa pilha de livros, jornais, revistas, papéis e vários outros exemplos de aplicação.

Pilha

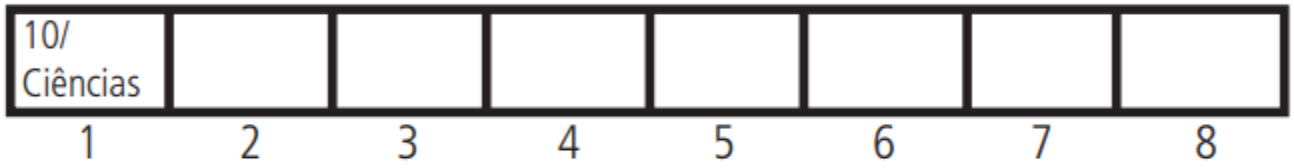
- Exemplos de aplicação:
- Calculadora para expressões matemáticas
- Conversão de número decimal para binário
- Retirada de mercadorias de um caminhão de entregas
- Mecanismo de fazer/desfazer do Word
- Mecanismo de navegação de páginas na Internet (avançar e retornar).

Pilha

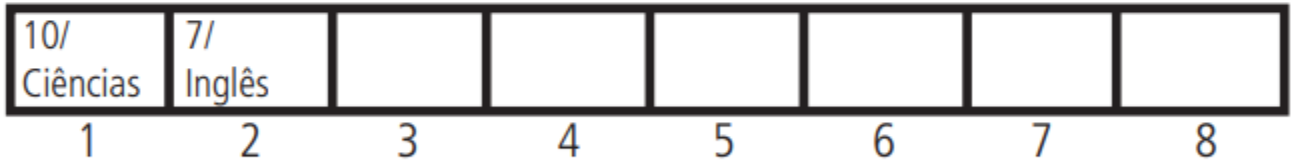
Vetor Vazio



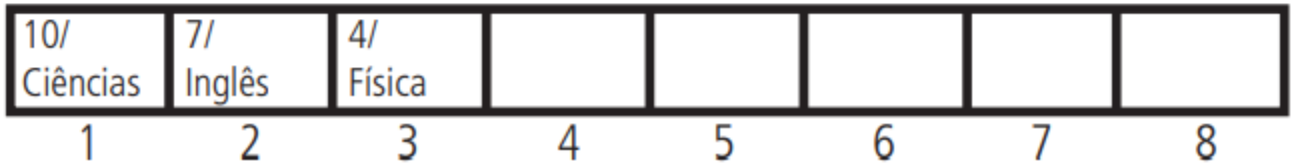
Inserção do Livro com código10 e título Ciências



Inserção do Livro com código7 e título Inglês

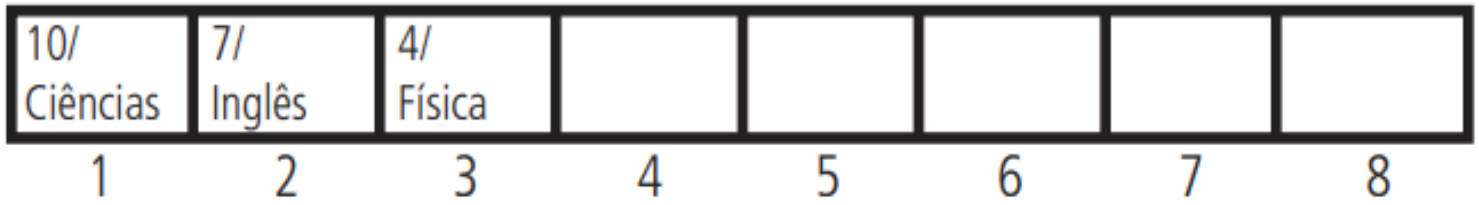


Inserção do Livro com código4 e título Física

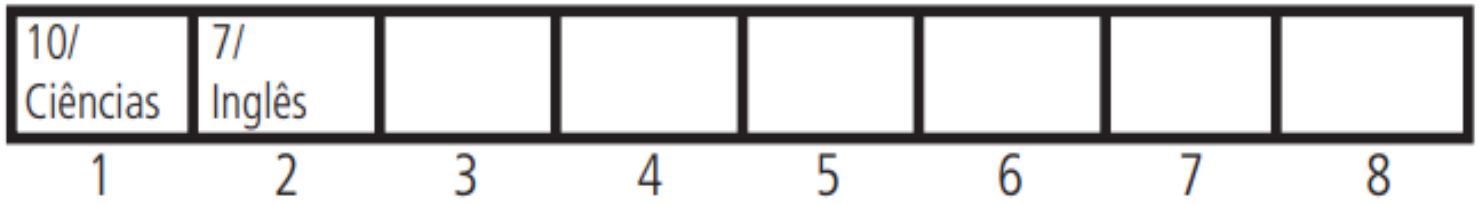


Pilha

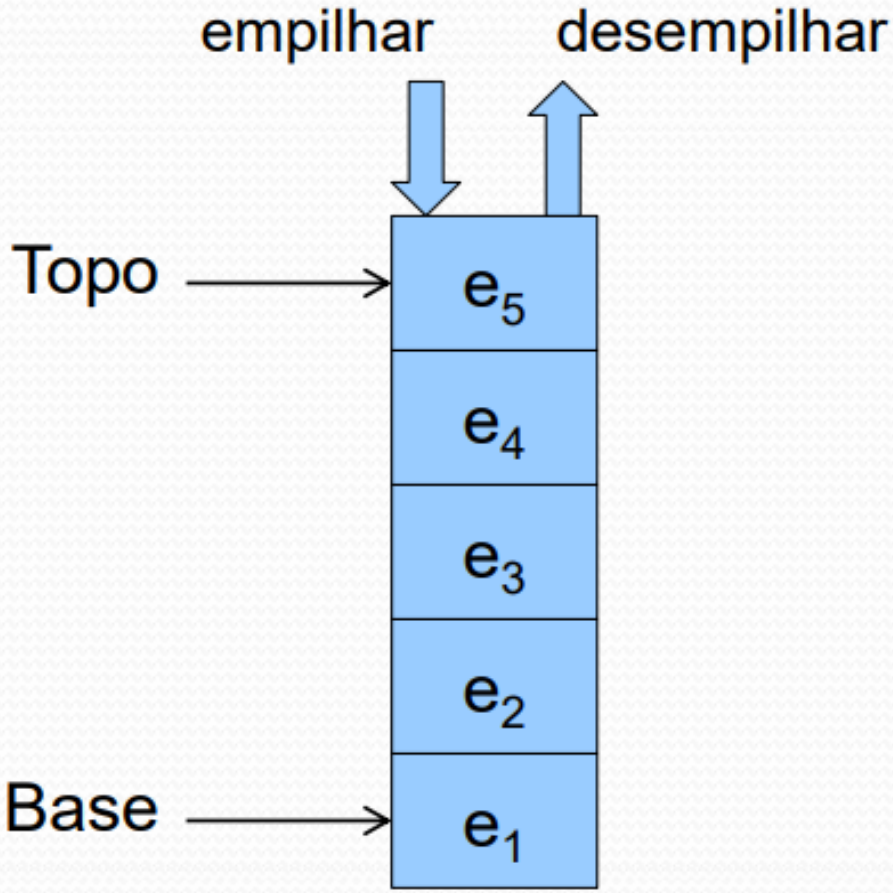
Pilha com três elementos



Pilha depois da remoção



Pilha



Algoritmo básico

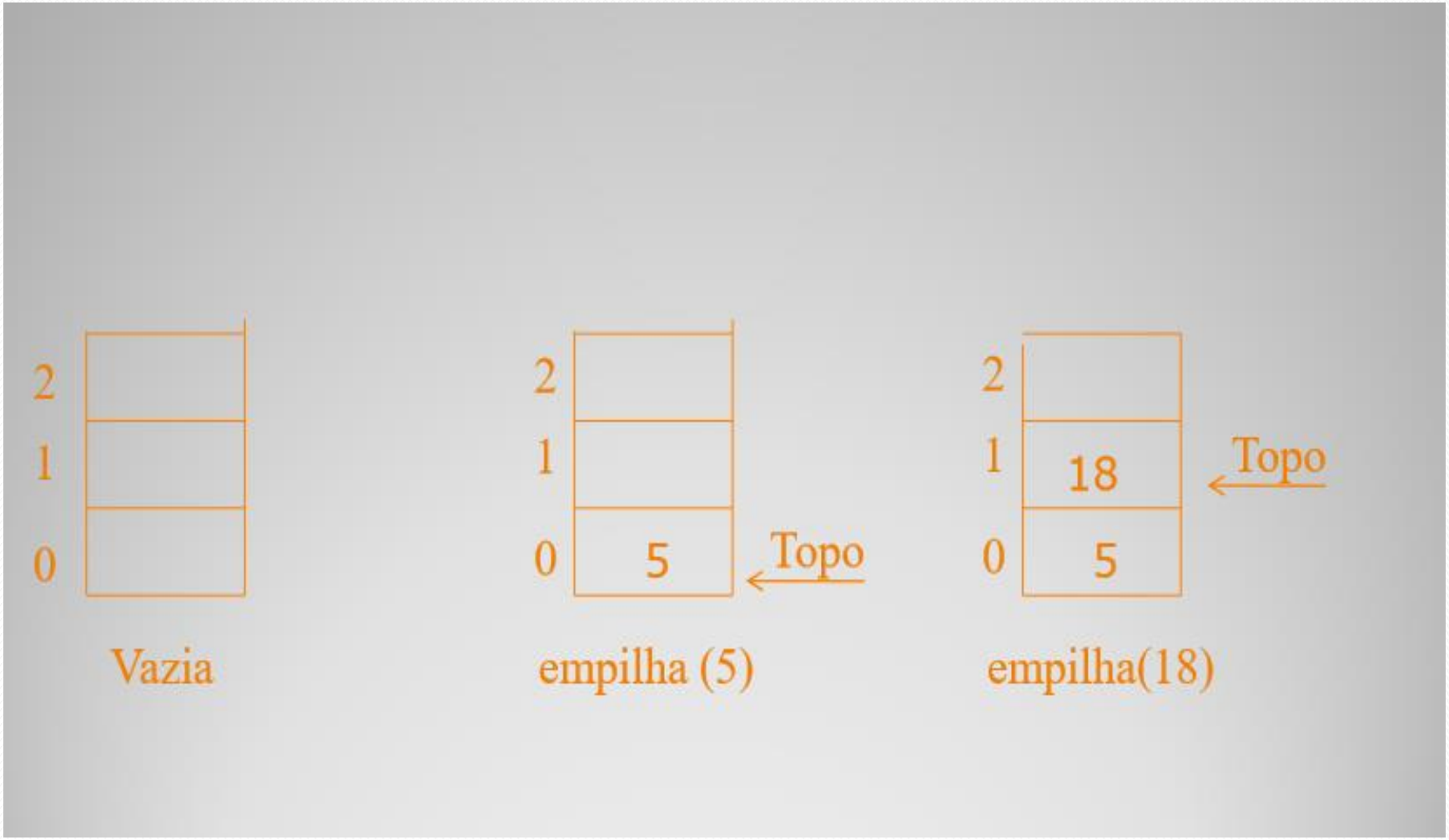
Algoritmo 8.1: EMPILHA(P, x)

```
1 se P.topo ≠ P.capacidade então
2     P.topo = P.topo + 1
3     P[P.topo] = x
```

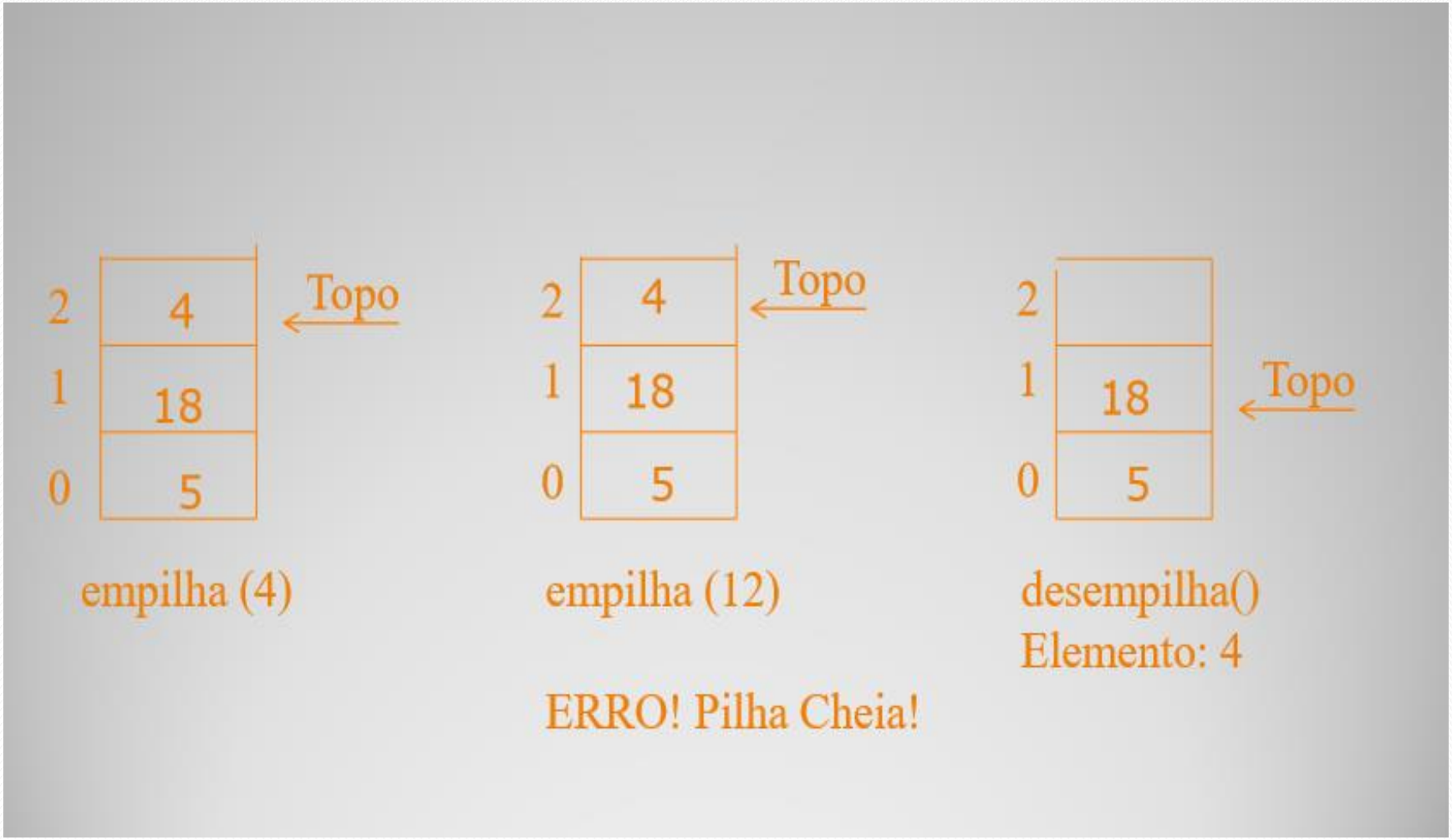
Algoritmo 8.2: DESEMPILHA(P)

```
1 se P.topo ≠ 0 então
2     x = P[P.topo]
3     P.topo = P.topo − 1
4     retorna x
5 senão
6     retorna null
```

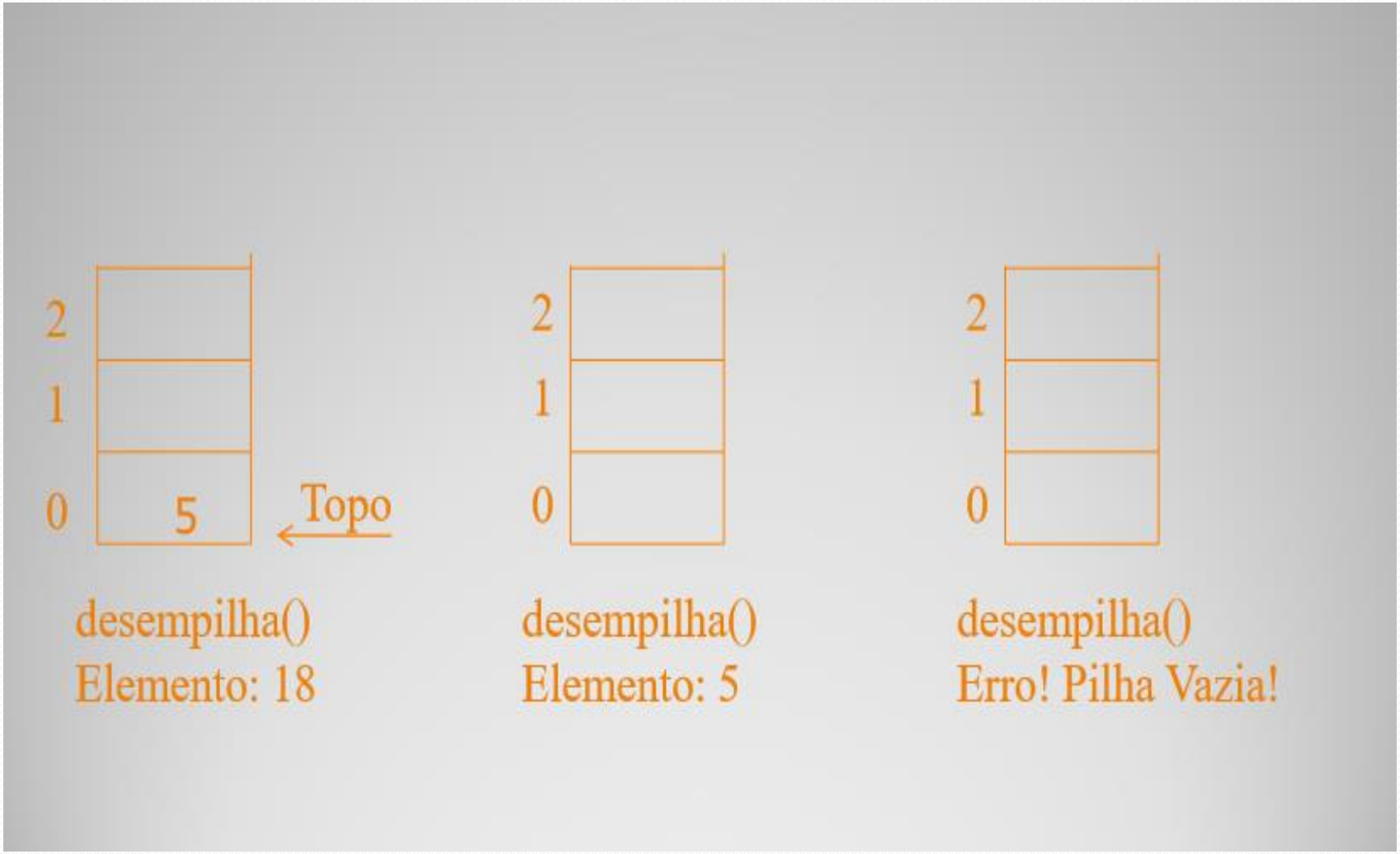
Pilha



Pilha


2
4
Topo
1
18
0
5
empilha (4)
2
4
Topo
1
18
0
5
empilha (12)
ERRO! Pilha Cheia!
2
1
18
Topo
0
5
desempilha()
Elemento: 4

Pilha


2
1
0
5
Topo
desempilha()
Elemento: 18
2
1
0
desempilha()
Elemento: 5
2
1
0
desempilha()
Erro! Pilha Vazia!

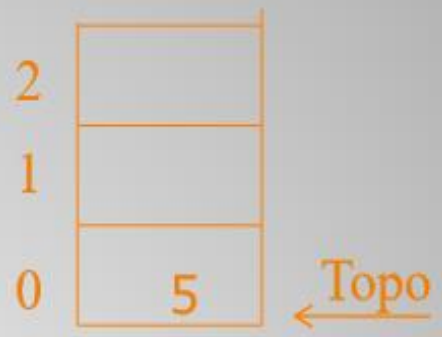
Pilha

Implementação com estrutura homogênea

Tipo dos Elementos: inteiro

Quais são os atributos?

- array
- topo
- capacidadeMax




```
public class Pilha{
        int  dados[];
        int topo;
        int capacidadeMax;
}
```

Pilha

Qual índice recebe o primeiro elemento?

0 (zero)

Como indicar que a pilha está vazia?

-1

Construtor:

```
public Pilha(int capacidade){
        dados = new int[capacidade];
        topo = -1;
        capacidadeMax=capacidade;
}
```

Pilha

Como obter o elemento do topo ?

```
public int obtemTopo(){
        return dados[topo];
}
```

Pilha

Como verificar se a pilha está vazia?

```
public  boolean vazia(){
        //se topo valer -1, então a pilha está vazia
        if (topo == -1)
                return true;
        return false;
}
```

Pilha

Como verificar se a pilha está vazia?

```
public  boolean cheia(){
        //se topo +1 for igual ao tamanho , então a pilha está cheia
        if (topo+1 == capacidadeMax)
                return true;
        return false;
}
```

Pilha

Como empilhar elemento?

```
public  void empilha(int e ){
        //verifica se a pilha está cheia, então mostra mensagem de erro
        if (cheia())
                System.out.println("ERRO! Pilha Cheia!");
        else
        {
                //se não estiver cheia, incrementa o topo
                //e adiciona elemento
                dados[++topo]=e;
        }
}
```

Pilha

Como desempilhar elemento?

```
public  int desempilha( ){
        //declara variável para guardar o elemento que será removido
        int elemento=-1;
        //verifica se a pilha está vazia, então mostra mensagem de erro
        if (vazia())
                System.out.println("ERRO! Pilha Vazia!");
        else
        {
                //se não estiver vazia, guarda elemento do topo
                //na variável local e decrementa o topo
                elemento = dados[topo--];
        }
        return elemento;
}
```

Pilha

- Aplicações
  - Recursividade
  - A solução de um problema depende da solução de instâncias menores do mesmo problema

Fatorial iterativo

```
int factorial (int n) {
  int result = 1;
  while(n > 1) {
    result *= n;
    n -= 1;
  }
  return result ;
}
```

Fatorial recursivo

```
int factorial (int n) {
  if (n≤1)
    return 1;
  else
    return n* factorial (n−1);
}
```

Comparativo

- Filas e pilhas têm regras bastante rigorosas para acessar dados.
- Pilhas e filas implementadas em vetores usam regiões contíguas de memória, listas não necessariamente.
- Além disso, a recuperação de um item da lista encadeada não causa a sua destruição. (*É preciso uma operação de exclusão específica para esta finalidade).*

Contatos

- Email: fabio.silva321@fatec.sp.gov.br
- Linkedin: https://br.linkedin.com/in/b41a5269
- Facebook: https://www.facebook.com/fabio.silva.56211