XLang

Max Heidinger, Pascal Riesinger

April 15, 2019

1 Einleitung

XLang ist eine Programmiersprache entwickelt im Rahmen der Vorlesung *Compilerbau* an der Dualen Hochschule Baden-Württemberg Karlsruhe.

2 Syntax

2.1 Grammatik

Die folgende Grammatik in der Extended-Backus-Naur-Form beschreibt die Syntax von XLang. Wiederholungen, die in geschweiften Klammern ({ }) gekennzeichnet sind, schließen null Wiederholungen (das leere Wort) mit ein.

```
1  Startsymbol: Programm
2
3  Programm ::= NamenTeil
4               TeilSeparator
5               VariablenTeil
6               TeilSeparator
7               CodeTeil.
8  TeilSeparator ::= '===' {'='}.
9
10 NamenTeil ::= Zeichen {Zeichen}.
11
12 VariablenTeil ::= {Variable ';'}.
13 Variable ::= [ImportExportFlag] VariablenName VariablenTyp.
14 ImportExportFlag ::= ImportFlag | ExportFlag.
15 ImportFlag ::= '->'.
16 ExportFlag ::= '<-'.
17 VariablenName ::= Buchstabe {Zeichen}.
18 VariablenTyp ::= 'int' | 'float' | 'string'.
19
20 CodeTeil ::= {Instruktion}.
21 Instruktion ::= (Zuweisung ';') | Verzweigung | Schleife | Kommentar.
22 Kommentar ::= '//' {AlleZeichen}.
23
24 Zuweisung ::= VariablenName '=' EvaluationsWert.
25 EvaluationsWert ::= Wert
26                    | Addition
27                    | Subtraktion
28                    | Multiplikation
29                    | Division
30                    | RestDivision
31                    | '(' EvaluationsWert ')'.
32 Wert ::= VariablenName | Literal.
33 Literal ::= String | Int | Float.
34 Addition ::= EvaluationsWert '+' EvaluationsWert.
35 Subtraktion ::= EvaluationsWert '-' EvaluationsWert.
36 Multiplikation ::= EvaluationsWert '*' EvaluationsWert.
37 Division ::= EvaluationsWert '/' EvaluationsWert.
38 RestDivision ::= EvaluationsWert '%' EvaluationsWert.
39
```

```
40 Verzweigung ::= IfVerzweigung.
41 Schleife ::= WhileSchleife | ForSchleife.
42 InstruktionsBlock ::= '{' {Instruktion} '}'.
43
44 IfVerzweigung ::= 'if' Vergleich InstruktionsBlock
45                   {ElseIfVerzweigung} [ElseVerzweigung].
46 Vergleich ::= GleichheitsVergleich
47              | UngleichheitsVergleich
48              | KleinerVergleich
49              | GroesserVergleich.
50 GleichheitsVergleich ::= EvaluationsWert '==' EvaluationsWert.
51 UngleichheitsVergleich ::= EvaluationsWert '!=' EvaluationsWert.
52 KleinerVergleich ::= EvaluationsWert '<' EvaluationsWert.
53 GroesserVergleich ::= EvaluationsWert '>' EvaluationsWert.
54 ElseIfVerzweigung ::= 'else if' Vergleich InstruktionsBlock.
55 ElseVerzweigung ::= 'else' InstruktionsBlock.
56
57 WhileSchleife ::= 'while' Vergleich InstruktionsBlock.
58 ForSchleife ::= 'for' Zuweisung ';' Vergleich ';' Zuweisung InstruktionsBlock.
59
60 Zeichen := Buchstabe | Ziffer
61 Buchstabe ::= 'A'|...|'Z'|'a'|...|'z'.
62 AlleZeichen ::= Zeichen | Sonderzeichen.
63 Sonderzeichen ::= ' ' | '!' | ... | '@'.
64 Ziffer ::= '0'|...|'9'.
65 String ::= ''''
66           | '""'
67           | ''' {AlleZeichen} '''
68           | '"' {AlleZeichen} '"'.
69 Int ::= Ziffer {Ziffer}.
70 Float ::= Int '.' Int.
```

2.2 Beschreibung

2.2.1 Struktur

Ein XLang Programm ist in 3 Teile aufgeteilt. Diese werden durch eine Folge von Gleichheitszeichen (=) voneinander abgetrennt, welche mindestens 3 Zeichen lang sein und in einer eigenen Zeile stehen müssen. Zunächst wird der Name des Programmes vermerkt. Anschließend folgt der Variablendeklarationsteil, in welchem alle im Programmteil verwendeten Variablen deklariert werden müssen. Der dritte Teil des Programmes ist der sogenannte Programmteil, welcher alle Instruktionen beinhaltet.

Leerzeichen können überall im Programm eingefügt werden und werden primär verwendet, um Schlüsselwörter von anderen Zeichen zu trennen. Kommentare können im Codeteil nur in eigenen Zeilen mit `//` markiert verwendet werden.

2.2.2 Variablen

Es gibt in XLang drei Datentypen:

- Ganzzahlen, deklariert durch den Typ `int`.
- Gleitkommazahlen, deklariert durch den Typ `float`.
- Zeichenfolgen, deklariert durch den Typ `string`.

Wie bereits angemerkt müssen alle im Programm verwendeten Variablen im Variablenteil delariert werden. Da XLang ohne Funktionen und Unterprogramme auskommt, müssen Variablen, welche als Eingabeparameter fungieren sollen, mit einer sogenannten `ImportFlag` gekennzeichnet werden. Eine Deklaration einer Eingabevariable sieht dann bespielsweise wie folgt aus: `-> name string`.

Die Eingabevariablen werden beim Programmstart aus der Kommandozeile ausgelesen. Dabei ist die Reihenfolge der Deklaration gleich der Reihenfolge der Übergabe.

Variablen, deren Wert am Ende des Programmes ausgegeben werden soll, müssen mit einer `ExportFlag` gekennzeichnet werden. Eine Deklaration einer Ausgabevariable sieht dann beispielsweise wie folgt aus:

`<- ergebnis float`. Die Ausgabevariablen werden in der Reihenfolge ausgegeben, in welcher sie deklariert wurden.

Es gibt keine Möglichkeit, Variablen während der Ausführung auszugeben oder einzulesen.

Variablen können Literale, also konstante Werte oder Ausdrücke zugewiesen werden. Ein Ausdruck ist bei numerischen Werten entweder eine andere Variable oder eine mathematische Formel. Bei Rechnungen können die Grundrechenarten (Addition, Subtraktion, Multiplikation und Division). verwendet werden. Hierbei gilt die gleiche Priorisierung, wie im C-Standard.

Bei Zeichenketten ist ein Ausdruck entweder eine andere Variable oder eine Verkettung von mehreren Literalen und Variablen über den Additionsoperator.

2.2.3 Konditionen

Die Konditionalausdrücke in XLang folgen dem Beispiel anderer Programmiersprachen. Wenn der Vergleich nach dem `if` zu einem logisch "wahren" Wert evaluiert, werden alle Anweisungen im folgenden Anweisungsblock ausgeführt. Der `if`-Anweisung können optional beliebig viele `else if`-Anweisungen folgen. Diese werden nur ausgewertet, wenn der Vergleich der voranstehenden Anweisung zu einem unwahren Wert evaluiert wurde. Abgeschlossen werden kann ein Konditionalausdruck durch ein `else`. Die Anweisungen im `else`-Block werden ausgeführt, wenn kein vorheriger Vergleich im Konditionalausdruck zu einem wahren Wert evaluiert werden konnte.

2.2.4 Kommentare

Es gibt in XLang nur Zeilenkommentare. Diese werden mit doppelten Schrägstrichen eingeleitet und können jegliche Zeichen enthalten. Kommentare können im Codeteil vorkommen und enden mit dem Ende der Zeile. Kommentare werden vom Compiler ignoriert und haben keine Bedeutung für das Programm.

2.2.5 Schleifen

Es gibt in XLang zwei Arten von Schleifen. Eine `while`-Schleife führt dein Codeblock so lange aus, bis die Bedingung im Kopf der Schleife zu einem unwahren Wert evaluiert wird. While-Schleifen sind kopfgesteuert.

Die zweite-Art der Schleifen in XLang sind `for`-Schleifen. Diese erlauben es, eine Zuweisung im Fuß der Schleife durchzuführen. Zunächst wird eine Variable zugewiesen, welche dann in der Kondition in der Schleife verwendet wird. Der dritte Teil der Schleife weißt der Variable einen neuen Wert zu. Vor jedem Durchlauf wird der Vergleich durchgeführt, dann der Programmblock, falls der Vergleich zu einem wahren Wert evaluiert, nach dem Ausführen des Programmblockes wird dann die Zuweisung ausgeführt.

3 Verwendetes Allgemeinwissen

In XLang wird verschiedenes Allgemeinwissen verwendet. Ein Beispiel ist, dass eine deklarierte Variable im Code verwendet werden kann. Auch das Verhalten der Kontrollstrukturen folgt den "Regeln" des Allgemeinwissen. Bei numerischen Berechnungen werden die allgemein bekannten Rechenregeln angewandt.

4 Beispielprogramme

4.1 Berechnung der Fakultät

```
1 Factorial Calculation
2 ======================
3
4 -> input_num int;
5 <- result int;
6 iterator int;
```

```
7
8  ================================
9
10 result = 1;
11 if input_num != 0 {
12   for iterator = 2; iterator < input_num + 1; iterator = iterator + 1 {
13     result = result * iterator;
14   }
15 }
```

4.2 Berechnung des größten gemeinsamen Teilers

```
1  Groesster Gemeinsamer Teiler
2  ================================
3
4  -> a int;
5  -> b int;
6  c int;
7  <- ergebnis int;
8
9  ================================
10
11 // Euklidischer Algorithmus
12 if a < b {
13   c = b;
14   b = a;
15   a = c;
16 }
17
18 c = 1;
19 while c > 0 {
20   c = a % b;
21   b = a / b;
22   if c > 0 {
23     a = b;
24     b = c;
25   }
26 }
27
28 ergebnis = c;
```

5 BNF

```
1  Startsymbol: Programm
2
3  Programm ::= NamenTeil
4               TeilSeparator
5               VariablenTeil
6               TeilSeparator
7               CodeTeil.
8  TeilSeparator ::= '==' TeilSeparatorZeichen.
9  TeilSeparatorZeichen ::= '=' | ('=' TeilSeparatorZeichen).
10
11 NamenTeil ::= Zeichen | (Zeichen NamenTeil).
12
13 VariablenTeil ::= Variable | (Variable VariablenTeil).
14 Variable ::= (ImportExportFlag VariablenName VariablenTyp ';')
15              | (VariablenName VariablenTyp ';').
16 ImportExportFlag ::= ImportFlag | ExportFlag.
17 ImportFlag ::= '->'.
18 ExportFlag ::= '<-'.
19 VariablenName ::= Buchstabe | Buchstabe Bezeichner.
20 VariablenTyp ::= 'int' | 'float' | 'string'.
21
22 CodeTeil ::= Instruktion | Instruktion CodeTeil.
23 Instruktion ::= (Zuweisung ';') | Verzweigung | Schleife | Kommentar.
24 Kommentar ::= '//' | '//' Zeichenfolge.
25
26 Zuweisung ::= VariablenName '=' EvaluationsWert.
```

```
27 EvaluationsWert ::= Wert
28                     | Addition
29                     | Subtraktion
30                     | Multiplikation
31                     | Division
32                     | RestDivision
33                     | '(' EvaluationsWert ')'.
34 Wert ::= VariablenName | Literal.
35 Literal ::= String | Int | Float.
36 Addition ::= EvaluationsWert '+' EvaluationsWert.
37 Subtraktion ::= EvaluationsWert '-' EvaluationsWert.
38 Multiplikation ::= EvaluationsWert '*' EvaluationsWert.
39 Division ::= EvaluationsWert '/' EvaluationsWert.
40 RestDivision ::= EvaluationsWert '%' EvaluationsWert.
41
42 Verzweigung ::= IfVerzweigung.
43 Schleife ::= WhileSchleife | ForSchleife.
44 InstruktionsBlock ::= '{' InstruktionsListe '}'.
45 InstruktionsListe ::= Instruktion | Instruktion InstruktionsListe.
46
47 IfVerzweigung ::= ('if' Vergleich InstruktionsBlock)
48                   | ('if' Vergleich InstruktionsBlock ElseVerzweigung)
49                   | ('if' Vergleich InstruktionsBlock ElseIfVerzweigungen)
50                   | ('if' Vergleich InstruktionsBlock
51                       ElseIfVerzweigungen ElseVerzweigung).
52 Vergleich ::= GleichheitsVergleich
53              | UngleichheitsVergleich
54              | KleinerVergleich
55              | GroesserVerlgleich.
56 GleichheitsVergleich ::= EvaluationsWert '==' EvaluationsWert.
57 UngleichheitsVergleich ::= EvaluationsWert '!=' EvaluationsWert.
58 KleinerVergleich ::= EvaluationsWert '<' EvaluationsWert.
59 GroesserVerlgleich ::= EvaluationsWert '>' EvaluationsWert.
60 ElseIfVerzweigung ::= 'else if' Vergleich InstruktionsBlock.
61 ElseIfVerzweigungen ::= ElseIfVerzweigung
62                        | ElseIfVerzweigung ElseIfVerzweigungen.
63 ElseVerzweigung ::= 'else' InstruktionsBlock.
64
65 WhileSchleife ::= 'while' Vergleich InstruktionsBlock.
66 ForSchleife ::= 'for' Zuweisung ';' Vergleich ';' Zuweisung InstruktionsBlock.
67
68 Zeichen ::= Buchstabe | Ziffer
69 Bezeichner ::= Zeichen | (Zeichen Bezeichner).
70 AlleZeichen ::= Zeichen | Sonderzeichen.
71 Zeichenfolge ::= AlleZeichen | (AlleZeichen Zeichenfolge).
72 Sonderzeichen ::= ' ' | '!' | ... | '@'.
73 Buchstabe ::= 'A'|...|'Z'|'a'|...|'z'.
74 Ziffer ::= '0'|...|'9'.
75 String ::= ''''
76           | '""'
77           | (''' Zeichenfolge ''')
78           | ('"' Zeichenfolge '"').
79 Int ::= Ziffer | Ziffer Int.
80 Float ::= Int '.' Int.
```