Nome: João Pedro D'Agostin

Atividade 2

Classe NO

```
public class No<E> {

        private E elemento;
        private No<E> next;

            public No(E elemento){
                this.elemento = elemento;
                this.next = null;
            }
            public No(E elemento, No<E> next){
                this.elemento = elemento;
                this.next = next;
            }



            public E getElemento() {
                return elemento;
            }
            public void setElemento(E elemento) {
                this.elemento = elemento;
            }



            public No<E> getNext() {
                return next;
            }
            public void setNext(No<E> next) {
                this.next = next;
            }



        @Override
        public java.lang.String toString() {
            StringBuilder stringBuilder = new StringBuilder();
            stringBuilder.append("[elemento =
").append(elemento).append(", proximo = ").append(next).append("]");
            //return "No [elemento=" + elemento + ", next=" + next + "]";
            return stringBuilder.toString();

        }
```

```
    }
```

Classe ListaEncadeada

```
public class ListaEncadeada<E>{

    private No<E> head;//criando o atributo cabeça do tipo classe NO
    private No<E> cauda;//criando o atributo cauda do tipo classe NO
    private int size = 0;//criando o atributo size

        public void pushHead(E elemento){//método para inserir pela
cabeça

            No<E> no = new No<E>(elemento);//instanciando um novo NO

            no.setElemento(elemento);//setando o elemento
            no.setNext(head);//apontando a nova cabeça
            head = no;//cabeça = ao novo objeto instanciado, assim como
cada objeto novo será
            size++;//incremente o tamanho

        }

        public void pushCauda(E elemento){

            No<E> no = new No<E>(elemento);// instanciando um novo NO

            if(size == 0){//se o tamanho da lista for 0
                this.head = no;//a cabeça é igual ao novo NO criado
            }else{//se não

                this.cauda.setNext(no);//aponte pra cauda

            }
            this.cauda = no;//cauda = ao novo objeto instaciado
            size++;//incremente o tamanho

        }

        public void popHead(){//metodo remover pela cabeça

            if(head != null){//se a cabeça for nula

                No no = this.head;//Faça com que o nó seja igual a cabeça
```

```
                this.head = this.head.getNext();//faça com que a cabeça
seja igual ao proximo
                no = null;//faça com que esse no seja igual a nulo
                this.size--;//desincremente

            }else{

                System.out.println("\nNão é possível remover um item ! --
>Lista Vazia !");

            }

        }

        //public void popCauda(){

        //}

        public void popAll(){
            do{

                this.popHead();//faça o método popHead

            }while(this.head != null);{//enquanto a cabeça for diferente
de null

                System.out.println("A cabeça está nula!");

            }
        }

        public int getSize(){

            return this.size;

        }

        public boolean isEmpty(){

            return(size == 0);

        }

        //public boolean isFull(){

        //}

        @Override
        public java.lang.String toString() {
```

```
            StringBuffer stringBuffer = new StringBuffer();
            stringBuffer.append("\n[ Node -->  ");

            No firtElelement = head;
            while(firtElelement != null){
                stringBuffer.append("[ ][elemento = " +
firtElelement.getElemento()+ "]]" + "--proximo-->");
                firtElelement = firtElelement.getNext();
            }
            stringBuffer.append("  ]\n");
            return stringBuffer.toString();
            //return "ListaEncadeada [head=" + head + ", cauda=" + cauda
+ ", size=" + size + "]";
        }

}
```

ListaEncadeadaTeste

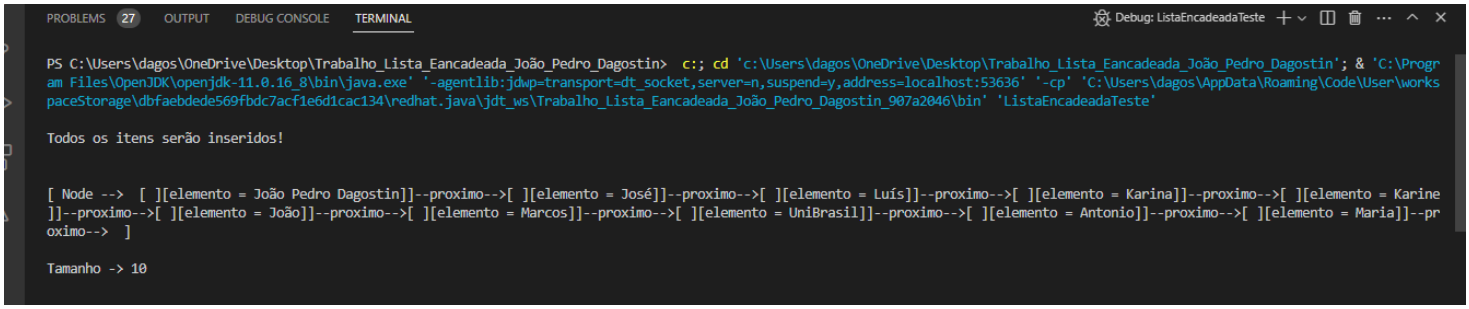
```
public class ListaEncadeadaTeste {

    public static void main(String[] args) {
        ListaEncadeadaTeste listaEncadeadaTeste = new
ListaEncadeadaTeste();
        listaEncadeadaTeste.execute();
    }

    public void adicionarHead(ListaEncadeada listaEncadeada){

        System.out.println("\nTodos os itens serão inseridos! \n");

        listaEncadeada.pushHead("Maria");
        listaEncadeada.pushHead("Antonio");
        listaEncadeada.pushHead("UniBrasil");
        listaEncadeada.pushHead("Marcos");
        listaEncadeada.pushHead("João");
        listaEncadeada.pushHead("Karine");
        listaEncadeada.pushHead("Karina");
        listaEncadeada.pushHead("Luís");
        listaEncadeada.pushHead("José");
        listaEncadeada.pushHead("João Pedro Dagostin");

        System.out.println(listaEncadeada);

        System.out.println("Tamanho -> " + listaEncadeada.getSize());

    }
```

```
    public void adicionarCauda(ListaEncadeada listaEncadeada){

        listaEncadeada.pushCauda("Maria");
        listaEncadeada.pushCauda("Antonio");
        listaEncadeada.pushCauda("UniBrasil");
        listaEncadeada.pushCauda("Marcos");
        listaEncadeada.pushCauda("João");
        listaEncadeada.pushCauda("Karine");
        listaEncadeada.pushCauda("Karina");
        listaEncadeada.pushCauda("Luís");
        listaEncadeada.pushCauda("José");
        listaEncadeada.pushCauda("João Pedro Dagostin");

        System.out.println(listaEncadeada);

        System.out.println("Tamanho -> " + listaEncadeada.getSize());

    }

    public void removerUmElemento(ListaEncadeada listaEncadeada){

        System.out.println("\n\nUm item da lista será removido!");
        listaEncadeada.popHead();
        System.out.println("\nTamanho -> " + listaEncadeada.getSize());
        System.out.println(listaEncadeada);

    }

    public void removerTodosOsElementos(ListaEncadeada listaEncadeada){

        System.out.println("\n\nTodos os itens da lista serão
removidos!\n");
        listaEncadeada.popAll();
        System.out.println("\nTamanho -> " + listaEncadeada.getSize());
        System.out.println(listaEncadeada);

    }

    public void execute(){
        ListaEncadeada<String> listaEncadeada = new ListaEncadeada<>();
        try {

            adicionarHead(listaEncadeada);
            //adicionarCauda(listaEncadeada);
            //removerTodosOsElementos(listaEncadeada);
            removerUmElemento(listaEncadeada);
            removerUmElemento(listaEncadeada);
            removerUmElemento(listaEncadeada);
```

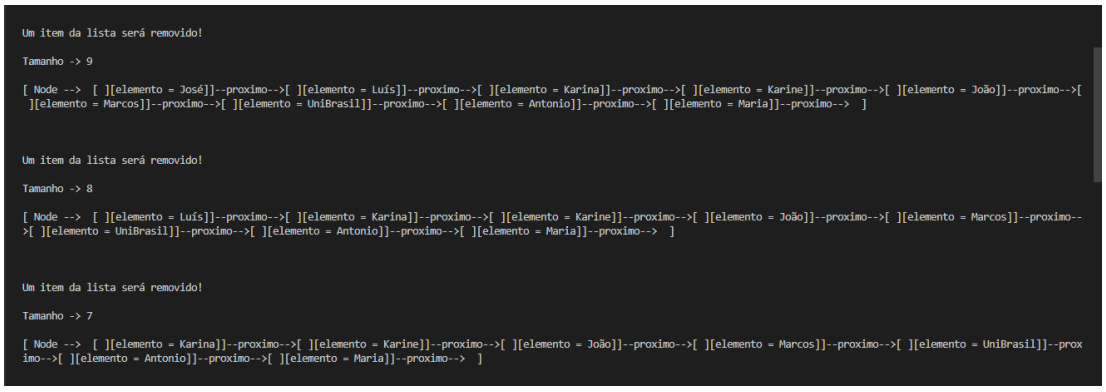
```
            removerUmElemento(listaEncadeada);
            removerUmElemento(listaEncadeada);
            removerUmElemento(listaEncadeada);
            removerUmElemento(listaEncadeada);
            removerUmElemento(listaEncadeada);
            removerUmElemento(listaEncadeada);
            removerUmElemento(listaEncadeada);
            removerUmElemento(listaEncadeada);

        } catch (Exception e) {
            System.out.println("Ocorreu um erro! Método Execute");
        }
    }
}
```

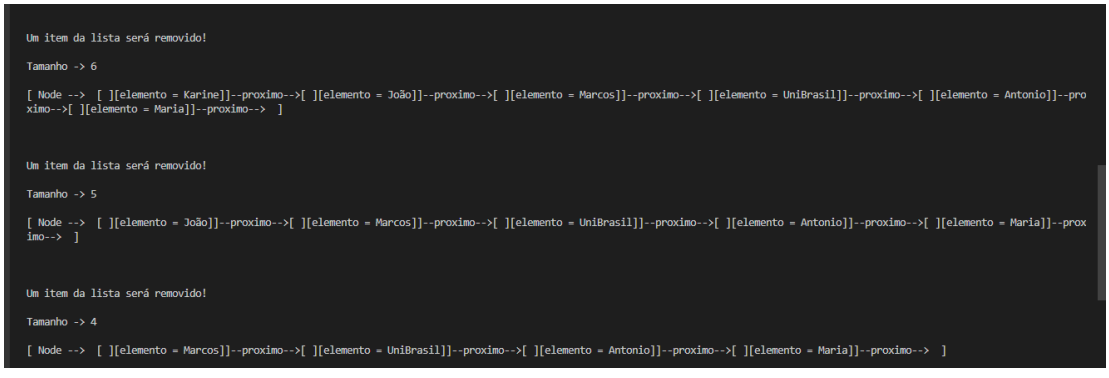
Prints

Nesta imagem podemos ver todo o Nó da lista encadeada depois da inserção, juntamente com o tamanho da lista.



Nestas próximas imagens podemos ver cada item sendo excluído e mostrando o tamanho a cada exclusão até chegar no erro de validação onde consta que a cabeça está vazia.





```
Um item da lista será removido!
Tamanho -> 3
[ Node -->  [ ][elemento = UniBrasil]]--proximo-->[ ][elemento = Antonio]]--proximo-->[ ][elemento = Maria]]--proximo-->  ]

Um item da lista será removido!
Tamanho -> 2
[ Node -->  [ ][elemento = Antonio]]--proximo-->[ ][elemento = Maria]]--proximo-->  ]

Um item da lista será removido!
Tamanho -> 1
[ Node -->  [ ][elemento = Maria]]--proximo-->  ]

Um item da lista será removido!
Tamanho -> 0
[ Node -->    ]
```

```
Um item da lista será removido!
Não é possível remover um item ! -->Lista Vazia !
Tamanho -> 0
[ Node -->    ]
PS C:\Users\dagos\OneDrive\Desktop\Trabalho_Lista_Eancadeada_João_Pedro_Dagostin>
```