Einfacher Plotter, Teil 4: Shunting-Yard-Algorithmus

Ziel dieses Teiles ist es, mit einer Eingabe in Infix-Notation (also intuitiv mit Klammern und Punkt-vor-Strich, …) einen Plot zu erzeugen. Dazu muss noch ein Algorithmus entwickelt werden, welcher aus einer Funktion in Infix-Schreibweise eine Funktion in Umgekehrter Polnischer Notation erzeugt. (Umgekehrte Polnische Notation können wir ja seit dem letzten Teil schon plotten.)

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

Um uns an dieser Stelle nicht mit String-Konkatenationen beschäftigen zu müssen, wurden im voherigen Teil schon Tokens eingeführt. Der Input der Funktion ist also ein Array aus Tokens in Infix-Schreibweise, z. B.

```
new Token[]{
        new FunctionToken("sin")
        new Token(TokenType.OPENING_PARENTHESIS),
        new NumberToken(8),
        new Token(TokenType.PLUS),
        new NumberToken(3),
        new Token(TokenType.MINUS),
        new NumberToken(5),
        new Token(TokenType.TIMES),
        new Token(TokenType.X),
        new Token(TokenType.CLOSING_PARENTHESIS),
}
```

Der Output soll ein umsortiertes Array sein, welches dieselben Tokens enthält, aber eben im Umgekehrter Polnischer Notation.

Der Shunting-Yard-Algorithmus wurde von Dijkstra entwickelt, um eben genau dieses Problem zu lösen. Heutzutage benutzt man meist andere Algorithmen, die auf weiterführendem Wissen aufbauen. (Z. B. kann man auch Parsing-Algorithmen durch Implementierungen von Grammatiken programmieren. Aber das ein andermal.)

Mehr Infos auf der Wikipedia-Seite:
https://en.m.wikipedia.org/wiki/Shunting-yard_algorithm

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!

Importiere zuerst das gegebene Paket shuntingyard, dieses findest du im Ilias. Du bekommst auch wieder 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).

Im Paket shuntingyard findest du eine Klasse ShuntingYard mit einer Methode convertToRPN. Diese Methode soll den Shunting-Yard-Algorithmus implementieren, welcher aus Infix-sortierten Tokens Tokens in Umgekehrter Polnischer Notation übersetzt. Des Weiteren gibt es zwei neue Hilfsmethoden als txt-Datei im Ilias, die du benutzen kannst: ranking und isLeftAssociative. Diese kannst du in die Klasse Token in deinen Code kopieren.

Auf der Wikipedia-Seite steht Pseudo-Code zum Shunting-Yard-Algorithmus. Schreibe nun diesen Pseudo-Code in Java-Code um.

In der Grid-Klasse müssen nun an die auszuführende Stelle Tokens in Umgekehrter Polnischer Notation geschrieben werden. Ein Beispiel sieht so aus:

```
Expression e = Expression.parseRPN(ShuntingYard.convertToRPN(
    new Token[]{
            new NumberToken(-1),
            new Token(TokenType.TIMES),
            new FunctionToken("exp"),
            new Token(TokenType.OPENING_PARENTHESIS),
            new Token(TokenType.X),
            new Token(TokenType.CLOSING_PARENTHESIS)
}))
```

Füge bitte die Funktionen $sin(x)$, $log(5 * (3 + x))$, $x^2 + 3$ in der richtigen Schreibweise in deine Grid-Klasse.

Checkliste

- convertToRPN in ShuntingYard
- Beispiele lassen sich plotten und sehen richtig aus