SWE	Softwareentwicklung 3 Medizin- und Bioinformatik	WS 22/23, Übung 4

Name: Lena Oppitz: **Aufwand in h:** 12

Punkte: _____**Kurzeichen Tutor/in:** _____

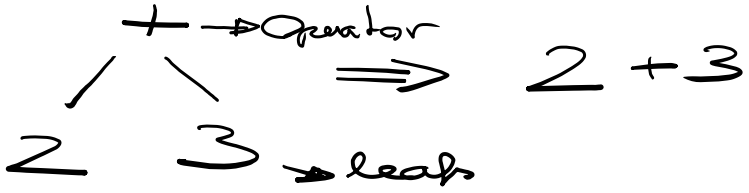
Inhalt

Lösungsidee....2
Ausdrucksbaum....2
NameList....2
NameListMap....2
Exceptiones....2
Pfc::Scanner....2
SynthaxTree....2
Parser....3
Grammatik:....3
Parse....3
Private Datenkomponenten....3
Is_tb_Funktionen....3
parse_Program()....3
Parse_Output()....3
Parse PExpression....3
Parse_Term/ Factor/Monom....4
Parse_WMonom....4
Exponent....4
Code....4
Testfälle....4

Lösungsidee

Ausdrucksbaum

Durch diesen Baum können arithmetische Berechnungen gespeichert werden. Die Zahlen sind dabei immer die Blätter des Baumes und die Operatoren die inneren Knoten bzw. der Wurzelknoten.



Ein weiter Vorteil des Baumes ist, dass so auch Variablen, gespeichert werden können, die aus zusammengesetzten arithmetischen Ausdrücken bestehen. Weiters kann der baum sehr einfach in Pre-, Post- oder Infix-Notation umgewandelt werden. Es braucht auch keine Klammern, da die Position im Baum, die Rechenreihenfolge vorgibt.

NameList

Die NameList ist eine Abstrakte Klasse und hat beinhaltet eine Map, die als Key die Variable und als Value einen Ausdrucksbaum speichert. Sie ist die Basis für NameListMap.

NameListMap

Diese Klasse beinhaltet die Funktionsimplementierung der in der Basisklasse definierten Funktionen. Es besteht die Möglichkeit Variablen in die Map zu speichern und diese auch zu lesen. Diese Klasse ist als generische Klasse implementiert würde es auch ermöglichen mit anderen Datentypen zu arbeiten.

Exceptiones

Es sollen verschiedene Exceptiones vorhanden sein:

- Fehler des Parsers (parser_exception), soll geworfen werden, wenn ein ungültiges Zeichen eingelesen wird.
- DivByZero: Wenn eine Division durch 0 versucht wird.

Pfc::Scanner

Der Scanner ließt nacheinander die Zeichen des Eingabestrings und bietet nützliche Funktionen für die Arbeit mit den Zeichenketten.

StNode (abstrakte Klasse)

Die Knoten sind für den Bau des Ausdruckbaumes wichtig. Es gibt drei verschiedene Knoten:

- Valueknoten: für Zahlen
- Operatorknoten: für Operatoren
- Identifier-Knoten: für Variablen

SynthaxTree

Beinhaltet die Funktionen des Baumes(evaluate, print).

- Print() -> Schreibt den Baum in Pre-, Post- oder Infix-Notation auf die Konsole+
- Evaluate() -> Berechnet den Wert des Baumes
- Print2d Upright -> gehört hier implentiert

Parser

Grammatik:

```
Program             =     {output|assignment}
Output              =     "print" "(" Expression ")" ";"
Assignment   =            "set" "(" Identifier "," Expression ")" ";"
Expression   =            Term {AddOp Term}
Term                =     Faktor {MultOp Faktor}
Factor              =     [AddOp] UFactor
UFactor             =     Monom | KExpression
Monom        =            WMonom [Exponent]
PExpression  =            "(" Expression ")"
WMonom              =     Identifier | Real
Exponent                  ="^" [AddOp] Real
AddOp        =            "+" | "-"
MultOp              =     "*" | "/"
```

Parse

Sollen den Scanner initialisieren und die Schlüsselwörter print/set im Scanner speichern.

Private Datenkomponenten

Der Scanner um Elemente des Datenstroms einzulesen.
Die NameListMap um Variablen zu speichern.

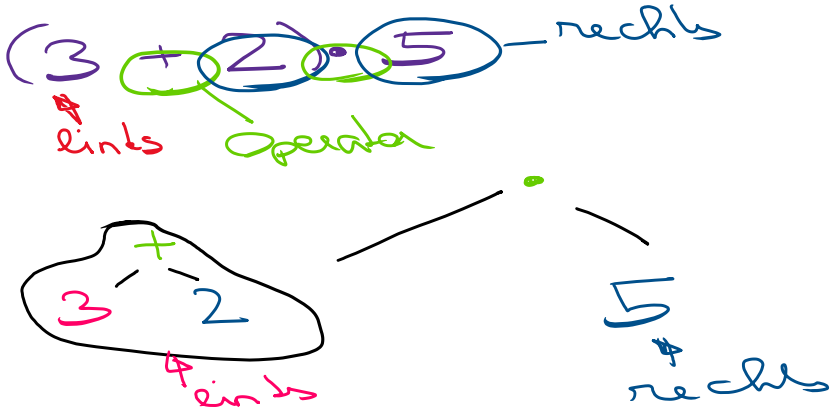
Is_tb_Funktionen

Sollen gemäß der Grammatik erstellt werden. Überprüfen, das eingelesene Wort und liefern einen Boolean zurück.

parse_Program()

Diese Funktion soll in Ausgabe und dem Setzen von Variablen aufteilen. Der SyntaxTree soll hier angelegt werden und nach dem vollständen Erstellen evaluiert werden.

In den parse-Funktionen soll immer der linke Knoten zwischengespeichert werden. Dann der Operator eingelesen werden und der rechte Wert rechts an den Baum gehängt werden. Der linke Wert soll anschließend links am Baum angehängt werden.



Parse_Output()

Parse_PExpression aufrufen.

Parse PExpression

Klammer einlesen und Parse_Expression aufrufen

Parse_Term/ Factor/Monom

Die Knoten so wie oben in der Zeichnung anfügen die jeweilige Funktion lauf der Grammatik aufrufen.

Parse_WMonom

Hier sollen Zahl und Variable in den Baum eingefügt werden -> Blätter des Baumes

Exponent

Kann behandelt werden wie ein binärer Operator.

Code

SWE_Oppitz_Ue07 -> File Parser.h/cpp

Testfälle

Da das Programm nicht funktioniert. -> mögliche Testfälle:

- Grundrechnungsarten (+, -, *, /) mittels einfachen Berechnungen z.B. print(3+2);
- Komplexere zusammengesetzte Ausdrücke z.B. print((2 + 3) * (4 - 2));
- Set und print testen
- Ausdrücke mit Variablen (set x, (4+1));
- Die Tests vom Übungszettel
- Ungültige Eingaben: (print(3 +);
- Undefinierte Variablen
- Negative Zahlen z.B. print(-5 + 3);
- Gleitkommerzahlen
- DivisionByZeroError z.B. print(3/0);
- Mit Strich-Punkt am Ende und ohne
- Mit und ohne Klammern
- Berechnungen mit Exponent z.B. 4 ^ 0.5
- Print(); set(); -> Falsche Anwendung der Funktionen
- Set(x, 2 +); Unvollständige Eingaben
- Print(2 * * 3);-> zwei Operator
- Set(2, y); -> Set Parameter vertauscht

Mögliche Fehlerquellen:

Das Programm bleibt in einer Endlosschleife hängen. Den genauen Fehler habe ich nicht gefunden bzw. das Programm ist aufgrund Zeitmangel nicht fertig implementiert. Es könnte sein, dass der Scanner nicht weitergerückt würde oder eine Funktion in der Funktion selber aufgerufen wird.