Theorie 1

Gruppe 35
Matias Heredia Novillo, Anastasia Paschalidou, Niclas Gregor

Aufgab 1.1

a)

addExpr::= expr ‘+’ expr

ternaryExpr ::= BOOL ‘?’ expr ‘:’ expr

subvectorExpr ::= expr (subvectorOp | ‘(’ subvectorOp ‘)’ subvectorOp)

subvectorOp ::= '{' constExpr ':' expr ':' constExpr '}’

b)

primitiveType ::= ‘int’ | ‘float’ | ‘bool’

vectorType ::= ‘vector’ ‘<’ (’int’ | ‘float’) ‘>’ ‘[’ constExpr ‘]’

c)

varAssign ::= (ID | primitiveType | vectorType | matrixType) ‘=’ expr ‘;’

compoundStmt ::= ‘{’ expr* ‘}’

foreachStmt ::= ‘foreach’ ‘(’ (‘var’| ’val’) (‘int’ | ‘float’ | vectorType | matrixType) ID ‘:’ expr ‘)’ compoundStmt

d)

constExpr ::= INT | constExpr arithOp constExpr | '(' constExpr ')' | negOp constExpr
arithOp ::= '+' | '-' | '*' | '/' | '^'
negOp ::= '-'

Aufgabe 1.2

a)

```
{
  var float n;
  n = (42 - 3) / 9;
  if(n != 5) {
    n = -n;
  }
}
```

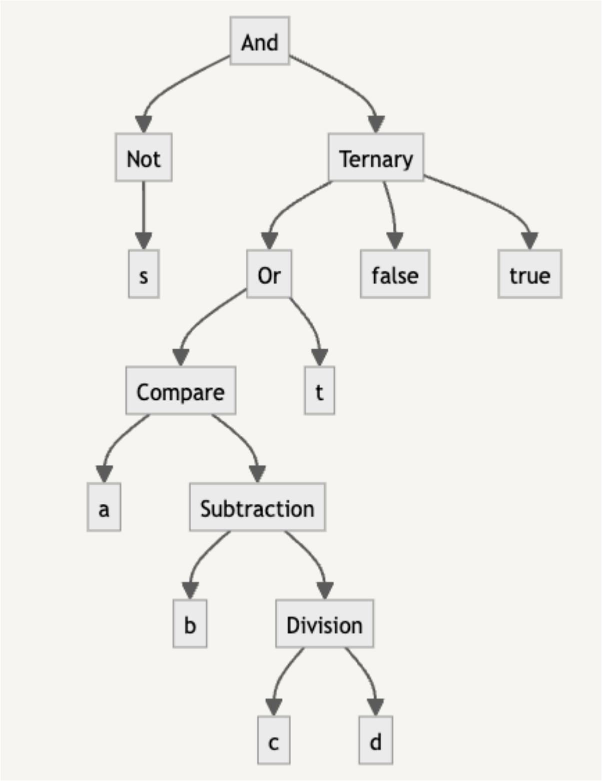
b)

```
function int extract(matrix<int> [4][4] m, int x, int y) {
  return m[x][y];
}
```

Aufgabe 1.3

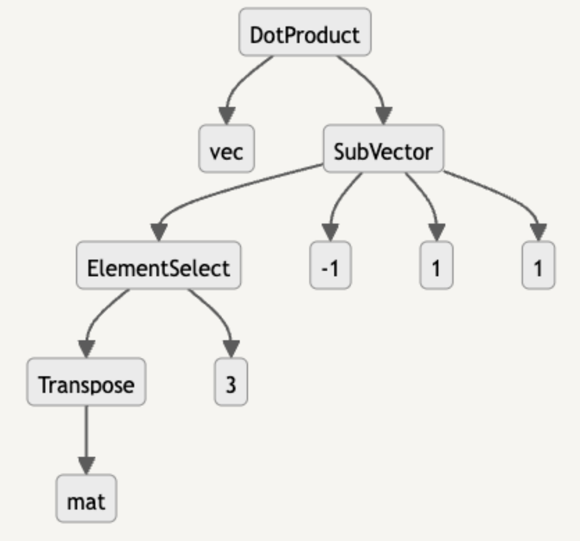
a)

```
graph TD
    A[And]
    A --> neg[Not]
    neg --> varS[s]
    A --> ternary[Ternary]
    ternary --> or[Or]
    ternary --> false[false]
    ternary --> true[true]
    or --> geq[Compare]
    or --> varT[t]
    geq --> varA[a]
    geq --> Sub[Subtraction]
    Sub --> varB[b]
    Sub --> Div[Division]
    Div--> varC[c]
    Div --> varD[d]
```



b)

```
graph TD
    A("DotProduct")
    A --> B("vec")
    A --> C("SubVector")
    C --> D("ElementSelect")
    D --> E("Transpose")
    E --> F("mat")
    D --> G("3")
    C --> H("-1")
    C --> I("1")
    C --> J("1")
```



c)

```
a) !false & ( ((12.34 >= (13 - (37 / 42))) | false) ? false : true) = false

b) vec .* (~mat)[3]{-1:1:1} = 98 //vec = [9, 8, 7] //(~mat)[3]{-1:1:1} = [18, 24, 56],  da hier 2 inkl (entspricht mat.T[3][0:2])
```