Übungsaufgabe Schlümpfe

Sinn dieser Übung ist es, den Umgang mit dem Konzept der Liste zu lernen. In der Liste sollen mit Hilfe der vorgegebenen Klasse `SmurfNode` Schlümpfe (Klasse `Smurf`) verwaltet werden. Die Schlümpfe in der `SmurfList` sollen zu jedem Zeitpunkt nach der Größe sortiert sein. Der erste Schlumpf in der Liste soll der kleinste sein, danach folgen Schlümpfe in aufsteigender Größe.
Außerdem sollen in der Liste nur Schlümpfe mit einer schlumpfkonformen Größe sein, d.h. es dürfen sich nur Schlümpfe in der Liste befinden deren Größe den durch `SMURF_MAXIMAL_SIZE` gegebenen Wert nicht überschreitet, bzw. nicht unter dem durch `SMURF_MINIMAL_SIZE` gegebenen Wert liegt.

Sie benötigen zusätzlich die beiden weiteren Klassen `Smurf` und `SmurfNode`. Die beiden Klassen bestehen nur aus Gettern und Settern und sollen nicht weiter modifiziert werden.

Aufgabe 1 addSmurf

Mit dieser Methode soll ein Schlumpf in die Liste eingereiht werden. Falls der Schlumpf zu groß oder zu klein ist, soll er NICHT in die Liste eingefügt werden. Andernfalls soll er so einsortiert werden, dass alle Schlümpfe vor ihm kleiner oder gleich groß sind und alle nachfolgenden

Schlümpfe mindestens so groß wie der einzusortierende Schlumpf.

Aufgabe 2 containsSmurf

Mit dieser Methode soll geprüft werden, ob ein Schlumpf in der Liste ist. Der Rückgabewert soll `true` sein, wenn der Schlumpf in der Liste vorhanden ist, ansonsten `false`.

Aufgabe 3 SmurfPosition

Mit dieser Methode soll die Position eines Schlumpfes in der Liste zurückgegeben werden. Der erste Schlumpf in der Liste soll, im Gegensatz zu Arrays, die Position 1 haben. Ist der Schlumpf nicht in der Liste zu finden, soll der Rückgabewert -1 sein.

Aufgabe 4 SmurfPosition

Die gleiche Methode wie zuvor, nur, dass der Schlumpf nicht durch ein Schlumpf-Objekt, sondern durch seinen Namen identifiziert werden soll.

HINWEIS: Nutzen Sie zum Vergleich die Methode String.compareTo(String). Wenn Sie eine Variable vom Typ String haben, z.B. String `name`, können Sie die Methode so aufrufen:
`name.compareTo(andererString)`
Ist der Rückgabewert 0, haben die beiden Strings `name` und `andererString` den gleichen Wert.

http://docs.oracle.com/javase/6/docs/api/java/lang/String.html#compareTo(java.lang.String)

ERLÄUTERUNG: Die beiden Methoden `schlumpfPosition` tragen offensichtlich den gleichen Namen. Für Java ist bei einem Aufruf der Methode dennoch klar, welche Methode aufgerufen werden soll, da dies an Hand des Parameters unterschieden werden kann. Ist der Parameter ein String, muss die untere Methode gemeint sein, ist es ein Schlumpf-Objekt muss die obere Methode aufgerufen werden.
Das Phänomen mehrere Methoden gleichen Namens, aber unterschiedlicher Parametrisierung zu haben, nennt man Überladung oder Overloading.

Aufgabe 5 downsizeSmurfs

Mittels dieser Methode sollen alle Schlümpfe in der Liste um den durch `cm` gegebenen Wert verkleinert werden. Sinkt dabei die Größe einzelner Schlümpfe unter den durch `SMURF_MINIMAL_SIZE` gegebenen Wert, sollen diese aus der Liste entfernt werden.

Aufgabe 6 stretchSmurfs

Mittels dieser Methode sollen alle Schlümpfe in der Liste um den durch cm gegebenen Wert vergrößert werden. Steigt dabei die Größe einzelner Schlümpfe über den durch `SCHLUMPF_MAXIMAL_GROESSE` gegebenen Wert, sollen diese aus der Liste entfernt werden.