Exercise 1: Implement a dynamic array

Summary: Develop an own dynamic array (linked list) for string variables. You read in continuously single words from the console. You test if these words are already stored in your dynamic array. New words are added at the end of the array. If the word already exists then you increase the counter for the existing word. Typing "end" ends the loop and you list the content of the dynamic array. You enter a second loop that removes the words you typed in. This loop is also terminated by typing "end". The content of the dynamic array is listed.

Task 1: Implement ListItem (2.5 points)

Each word is represented by an instance of the class **ListItem**.

The word is stored in *theWord*.

counts states how often you typed it in.

The value of *position* corresponds to its position in the dynamic array.

nextItem is a reference to the next element in the dynamic array.

The constructor initializes *theWord* and *position* and sets *counts* = 1.

ListItem
−nextItem : ListItem −theWord : String −counts, position : int
+ListItem(String, int) +attachItem(ListItem) : void +getWord() : String +increaseCounts() : void +hasNext() : boolean +getNext() : ListItem +toString() : String

attachItem(arg) attaches an instance of **ListItem** to *nextItem*. *getWord()* returns the value of *theWord*. *increaseCounts()* increases the value of *counts* by 1. *hasNext()* checks if an object is attached to *nextItem* and *getNext()* returns it. *toString()* should return "The word *theWord* is placed at the position *position* and has occured *counts* times."

Task 2: Implement DataBase (4.5 points)

start stores the address of the first element of the dynamic array.

thePosition is a counter for the length of the dynamic array while we expand it. Its value sets *position* in instances of **ListItem**.

The constructor sets *thePosition* = 1.

DataBase
−start : ListItem −thePosition : int
+DataBase() +add(String) : void +delete(String) : boolean +toString() : String

The method *add(arg)* checks if the linked list of instances of **ListItem** contains one instance with a value of *theWord* that equals *arg*. If it does, then the value of *counts* of that instance is increased by 1. If not then the value of *thePosition* is increased by one and a new instance of **ListItem** is created with *arg* and added to the end of the linked list.

The method *delete(arg)* goes through the linked list and determines if one instance of **ListItem** has a value of *theWord* that equals *arg*. If it finds one then it removes this instance and closes the gap in the linked list. Its return value is *true* if the linked list contains instances of **LinkedList** and *false* if the linked list no longer has any element.

Task 3: Implement Exercise (3 points)

This class contains the method *main*. Implement a loop that reads in words from the console until you type "end". Catch IO errors. The words are added to the linked list. Print out the content of the database after this loop. A second loop asks for words that should be removed. This loop ends either if you type "end" or if the database is empty. Print out the content of the database after this loop.