Hertentamen Vertalerbouw—13 april 2006

De nagekeken tentamens zijn af te halen op het onderwijsbureau.

Opmerkingen:

- Schrijf **netjes** en duidelijk, met zwarte of blauwe pen.
- Zet op het eerste blad alle gegevens als naam, etc., en het totaal aantal ingeleverde bladen, en nummer de ingeleverde bladen.
- Lees de opgaven eerst goed door.
- Motiveer uw antwoorden.
- Wilt u 7 EC (oude stijl) halen zet dat dan boven het tentamen! U dient dan ook som 2 over attributen-grammatica's te maken.

1. (45 minuten)

   a) Geef voor alle nonterminals uit onderstaande produkties de sets *first* en *follow*.

   b) Is de grammatica, gegeven door de volgende produkties met startsymbool A, $LL(1), LR(0)$, $SLR(1)$, $LR(1)$?
   Geef in geval van conflicten deze duidelijk aan. Geef, ingeval de conflicten volgens U oplosbaar zijn, aan hoe de oplossing verloopt.

   $$\begin{array}{lll} & A & \rightarrow & Ba \\ , & B & \rightarrow & Bb \\ , & B & \rightarrow & C \\ , & C & \rightarrow & Cc \\ , & C & \rightarrow & D \\ , & D & \rightarrow & A \\ , & D & \rightarrow & \end{array}$$

2. (45 minuten, **oude stijl!!**)

   Gegeven is de volgende grammatica voor lambda–expressies:

   $$\begin{array}{lll} & LE & \rightarrow & lambdasym\ Idlst\ Body\ lpar\ Arglst\ rpar \\ , & Idlst & \rightarrow & ident\ Idtl \\ , & Idtl & \rightarrow & dotsym\ Idlst\ \mid \\ , & Body & \rightarrow & lpar\ Exp\ rpar \\ , & Arglst & \rightarrow & intdenot\ Argtl \\ , & Argtl & \rightarrow & Arglst\ \mid \\ , & Exp & \rightarrow & Fac\ Exptl \\ , & Exptl & \rightarrow & plussym\ Exp\ \mid \\ , & Fac & \rightarrow & ident\ \mid\ intdenot\ \mid\ Body \end{array}$$

   Een voorbeeld volgens deze grammatica is:

   $$\lambda x.y(x + (y + 33 + x) + 2)(3\ 7)$$

Voorzie deze syntaxregels van rekenvoorschriften en attributen, zodanig dat de volgende checks uitgevoerd worden:

- er zijn evenveel argumenten als identifiers
- in de body komen uitsluitend identifiers voor die achter het lambda-teken staan genoemd

Declareer alle benodigde hulpdatastructuren en geef van attributen duidelijk aan of ze *synthesized* danwel *inherited* zijn. De terminal `ident` heeft een synthesized attribuut `symid`, dat een unieke identificatie van de identifier vormt.

3. (50 minuten)

Gegeven is het volgende Pascal(-achtige) programma:

```
PROGRAM tent (input,output);

VAR a: integer;
    i: integer;

PROCEDURE p (VAR a: integer; b: integer);
VAR i: integer;

   PROCEDURE q (i,j: integer);
   BEGIN IF (i < 2) THEN
            a := b + i;                    (* 1 *)
         ELSE
            b := a * j                     (* 2 *)
   END;

BEGIN i := 1;
      q (i , 5);                           (* 3 *)
END (* p *)

FUNCTION r (): integer;
   VAR n: integer;
BEGIN n := 2;
      p (n , 0);                           (* 4 *)
      return n;
END; (* r *)

BEGIN ...
      i := r ();                           (* 5 *)
      ...
END (* tent *).
```

Voor het geheugenbeheer worden de volgende registers gebruikt:

gp het base address van het activation record van het hoofdprogramma

lnb het base address van het huidige activation record

lfa het adres van de eerste vrije stack locatie

(We gaan ervan uit dat het return adres op een aparte stack wordt bewaard, zodat U daarmee geen rekening hoeft te houden.)

Voor het overdragen van de omgeving van een aan te roepen procedure kan het register **env** worden gebruikt. Verder zijn er voldoende registers (R0, R1, . .) voor het opslaan van tussenresultaten. Merk op dat het (Pascal) keyword `VAR` betekent dat het argument *by reference* wordt doorgegeven.

- Teken de AR (activation record) voor procedure `p`;
- Geef de te genereren (pseudo-)instructies voor de **entry** en **exit** van `r`;
- Geef de te genereren (pseudo-)instructies voor de vijf genummerde regels.

4. (40 minuten)

Gegeven is een eenvoudig taaltje, dat syntactisch gespecificeerd wordt door de productieregels P:

$$
\begin{array}{lllll}
P = & \{ & S & \rightarrow & E\ Stl \\
 & , & Stl & \rightarrow & semicolon\ S \\
 & , & Stl & \rightarrow & \\
 & , & E & \rightarrow & T\ Etl \\
 & , & Etl & \rightarrow & addopsym\ E \\
 & , & Etl & \rightarrow & \\
 & , & T & \rightarrow & F\ Ttl \\
 & , & Ttl & \rightarrow & mulopsym\ T \\
 & , & Ttl & \rightarrow & \\
 & , & F & \rightarrow & intdenotation \\
 & , & F & \rightarrow & (S) \\
 & \} & & &
\end{array}
$$

a) Laat zien dat deze grammatica $LL(1)$ is.
b) Voor een recursive descent parser **met** error-recovery hebben we o.a. de volgende procedures nodig: `pStl, pTtl, match en delete`.

Geef de implementatie van deze procedures.