Aufgabe 6.1

# Stapel (31 Punkte)

Mit den beiden Klassen `Stapel` und `StapelElement`, die in den Dateien Stapel.java und StapelElement.java enthalten sind, soll ein Stapel implementiert werden, der Zeichenketten speichern kann. Dabei soll die Klasse Stapel neben dem Standardkonstuktor die folgenden `public`-Methoden besitzen:

- `void fuegeElementHinzu(String s)` soll `s` dem Stapel hinzufügen
- `String entferneOberstesElement()` entfernt den zuletzt hinzugefügten `String` und liefert diesen zurück
- `String liefereOberstesElement()` liefert den zuletzt hinzugefügten `String` zurück
- `long liefereGroesse()` liefert die Anzahl der gespeicherten Elemente zurück
- `boolean istLeer()` liefert `true` zurück, wenn der Stapel keine Elemente enthält, ansonsten `false`

Sollte der Stapel keine Elemente beinhalten, sollen `entferneOberstesElement` und `liefereOberstesElement` `null` zurückliefern.
Der Stapel soll intern Objekte der Klasse `StapelElement` zur Speicherung der Werte verwenden.
Der Entwurf dieser Klasse bleibt Ihnen überlassen. Beachten Sie jedoch bei Ihrer Implementierung das Geheimnisprinzip.
Zusätzliche Hilfsmethoden sind erlaubt. Testen Sie ihre Lösung, indem Sie den in StapelTest.java enthaltenen Testfall ausführen. (31 Punkte)

## Meine eingereichte Lösung:

Alle Tests des Testfalls wurden erfolgreich bestanden.

Keine Garantie auf Richtigkeit oder Vollständigkeit.

Bitte ändert die Kommentare & Variablennamen, wenn ihr die Lösung einreichen wollt.

```java
public class Stapel {

	//Klassenattribut
	private StapelElement anfang;

	//Konstruktor
	public Stapel() {
		this.anfang = null;
		//beim Erschaffen der Liste--> anfang = null;
	}
```

```
	public void fuegeElementHinzu(String s) {
		StapelElement neuesElement = new StapelElement(s, this.anfang);
		//neues Element vom Typ StapelElement wird erzeugt
		//es werden der zu speichernde String und das vorhergegangene Element
		übergeben
		this.anfang = neuesElement;
		//das Attribut für das Element an erster Stelle wird aktualisiert
	}

	public String entferneOberstesElement() {
		if(this.anfang != null) {
			String entfernt = this.liefereOberstesElement();
			this.anfang = this.anfang.naechstesElement();
			return entfernt;
			//entfernt den zuletzt hinzugefügten String und liefert diesen
			zurück
		}
		return null;
		//Rückgabe bei leerer Liste
	}

	public String liefereOberstesElement() {
		if(this.anfang != null) {
			return this.anfang.inhalt();
			//liefert den zuletzt hinzugefügten String zurück
		}
		return null;
		//Rückgabe bei leerer Liste
	}

	public long liefereGroesse() {
		return this.liefereGroesse(this.anfang);
	//liefert die Anzahl der gespeicherten Elemente zurück
	//ruft rekursive Hilfsmethode auf
	}
	public long liefereGroesse(StapelElement aktuellesElement) {
			if(aktuellesElement == null) {
				return 0;
			}
			return liefereGroesse(aktuellesElement.naechstesElement()) + 1;
			//rekursive Zählschleife

	}

	public boolean istLeer() {
		if(this.anfang == null) {
			return true;
		}
		return false;
	//liefert true zurück, wenn der Stapel keine Elemente enthält, ansonsten false
	}
}




public class StapelElement {

	private String inhalt;
```

```
	private StapelElement naechstesElement;

	StapelElement(String s, StapelElement e){
		this.inhalt = s;
		this.naechstesElement = e;
	}

	//Getter
	public String inhalt() {
		return this.inhalt;
		//gibt den String-Inhalt des Elements zurück
	}
	//Getter
	public StapelElement naechstesElement() {
		return this.naechstesElement;
		//gibt den Verweis auf das nächste Element zurück
	}

}
```