**Einführung in den Compilerbau**

Gruppe : Leonard Bongard; Duy Quang Nguyen; Egemen Ulutürk

**Aufgabe 1**

a)

starters[[A]] = {0;1}

starters[[X]] = {0,a,2}

starters[[Y]] = {a}

follow[[A]] = {a}

follow[[X]] = {b, 0, a, 2}

follow[[Y]] = {b}

b) starters[[0]] ꓵ starters[[1]] = {0} ꓵ {1} = ∅

c) starters[[A]] ꓵ (starters[[ε]] U follow[[B]]) = {0,1} ꓵ ({∅} U {a} = ∅

d) Es gibt 3 Fälle :

1. 0 | (a 1)

starters[[0]] ꓵ starters[[a 1]] = {0} ꓵ {a} = ∅

2. (a 1) | (2 2 b)

starters[[a 1]] ꓵ starters[[2 2 b]] = {a} ꓵ {2} = ∅

3. 0 | (2 2 b)

starters[[0]] ꓵ starters[[2 2 b] = {0} ꓵ {2} = ∅

e) starters[[X]] ꓵ follow[[X\*]] = {0,a,2} ꓵ {b} = ∅

**Aufgabe 2**

a) Wenn man das Präfix “@” entfernt, kann es sein, dass der Ausdruck als Selektion von Strukturelementen geparst wird statt als Selektion von Record-Elementen.  
Deutlich wird es bei der Selektion von Record-Elementen mit einem Parameter:

Bsp. val Point q = Point[3.0];

Hier haben wir auf der rechten Seite des Gleichheitszeichens: ID “[“ expr “]”, die mit der Grammatik der Selektion von Strukturelementen übereinstimmt.

b) parseExpr(){

switch(currentToken[0])

case INT :

acceptIt();

break;

case „-“ :

parseNegation();

break;

case ID :

if(currentToken[1] == „[“ ) {

if(currentToken[2] == ID)

parseRecordLiteral();

else parseElementSelect();

}

else accept(ID)

break;

default: error();

**Aufgabe 3**

Idents:

alice ->

|  |  |  |
| --- | --- | --- |
| 15 | 6 | 3 |

claire ->

|  |  |  |
| --- | --- | --- |
|  | 14 | 4 |

bob ->

|  |  |  |
| --- | --- | --- |
|  |  | 11 |

Die grau markierten Zellen beinhalten die oberste Elemente des Stacks.

scopes:

|  |
| --- |
|  |
|  |
|  |
| {alice, claire} |
| {claire, bob} |
| {alice, claire} |

Die grau markierte Zelle ist das oberste Element des Stacks.

**Aufgabe 4**

Kopf der for-Schleife: ‘“for” “(“ InitVarName “=” InitValue “;” Schleifenbedingung “;” IncrVarName “=” IncrExpr “)”

Für die For Schleife müssen nun folgende Regeln gelten.

* InitVarName
  + String; ID des Index
  + Name der zu Initialisierenden Variable
  + Muss den Typ INT oder FLOAT haben
* InitValue
  + INT oder FLOAT
  + Muss der gleiche Typ wie der Typ des Feldes InitVarName sein
  + vor der ersten Iteration wird ihm der Wert des Ausdrucks zugewiesen
* Schleifenbedingung
  + BOOL
  + Wird vor jeder Iteration geprüft
    - Falls der Wert TRUE ist, dann wird der Body-Codeblock ausgeführt
    - Falls der Wert FALSE ist, dann wird die Schleife abgebrochen
* IncrVarName
  + STRING; ID des Index
  + Muss den gleichen Typ und Wert wie InitVarName haben
* IncrExpr
  + INT oder FLOAT
  + muss den selben Typen wie IncrVarName haben
  + wenn der Wert der Schleifenbedingung TRUE ist, dann wird dem Feld InitValue der Wert des Ausdrucks zugewiesen