Tentamen

Översättarteknik, 5p
13 januari 1995

- Skrivtid 16–22
- Hjälpmedel: T.W. Parsons: "*Introduction to Compiler Construction*".
  Dessutom 2 sidor om *boolen expressions* och *statements*.
  (Finns ett par exemplar hos skrivvakten.)
- Maxpoäng 44 (för godkänt krävs 22p, väl godkänt 33p)
- Uppgifterna är slumpmässigt ordnade.
- Börja varje uppgift på *nytt* blad och skriv uppgiftsnummer längst ut till höger på bladet.
- Skriv namn på *varje* blad.
- Tentamensgenomgång (+ återlämning) sker onsdagen den 25 januari kl 8:15 i MC313
- Resultat kommer upp på anslagstavlan så snart jag rättat klart.
  (Kommer även att finnas tillgängligt på WWW.)
- Fråga om någonting är oklart!

  Lycka Till!

Uppgift 1 (3 + 2 p)

(a) Beskriv kortfattat tre olika faser hos en kompilator, vad som utförs under var och en av dem.

(b) En kompilator kan bestå av ett eller flera **pass**. Vad menas med detta? Nämn en för- och en nackdel med en enpasskompilator.

Uppgift 2 (3 + 3 p)

Använd tekniken med listor av kvadrupelnummer och *backpatching* för att skriva ett översättningsschema för följande produktioner. Uttrycken ska ha två attribut, *truelist* och *falselist*, medan satser bara ska ha attributet *nextlist*. Använd markörer om så behövs.

(a) $\mathrm{E} \rightarrow \mathrm{E}_1$ **nor** E_2
Uttrycket E_1 **nor** E_2 är sant om både E_1 och E_2 är falska, annars falskt.

(b) $\mathrm{S} \rightarrow$ **repeat** S_1 **until** E

Uppgift 3 (5 p)

I språket LPL finns de binära operatorerna $\times$ och $\circ$ som båda är högerassociativa. $\times$ har högre prioritet. Ett uttryck i LPL kan bestå av en **id**, eller någon av de ovan nämnda operatorerna applicerade på två uttryck (i vanlig infix notation). Konstruera en kontextfri grammatik för uttryck i LPL, som också uttrycker operatorernas associativitet och prioritet.

Uppgift 4 (2 + 2 + 2 + 2 p)

Givet följande grammatik för listor, som är inte LL(1).

S → (L)
S → **a**
L → L , S
L → S

(a) Skriv om grammatiken så att den blir LL(1). Förklara vilka omskrivningar du gör.

(b) Konstruera FIRST– och FOLLOW–mängderna för den omskrivna grammatikens icketerminaler.

(c) Konstruera en LL(1)–parser för den omskrivna grammatiken.

(d) Visa hur din LL–parser analyserar strängen: ((**a**))

Uppgift 5 (6 + 2 p)

Använd den *givna* grammatiken för listor från föregående uppgift även till denna uppgift.

(a) Konstruera en SLR(1)–parser för grammatiken. (Lämna in både tabell och "goto"–mängderna/funktionerna.) Om du stöter på konflikter, välj någon tolkning och kommentera denna.

(b) Visa hur din SLR–parser analyserar strängen: **(a , (a))**

Uppgift 6 (2 + 2 p)

(a) Nämn 2 olika "krav" man bör ställa på optimeringsdelen hos en kompilator. Förklara även kortfattat varför.

(b) Ge en kort motivering till varför man *inte* skulle vilja ha optimering. (Minst två anledningar.)

Uppgift 7 (3 + 3 + 2 p)

Utgående från följande reguljära uttryck:
(a | b)*a

(a) Konstruera en icke–deterministisk finit automat som accepterar det språk som beskrivs av uttrycket. Använd de metoder som beskrivs i boken, med följande tillägg:

- Ingen övergång får finnas från ett accepterande tillstånd.
- Ingen övergång får finnas till starttillståndet.

(b) Konstruera en ekvivalent deterministisk automat utifrån den icke–deterministiska automaten i (a).

(c) Den icke–deterministiska automaten i (a) innehåller ju onödigt många tillstånd. Skapa en icke–deterministisk automat som består av högst 3 tillstånd som accepterar samma språk. (Ni behöver inte använda några förenklingsregler utan skapa automaten "direkt" från det reguljära uttrycket.)