INF5110
Obligatorisk Oppgave 1

Andreas Svendsen
SINTEF

19. Februar 2009

Oversikt

- Om meg
- Praktisk informasjon
- Oppgaven
- Språket (D-flat)
- Terminaler
- Metasymboler
- Grammatikken
- JFlex
- CUP
- Oppsummering
- Neste uke

Om meg

- Stipendiat i MoSiS-prosjektet ved SINTEF
  - Veileder er Birger Møller-Pedersen og Øystein Haugen
  - Startet januar 2008
- Mastergrad fra UiO i 2007
  - Utvidet Java med nye begreper (generiske traits) ved hjelp av AntLR
- Tok INF5110 våren 2006

3

Praktisk informasjon

- Obligen er tilgjengelig på kurssiden
- Grupper på opptil tre personer
- Frist
  - 20. mars
- Veiledning per e-post eller avtale
  - E-post: Andreas.Svendsen@sintef.no
  - TLF: 22 06 76 78

4

Oppgaven

- Bruke scannings- og parseringsverktøy
  - JFlex
  - CUP
- Skrive om en grammatikk fra én form til en annen
  - Utvidet BNF -> BNF som passer verktøyet
- Kontrollere presedens og assosiativitet på to måter:
  - Entydig grammatikk
  - Funksjonalitet i CUP
- Bygge abstrakt syntaks tre (AST)
- Skrive det abstrakte syntakstreet til et gitt format
- Rapport som beskriver løsningen

5

Språket (D-flat)

```
struct Complex
{
   var float Real;
   var float Imag;
}

func ret Complex Add( Complex a, Complex b )
{
   var Complex retval;
   retval := new Complex();
   retval.Real := a.Real + b.Real;
   retval.Imag := a.Imag + b.Imag;

   return retval;
}

func Main()
{
   func ret float Square( float val )
   {
      return val ** 2.0;
   }
   var float num;

   num := 6.480740;
   print_float( num );
   print_str( " squared is " );
   print_float( Square( num ) );
   return;
}
```

Strukt

Variabel

Funksjon

Opprette ny instans av strukten Complex

Aksessere variable i strukter (dot-notasjon)

Nestede blokker

Biblioteksfunksjoner

Merk:
Deklarasjoner
før statements.

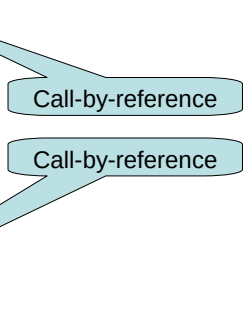
6

Språket (D-flat) (2)

```
func Swap( ref int a, ref int b )
{
   var int tmp;
   tmp := a;
   a := b;
   b := tmp;
}

func Main()
{
   var int x;
   var int y;

   x := 65;
   y := 12;
   Swap(ref x, ref y);
   return;
}
```



Terminaler

- Terminaler uten verdi:
  - IF, THEN, ELSE, WHILE, DO, RETURN, etc.
- Terminaler med verdi:
  - NAME
    - Starte med bokstav, og deretter en sekvens av bokstaver, siffer og underscore.
  - INT_LITERAL
    - Ett eller flere siffer
  - FLOAT_LITERAL
    - Ett eller flere siffer, fulgt av punktum, fulgt av ett eller flere siffer.
  - STRING_LITERAL
    - Tekststreng mellom to anførselstegn (”) uten linjeskift.

Metasymboler

- {…}
  - Gjentakelse null eller flere ganger (regex: *)
- […]
  - Er enten med eller ikke (regex: ?)
- Må skrives om til vanlig BNF

9

Grammatikken

```
PROGRAM      -> DECL { DECL }

DECL         -> VAR_DECL | FUNC_DECL | STRUCT_DECL

VAR_DECL     -> "var" TYPE NAME ";"

FUNC_DECL    -> "func" [ "ret" TYPE ] NAME "("
                        [ PARAM_DECL { "," PARAM_DECL } ] ")"
                        "{" { DECL } { STMT } "}"

STRUCT_DECL  -> "struct" NAME "{" { VAR_DECL } "}"
```

Én eller flere deklarasjoner

Statements kun i funksjoner

10

Grammatikken (2)

```
EXP   -> EXP LOG_OP EXP
       | "!" EXP
       | EXP REL_OP EXP
       | EXP ARIT_OP EXP
       | "(" EXP ")"
       | LITERAL
       | CALL_STMT
       | "new" NAME "(" ")"
       | VAR

VAR   -> NAME | EXP "." NAME
```

Hva med presedens og assosiativitet?

Støtte for dot-notasjon

11

Grammatikken (3)

```
STMT          -> ASSIGN_STMT ";"
                 | IF_STMT
                 | WHILE_STMT
                 | RETURN_STMT ";"
                 | CALL_STMT ";"

ASSIGN_STMT   -> VAR ":=" EXP

IF_STMT       -> "if" EXP "then" "{" { STMT } "}"
                    [ "else" "{" { STMT } "}" ]
```

12

JFlex

```
package oblig1parser;
import java_cup.runtime.*;

%%

%class Lexer
%unicode
%cup
%line
%column
%public

%{
%}

WhiteSpace          = [ \t\f]

%%

{WhiteSpace} { }
"struct"        { return new Symbol(sym.STRUCT); }
"{"             { return new Symbol(sym.LBRACK); }
"}"             { return new Symbol(sym.RBRACK); }

.               { throw new Error("Illegal character '" + yytext() + "' at line " + yyline + ",
                        column " + yycolumn + "."); }
```

Skilletegn

Kompatibilitet med CUP

Makro-definisjoner

Returnerer et nytt Symbol-objekt

Brukerkode

Opsjoner og deklarasjoner

Leksikalske regler og operasjoner

CUP

```
package oblig1parser;
import java_cup.runtime.*;
import syntaxtree.*;
import java.util.*;

parser code {:
:};

/* Terminals */
terminal    STRUCT;
terminal    LBRACK, RBRACK;

/* Non terminals */
non terminal            Program    program, struct_decl;

/* The grammar */
program     ::= struct_decl:sd {: RESULT=sd; :}
            ;
struct_decl ::= STRUCT LBRACK RBRACK {: RESULT=new Program(); :}
            ;
```

Typen til resultatet av produksjonen

Labels

Actions

Brukerkode

Deklarasjoner

Grammatikken

Oppsummering

- Lage to grammatikker og løse skift/reduser-konflikter
- Bygge opp syntakstre ved hjelp av parserverktøyet
- Skrive ut syntakstreet

15

Neste uke

- Mer detaljert gjennomgang av JFlex og CUP
- Les gjennom oppgaveteksten og se på eksempel-filene
- Send spørsmål så snart noe er uklart

16