Compiler Construction
Under 12 parsers

Flip van Spaendonck & Lars Kuijpers

March 13, 2018

Parser Structure

- Bottom-up
- Separate lexer and parser
- Written in Java

Grammar Transformation

```
FArgs = id [',' FArgs]
Field = [('.hd' | '.tl' | '.fst' | '.snd') Field]
Exp == id Field
| BoolExp2
| NumRng
| FunCall
| '[]'
| '(' Exp')'
| '(' Exp, Exp')'
```

Grammar Transformation

```
BoolExp2 = BoolExp1 && BoolExp2
| BoolExp1 || BoolExp2
| NumRng == NumRng
| NumRng ! = NumRng
| NumRng < NumRng
| NumRng > NumRng
| NumRng <= NumRng
| NumRng >= NumRng

BoolExp1 = [’!’] BoolExp0

BoolExp0 = ’True’
| ’False’
| ’(’ BoolExp2’)’
```

Grammar Transformation

```
NumRng = NumFld '%' NumRng
| NumFld '/' NumRng
| NumFld '*' NumRng
| NumFld

NumFld = NumSng '+' NumFld
| NumSng '-' NumFld
| NumSng

NumSng = int
| ' ′ ' char ' ′ '
| '(' NumRng ')'
```

Difficulties

- Input "4 -5" is recognized a two digits, resulting in error

Extensions

- Higher level tokens for expressions

Questions ?