Einfacher Plotter, Teil 5: Tokenize

Im letzten Teil geht es darum, einen Eingabestring einzulesen und diesen in ein Array von Tokens umzuwandeln. Dazu muss verstanden werden, wie Tokens funktionieren.

Falls du irgendwo Probleme hast, zögere bitte nicht, um Hilfe zu fragen! Wenn du irgendwo hängst, helfen wir dir gerne weiter! Melde dich bei einem der zuständigen Betreuer*innen: Lorenz Braun, Fabian Sand, Robin Fritz, Furkan Keserci, Julia Kisela. Wir sind über Discord, Mattermost und E-Mail erreichbar. Wen könnt ihr für Hilfe ansprechen? (E-Mail, Mattermost, Discord)

- **Robin Fritz: frro1020@h-ka.de, @robin.fritz, Snobo#4754**
- **Julia Kisela: kiju1012@h-ka.de, @julia.kisela, lohikäärme#3336**
- **Furkan Keserci: kefu1011@h-ka.de, @furkan, Killnexx#9999**
- **Lorenz Braun: brlo1014@h-ka.de, @lorenzbraun, MrRaptorious#7632**
- **Fabian Sand: safa1017@h-ka.de, @safa1017, Fabian S#7613**

Lernmaterial

Ein Token besteht aus einem oder mehreren Symbolen, die zusammengehören. Klammern, Unäre Operatoren, Binäre Operatoren, Unäre Funktionen sind jeweils ein Token. Eine Zahl, die auch aus mehreren Chars bestehen kann, ist ein Token. X ist ein Token.

Wenn dir nicht klar ist, auf welche Weise du Strings verarbeiten kannst, solltest du das noch einmal nachlesen. Z.B.: String.charAt(), Character.isdigit(), Character.isAlphabetic(), String.substring().

Verständnisfragen

- Was sind Tokens?
- Kennzeichne in $f(x) = sin(5 * x - (3 + (-3,5)))$ alle Tokens. Wie erkennst du, um welche Art von Token es sich handelt?
- Wenn du von links nach rechts durchgehst, wie findest du heraus, was eine Zahl ist?
- Wie kannst du negative Zahlen finden?
- Hast du auch eine Idee, wie man negative Funktionen (z. B. -sin(x)) identifizieren kann? (Dies ist eine Zusatzaufgabe, eine funktionierende Ausgabe für -1*sin(x) reicht aus.)

Aufgabenstellung

Wichtiger Hinweis: Im bereitgestellten Code finden sich Dokumentationskommentare, welche Aufschluss auf die Funktionsweise oder die zu implementierende Funktionalität geben. Bitte deshalb den Code und die Kommentare aufmerksam studieren! Bitte verändere gegebene Klassennamen und Methodennamen nicht, da diese für die Tests verwendet werden!

In diesem Teil bekommst nur einen neuen Testordner von uns, um deinen Code zu testen, ersetze den alten durch den neuen. (Die bisherigen Testfälle wird es trotzdem weiterhin geben, aber zusätzlich kommen neue hinzu.)

Um nun aus einem String ein Array aus Tokens zu machen, implementiere in der Klasse Token die Methode tokenize. Ihre Methoden-Signatur und Javadoc kannst du hier sehen:

```
/**
 * tokenize teilt den String s, der einen
    mathematischen Ausdruck enthaelt, in ein Array von
    Tokens (Sinneinheiten, also z. B. Zahlen, einzelne
    Klammern, Operatoren, Funktionsnamen) auf.
 *
 * @param s Mathematischer Ausdruck als String.
 * @return Array von Tokens (einzelne mathematische
    Symbole) in dem String.
*/
public static Token[] tokenize(Sting s) {}
```

Du musst dir dazu überlegen, auf welche Weise du den String zerteilst, um die richtigen Tokens zu erhalten.

Als letzter Schritt, um deinen Plotter funktionsfähig zu bekommen, musst du noch eine Methode in die Expression-Klasse einfügen, die deinen eingelesenen String verarbeitet. Diese Methode musst du dann auch in der Grid-Klasse aufrufen. Die Signatur dieser Methode ist folgende:

```
public static Expression parseInfixString(String s) {
    return parseRPN(ShuntingYard.convertToRPN(
                                Token.tokenize(s))); }
```

Schreibe zu dieser Methode auch eine ausführliche Javadoc. Sie sollte genau erklären, was hier alles passiert. Gehe dazu auf alle drei aufgerufenen Methoden ein. Beantworte dabei folgende Fragen: Wo werden Tokens benutzt? Wo Expressions? Wo ist ein Syntaxbaum?

Checkliste

- Strings einlesen und zu Tokens machen
- Mindestanforderung: (-1)*x statt -x und (-1)*sin(x) statt -sin(x)
- Eingelesener String wird auch geplottet