Estrutura de Dados I
Pilhas

Matheus Gabriel

Agosto 2024

1 Continuando pilhas

1.1 Mais sobre a implementação do zero

1.1.1 Relembrando o código

```java
// Stack.java
public class Stack {
    private int[] data;
    private int count;

    // Construtor para inicializar a pilha com um tamanho específico
    public Stack(int size) {
        data = new int[size];
        count = 0;
    }

    // Adiciona um elemento ao topo da pilha
    public void push(int value) {
        if (count == data.length) {
            System.out.println("Erro: pilha cheia.");
            return;
        }
        data[count++] = value;
    }

    // Remove e retorna o elemento do topo da pilha
    public int pop() {
        if (isEmpty()) {
            System.out.println("Erro: pilha vazia.");
            throw new RuntimeException("Pilha vazia");
        }
        return data[--count];
    }
```

```java
// Retorna o elemento do topo da pilha sem removê-lo
public int peek() {
    if (isEmpty()) {
        System.out.println("Erro: pilha vazia.");
        throw new RuntimeException("Pilha vazia");
    }
    return data[count - 1];
}

// Verifica se a pilha está vazia
public boolean isEmpty() {
    return count == 0;
}

// Retorna o número de elementos na pilha
public int size() {
    return count;
}

// Remove todos os elementos da pilha
public void clear() {
    count = 0;
}

// Representa a pilha como uma string
@Override
public String toString() {
    StringBuilder sb = new StringBuilder("Pilha: [");
    for (int i = 0; i < count; i++) {
        sb.append(data[i]);
        if (i < count - 1) {
            sb.append(", ");
        }
    }
    sb.append("]");
    return sb.toString();
}

// Método principal para teste
public static void main(String[] args) {
    Stack stack = new Stack(5);

    // Adiciona elementos à pilha
    stack.push(1);
    stack.push(2);
```

```
        stack.push(3);

        // Exibe a pilha
        System.out.println(stack);

        // Exibe o topo da pilha
        System.out.println("Topo da pilha: " + stack.peek());

        // Remove e exibe elementos da pilha
        while (!stack.isEmpty()) {
            System.out.println("Removido: " + stack.pop());
        }

        // Exibe se a pilha está vazia
        System.out.println("A pilha está vazia? " + stack.isEmpty());

        // Exibe a pilha após limpar
        stack.clear();
        System.out.println(stack);
    }
}
```

1.1.2 StringBuilder

```
    // Representa a pilha como uma string
    @Override
    public String toString() {
        StringBuilder sb = new StringBuilder("Pilha: [");
        for (int i = 0; i < count; i++) {
            sb.append(data[i]);
            if (i < count - 1) {
                sb.append(", ");
            }
        }
        sb.append("]");
        return sb.toString();
    }
```

Para manter a eficiência sempre use `StringBuilder` ao invés de usar uma string temporária no seu método `toString()`.