

# Interpreter & Compiler: exercises

Lectures 14-15-16

Formal Languages and Compilers 2011

Nataliia Bielova

# Exercises for compiler

- for each in vectors (test\_foreach.cre)

```
var l: array[5] of int;  
...  
for each i in v do  
  begin  
    i := n;  
    n := n - 1;  
  end;
```

# Intermediate code for FOR

- Node in syntax tree For (i, 1, n, cmd)
- Generated code (gencommand in commands.ml)

```
startfor:  CG      increg      n   tmp      // tmp = (increg > n)
           JNE     tmp         0   endfor // if (increg > n) then jump
           ...                                     // code generated for cmd
           ADD     increg      1    tmp
           CPY     tmp         NULL  increg
           GOTO   startfor    NULL  NULL
endfor:    ...
```

## Intermediate code for FOR EACH

- Node in syntax tree `Foreach (e, v, cmd)`
- Lower bound `start=0`
- Upper bound `finish= dim-1`
- Take a new `increg = start`
- start loop (`increg <= finish`)
- $e \leftarrow v[\text{increg}]$
- generate commands for `cmd`
- $v[\text{increg}] \leftarrow e$
- $\text{increg} \leftarrow \text{increg} + 1$
- end loop

# Exercises for compiler

- interactive input from console (test\_read.cre)

```
var i: int;  
var f: float;  
...  
readInt(i);  
readFloat(f);
```

- change: lexer, parser, syntax tree
- type checking (semantic.ml)
- intermediate code generation (commands.ml)
- + declaration of new intermediate code command (intermediate.ml)
- + target code generation (target.ml)

# Exercises for interpreter

- Lecture 9: code of the interpreter
- Implement **for each** in vectors (test\_foreach.cre)

```
var l: array[5] of int;  
...  
for each i in v do  
  begin  
    i := n;  
    n := n - 1  
  end;
```

# For: semantics

$C \parallel \text{for } i := \text{min to max do } c \parallel_{rs} =$   
 $= C' \parallel \text{for } i := \text{min to max do } c \parallel_{rs}$

where :

$l_1 = \Lambda \parallel i \parallel_{rs}$

$s_0 = \text{updatemem}(s, l_1, \text{min})$

$C' \parallel \text{for } i := \text{min to max do } c \parallel_{rs} =$   
 $= \begin{cases} C' \parallel \text{for each } e \text{ in } v \text{ do } c \parallel_{rs} & \text{if } B \parallel i < \text{max} \parallel = \text{true} \\ s' & \text{otherwise} \end{cases}$

where :

$s' = C \parallel c \parallel_{rs}$

$v = E \parallel i \parallel_{rs}$

$s' = \text{updatemem}(s', l_1, v + 1)$

# For each : semantics

```

C || for each e in v do c ||rs =
= { C || for each e in v do c ||rs  if i ≤ max
  s                                otherwise
local variables : min = lb, max = ub, i = min
where :
s0 = updatemem(s, le, vvi)
le = Λ || e ||rs
vvi = E || v[i] ||rs
s' = C || c ||rs0
s' = updatemem(s', lvi, ve)
lvi = Λ || v[i] ||rs
ve = E || e ||rs

```