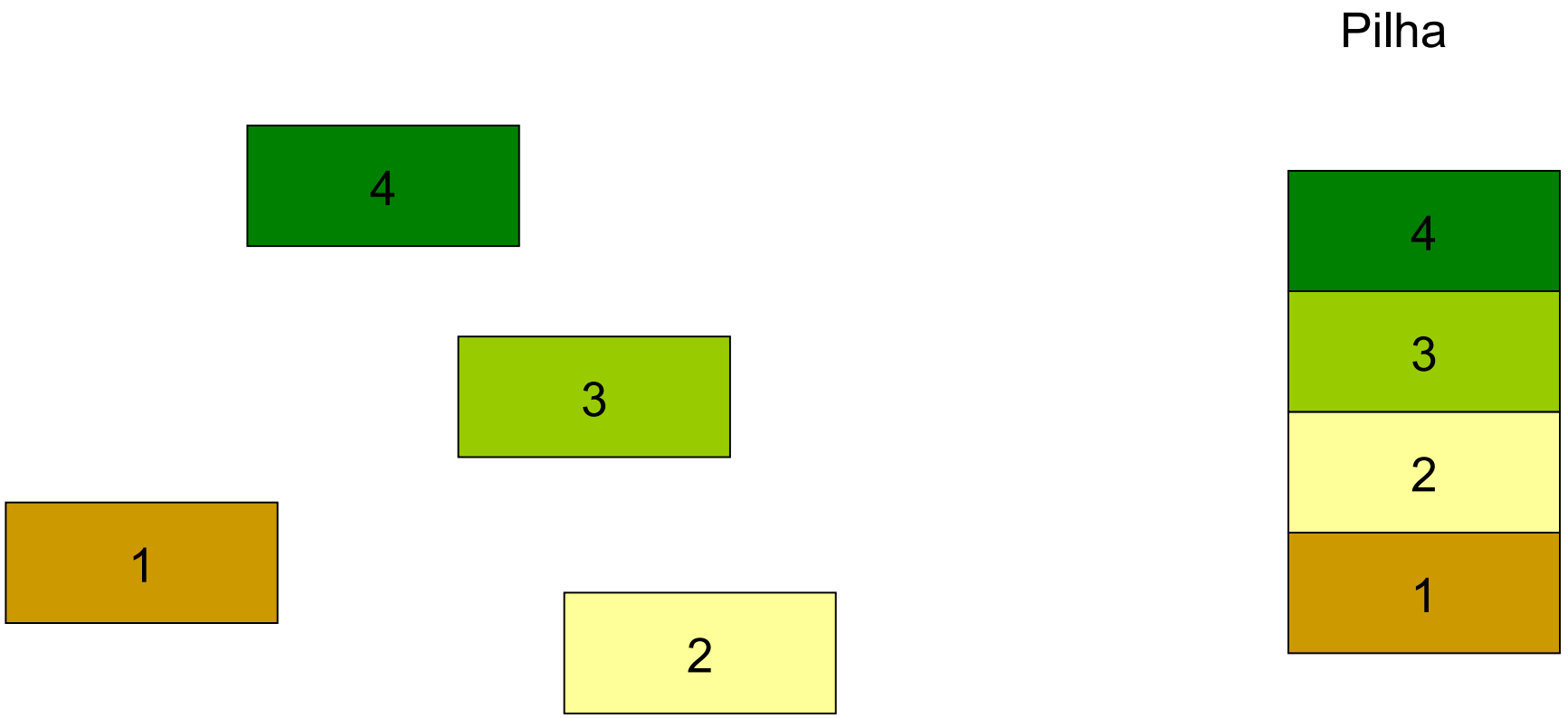
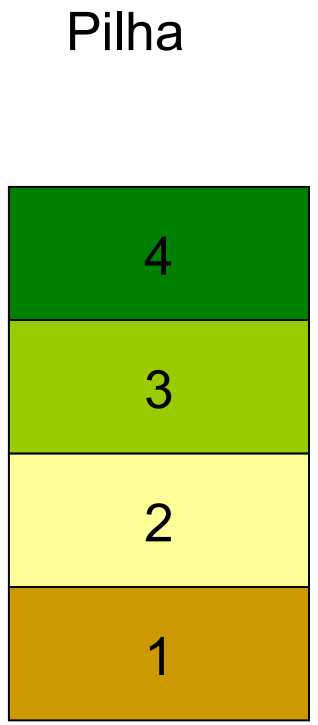
TAD Pilha

Luciano Nascimento Moreira

O que é uma pilha?



O que é uma pilha?



Pilha

- É uma lista linear em que todas as inserções, retiradas e, geralmente, todos os acessos são feitos em apenas um extremo da lista.
- Os itens são colocados um sobre o outro. O item inserido mais recentemente está no topo e o inserido menos recentemente no fundo.

Pilha

- O modelo intuitivo é o de um monte de pratos em uma prateleira, sendo conveniente retirar ou adicionar pratos na parte superior.
- Esta imagem está frequentemente associada com a teoria de autômato, na qual o topo de uma pilha é considerado como o receptáculo de uma cabeça de leitura/gravação que pode empilhar e desempilhar itens da pilha.

Propriedades das Pilhas

- Propriedade: O **último** elemento a ser inserido é o **primeiro** a ser retirado/ removido

(LIFO – Last in First Out)

- Analogia: pilha de pratos, livros, etc.

Aplicações de Pilhas

- É ideal para processamento de estruturas aninhadas de profundidade imprevisível.
- Uma pilha contém uma sequência de obrigações adiadas. A ordem de remoção garante que as estruturas mais internas serão processadas antes das mais externas.

Aplicações de Pilhas

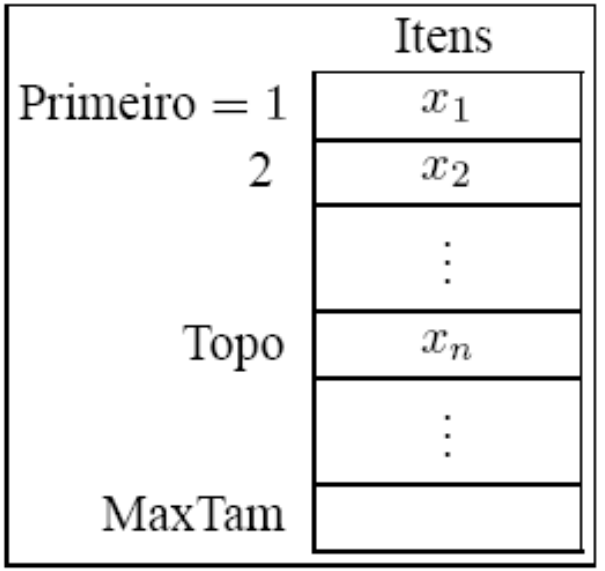
- Aplicações em estruturas aninhadas:
  - Quando é necessário caminhar em um conjunto de dados e guardar uma lista de coisas a fazer posteriormente.
  - O controle de sequências de chamadas de subprogramas.
  - A sintaxe de expressões aritméticas.
- As pilhas ocorrem em estruturas de natureza recursiva (como árvores). Elas são utilizadas para implementar a **recursividade**.

TAD Pilha

- Conjunto de operações:
  1. Cria uma pilha Vazia.
  2. Verifica se a lista está vazia. Retorna true se a pilha está vazia; caso contrário, retorna false.
  3. Empilhar o item x no topo da pilha.
  4. Desempilhar o item x no topo da pilha, retirando-o da pilha.
  5. Verificar o tamanho atual da pilha.
- Existem várias opções de estruturas de dados que podem ser usadas para representar pilhas.
- As duas representações mais utilizadas são as implementações por meio de arranjos e de apontadores

Implementação de Pilhas através de Arranjos

- Os itens da pilha são armazenados em posições contíguas de memória.
- Como as inserções e as retiradas ocorrem no topo da pilha, um campo chamado Topo é utilizado para controlar a posição do item no topo da pilha.



Estrutura de Dados de Pilha através de Arranjos

- Os itens são armazenados em um arranjo de tamanho suficiente para conter a pilha.
- O outro campo do mesmo registro contém uma referência para o item no topo da pilha.
- A constante maxTam define o tamanho máximo permitido para a pilha.

Estrutura de Dados de Pilha através de Arranjos

```
public class Pilha<T> {
    private T item[];
    private int topo;

    // Operações
    public Pilha(int maxTam) { // Cria uma Pilha vazia
        item = new T[maxTam];
        topo = 0;
    }

    public void empilha(T item) throws Exception {
        if (topo == item.length)
            throw new Exception("Erro: A pilha esta cheia");
        else item[topo++] = item;
    }
```

Estrutura de Dados de Pilha através de Arranjos

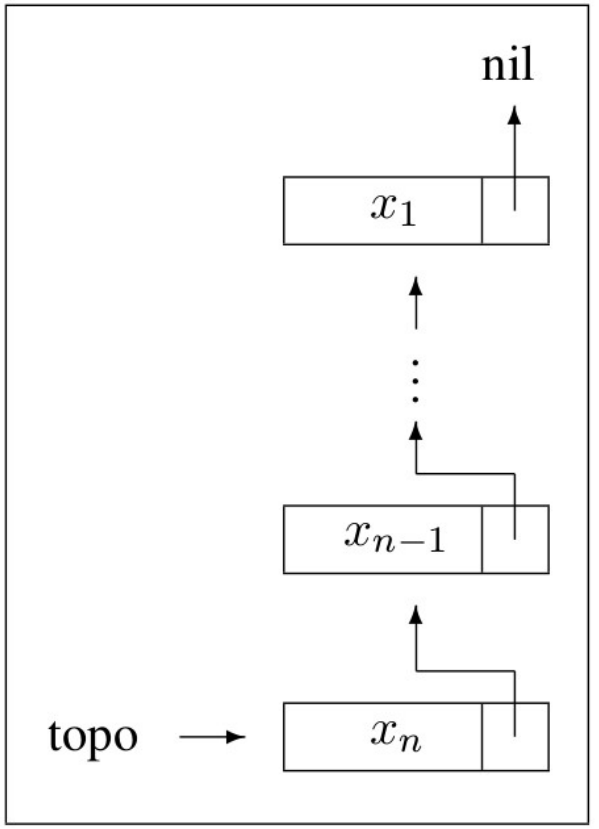
```
    public T desempilha() throws Exception {
        if (vazia())
            throw new Exception("Erro: A pilha esta vazia");
        return item[--topo];
    }

    public boolean vazia() {
        return (topo == 0);
    }

    public int tamanho() {
        return topo;
    }
}
```

Implementação de Pilhas por meio de Estruturas Auto-Referenciadas

- Ao contrário da implementação de listas lineares por meio e estruturas auto-referenciadas não há necessidade de manter uma célula cabeça é no topo da pilha.
- Para desempilhar o item x_n basta desligar a célula cabeça da lista e a célula que contém x_{n-1} passa a ser a célula cabeça.
- Para empilhar um novo item, basta fazer a operação contrária, criando uma nova célula para receber o novo item.



Implementação de Pilhas por meio de Estruturas Auto-Referenciadas

- O campo *tam* evita a contagem do número de itens no método *tamanho*.
- Cada célula de uma pilha contém um item da pilha e uma referência para outra célula.
- A classe *Pilha* contém uma referência para o topo da pilha.

Implementação de Pilhas por meio de Estruturas Auto-Referenciadas

```
public class Pilha<T> {
    private class Celula {
        T item;
        Celula prox;
    }
    private Celula topo;
    private int tam;

    // Operações
    public Pilha() { // Cria uma Pilha vazia
        topo = null;
        tam = 0;
    }
```

Implementação de Pilhas por meio de Estruturas Auto-Referenciadas

```
public void empilha(T x) {
    Celula aux = topo;
    topo = new Celula();
    topo.item = x;
    topo.prox = aux;
    tam++;
}
public T desempilha() throws Exception {
    if (vazia ())
        throw new Exception("Erro : A pilha esta vazia");
    Object item = topo.item;
    topo = topo.prox;
    tam--;
    return item ;
}
```

Implementação de Pilhas por meio de Estruturas Auto-Referenciadas

```
    public boolean vazia() {
        return (topo == null);
    }
    public int tamanho() {
        return tam;
    }
}
```

Exemplo de Uso Pilhas
Editor de Textos (ET)

- Vamos escrever um Editor de Texto (ET) que aceite os comandos:
  - Cancela caracter
  - Cancela linha
  - Imprime linha
- O ET deverá ler um caractere de cada vez do texto de entrada e produzir a impressão linha a linha, cada linha contendo no máximo 70 caracteres de impressão.
- O ET deverá utilizar o tipo abstrato de dados Pilha definido anteriormente, implementado por meio de arranjo.

Exemplo de Uso Pilhas Editor de Textos (ET)

- “#”: cancelar caractere anterior na linha sendo editada.
  - Ex.: CFM##EFETP# → CEFET
- “\”: cancela todos os caracteres anteriores na linha sendo editada.
- “*”: salta a linha.
- “!”: Imprime os caracteres que pertencem à linha sendo editada, iniciando uma nova linha de impressão a partir do caractere imediatamente seguinte ao caractere salta-linha. Ex: DCCTIM*CEFET*
  - DCCTIM
  - CEFET

Sugestão de Texto para Testar o ET

Este et# um teste para o ET, o extraterrestre em Java.*Acabamos de testar a capacidade de o ET saltar de linha, utilizando seus poderes extras (cuidado, pois agora vamos estourar a capacidade máxima da linha de impressão, que é de 70 caracteres.)*O k#cut#rso dh#e Estruturas de Dados et# h#um cuu#rsh#o #x# x? *!#?!#+.* Como et# bom n#nt#ao### r#ess#tt#ar mb#aa#triz#cull#ado nn#x#ele!\ Sera que este funciona\\\? O sinal? não### deve ficar! ~

ET - Implementação

- Este programa utiliza um tipo abstrato de dados sem conhecer detalhes de sua implementação.
- A implementação do TAD Pilha que utiliza arranjo pode ser substituída pela implementação que utiliza apontadores sem causar impacto no programa