Código 2 - Estruturas de dados genéricas personalizadas e lista encadeada

Resumo: O código apresentado exemplifica o uso de conceitos fundamentais de Java, como generics, encapsulamento, orientação a objetos e manipulação de listas encadeadas. Ele demonstra a criação de uma estrutura de dados genérica representada pela classe No, que armazena dados e referências para outros nós, formando uma lista encadeada. O código também inclui exemplos de como criar e percorrer essa lista, vinculando nós sequencialmente. As principais referências utilizadas para elaborar este material incluem o livro "Java: Como Programar" (Deitel) para os conceitos de orientação a objetos, encapsulamento, e generics, além da documentação oficial da linguagem Java para detalhes técnicos específicos.

Será detalhado abaixo cada parte das classes ***No*** e ***Principal***, explicando os conceitos usados nos códigos.

Classe “No”

```
class No<T>{
    private T dado;
    private No<T> nextNo;
```

Essa classe define um nó **genérico No<T>** para uma estrutura de dados. O **T** é um parâmetro de tipo genérico que permite que a classe funcione com qualquer tipo de dado. O atributo **dado** armazena o valor do nó, e **nextNo** é uma referência ao próximo nó na lista.

```
public No(T dado){
    this(dado, null);
}
```

Esse construtor cria um novo nó inicializando o **dado** com o valor fornecido e define **nextNo** como **null**, indicando que o nó não está vinculado a outro nó.

```
public No(T dado, No<T> no){
    this.dado = dado;
    this.nextNo = no;
}
```

Esse construtor permite criar um nó e ao mesmo tempo especificar a referência para o próximo nó (**nextNo**). Se **no** for **null**, o nó será o último da lista.

```
public void setDado(T dado){
    this.dado = dado;
}

public T getDado(){
    return this.dado;
}
```

Esses são os métodos de acesso (**getters** e **setters**) para o atributo **dado**. Eles permitem ler e modificar o valor armazenado no nó.

```
public void setNextNo(No<T> nextNo){
    this.nextNo = nextNo;
}

public No<T> getNextNode(){
    return this.nextNo;
}
```

Esses métodos são usados para definir e acessar o próximo nó na lista. **setNextNo** permite vincular o nó atual a outro nó, e **getNextNode** retorna a referência ao próximo nó.

```
@Override
public String toString(){
    return "Dado{ " + getDado() + " }";
}
```

O método **toString** foi sobrescrito para fornecer uma representação em string do objeto **No**. Ele retorna uma **string** que exibe o valor armazenado em **dado**.

Classe "Principal"

```
import java.util.Scanner;
```

Essa linha de código importa a classe **Scanner** da biblioteca Java, que é usada para ler a entrada do usuário a partir do console.

```
public class Principal{
    public static void main(String[] args){
```

Essa é a classe **Principal** onde o método **main** está localizado. O método **main** é o ponto de entrada do programa.

```
Scanner input = new Scanner(System.in);
```

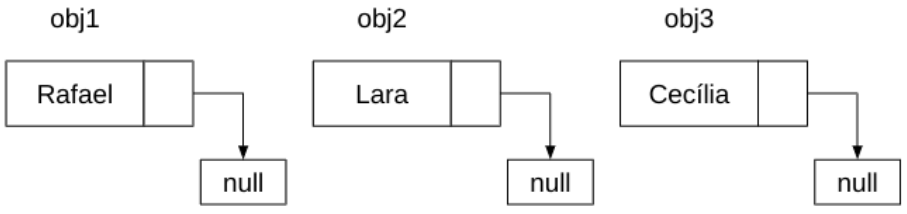
Cria uma instância de **Scanner** para ler a entrada do usuário.

```
No<String> obj1 = new No<String>("Rafael");
System.out.println(obj1.toString());

No<String> obj2 = new No<String>("Lara");
System.out.println(obj2.toString());

No<String> obj3 = new No<String>("Cecília");
System.out.println(obj3.toString());
```

Aqui, três objetos do tipo **No<String>** são criados, cada um armazenando um nome (Rafael, Lara, e Cecília). Os valores desses nós são impressos no console utilizando o método **toString**.

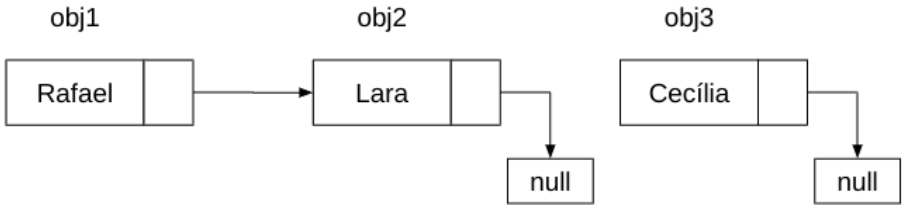


```
System.out.println("Imprimindo dadados da lista.");
System.out.println(obj1.toString());
```

A segunda linha imprime o valor armazenado no primeiro nó (**obj1**), confirmando o dado armazenado.

```
obj1.setNextNo(obj2);
```

Cria a ligação entre o nó **obj1** e o **obj2**.

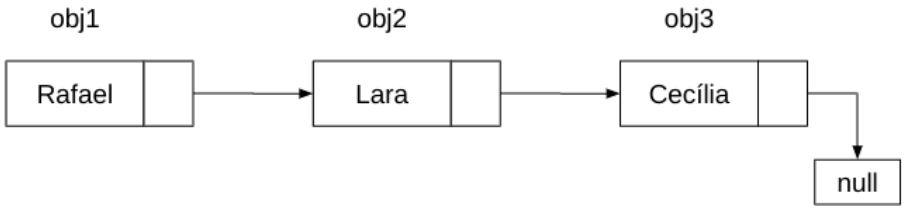



```
System.out.println(obj1.getNextNode().toString());
```

Essa linha imprime o valor armazenado no segundo nó (**obj2**), confirmando o vínculo com o nó **obj1**.

```
obj2.setNextNo(obj3);
```

Cria a ligação entre o nó **obj2** e o **obj3**.




```
System.out.println(obj2.getNextNode().toString());
```

Essa linha imprime o valor armazenado no terceiro nó (**obj3**), confirmando o vínculo com o nó **obj2**.

Essas linhas criam a ligação entre os nós. **obj1** é vinculado a **obj2** e **obj2** é vinculado a **obj3**. O programa então imprime os valores dos nós vinculados usando **getNextNode**.

```
System.out.println("Imprimindo dadados da lista com o while.");
No<String> aux = obj1;
while(aux != null){
    System.out.println(aux.toString());
    aux = aux.getNextNode();
}
```

Esse trecho de código percorre a lista encadeada a partir de **obj1** até o final (onde **nextNo** é **null**). Em cada iteração, ele imprime o valor do nó atual e avança para o próximo nó usando **getNextNode**. O loop **while** continua até que **aux** seja **null**, indicando o final da lista.