Kompilatorer och interpretatorer: Hemtentamen 2021-01-05

Det här är hemtentan som går tisdag 5 januari 2021 i kursen DT135G Kompilatorer och interpretatorer, provkod A001. Den gäller även för den gamla kurskoden, DT125G. Ansvarig lärare är Thomas Padron-McCarthy (thomas.padron-mccarthy@oru.se), telefon **070-73 47 013**.

Tid: 08:15 - 13:15

Instruktioner

1. Den här hemtentan ersätter den planerade salstentan.
2. Uppgifterna ska lösas enskilt, dvs inga grupper av två eller flera studenter.
3. **Du får använda dator, böcker och vilka andra hjälpmedel som helst,** men du får inte samarbeta eller fråga någon (utom mig). Exempelvis är det tillåtet att söka och läsa på webbplatser som Stack Overflow, men inte att ställa egna frågor. Du kan alltså provköra kommandon och program, om du vill.
4. **Diskutera inte uppgifterna eller dela med dig av svar förrän tidigast dagen efter tentan.**
5. Lös de angivna uppgifterna och samla svaren på lämpligt sätt, till exempel i en PDF. Skicka sen in lösningarna till mig i Blackboard. (Gå till kursinnehållet, **Course Content**, gå in i mappen **Hemtentor**, och välj den rätta tentan.) Där kan man sedan välja att skicka in sin lösning. Då blir bedömningen anonym.
6. Tentatiden är utökad med en extra timme för att täcka in problem med e-post, inscanning eller fotografering av diagram, och liknande.
7. Skulle något krångla, med Blackboard eller vad det nu kan vara, är det fortfarande möjligt att skicka in lösningarna efter sluttiden, men då behöver man ge någon form av förklaring till det.
8. Om det inte alls går att skicka in lösningarna som avsett i Blackboard, kan man i nödfall skicka in svaren antingen som ett kursmeddelande i Blackboard eller via vanlig e-post (thomas.padron-mccarthy@oru.se), senast vid tentatidens slut. Då blir bedömningen inte anonym.
9. I Blackboard ser du att du skickat in din lösning. Om du i stället skickade in med e-post, och inte senast en timme efter tentatidens slut fått ett svar från mig med en bekräftelse på att du skickat in svaren, bör du kontakta mig, enklast genom att ringa eller SMS:a mig (ifall det är e-posten som inte fungerar). Tänk på att en del mailtjänster (särskilt Microsoft-tjänster som Hotmail.com, Outlook.com och Live.com) ibland kastar bort brev med bilagor, utan att meddela det.
10. Skriv gärna svaren i ett ordbehandlingsprogram. Rita gärna eventuella diagram i ett ritprogram. Det är inte förbjudet att skriva och rita för hand, men då måste text och bilder scannas in eller fotograferas. Det finns scanner-appar till Android och iPhone, till exempel Adobe Scan, som ger bättre resultat än att bara ta vanliga kort med kameran.
11. Om du behöver fråga något, så kontakta gärna mig. Ring eller skicka SMS, för jag kanske inte kommer att sitta vid datorn hela tiden.
12. Oklara och tvetydiga formuleringar kommer att misstolkas. Lösningar som inte går att läsa eller förstå kan naturligtvis inte ge några poäng.

13. Antaganden utöver de som står i uppgifterna måste anges. Gjorda antaganden får inte förändra den givna uppgiften.
14. Skriv gärna förklaringar om hur du tänkt. Även ett svar som är fel kan ge poäng, om det finns med en förklaring som visar att huvudtankarna var rätt.
15. Maximal poäng är 30. För godkänt betyg krävs minst 16 poäng.
16. Resultat meddelas senast 15 arbetsdagar efter tentamensdatum. Eftersom svaren skickas in elektroniskt scannas tentorna inte för retur.

Formelsamling

1. Eliminering av vänsterrekursion

En vänsterrekursiv grammatik kan skrivas om så att den inte är vänsterrekursiv. Antag att en regel (eller, korrektare uttryckt, två produktioner) i grammatiken ser ut så här:

```
A -> A x | y
```

A är en icke-terminal, men **x** och **y** står för godtyckliga konstruktioner som består av terminaler och icke-terminaler.

Regeln ersätts av följande två regler (eller, korrektare uttryckt, tre produktioner), som beskriver samma språk men som inte är vänsterrekursiva:

```
A -> y R
R -> x R | empty
```

2. Vänsterfaktorisering

Antag att grammatiken innehåller denna regel (två produktioner):

```
A -> x y | x z
```

A är en icke-terminal, men **x**, **y** och **z** står för godtyckliga konstruktioner som består av terminaler och icke-terminaler.

Skriv om till dessa tre produktioner:

```
A -> x R
R -> y | z
```

Scenario till uppgifterna

Vi ska skriva ett program där man kan mata in dessa "helghälsningar":

- God Jul!
- God Helg!
- Gott Nytt År!
- Glad Påsk!
- Trevlig Midsommar!

En komplett inmatning består av en eller flera sådana hälsningar, på fritt format, och avslutas med "Klart!". Exempel:

```
God Jul!
God Jul!
 God

   Jul!

Trevlig Midsommar! Glad Påsk! Klart!
```

Uppgift 1 (2 p)

Vilka terminaler behövs för att man ska kunna skriva en grammatik för språket?

Uppgift 2 (4 p)

Skriv en scanner som kan dela upp inmatningen i tokens. Du får själv välja om du ska använda Flex eller skriva den för hand i C, C++ eller ett liknande språk. Scannern ska bestå av en funktion som returnerar en token-kod. Du kan anta att token-koderna finns definierade i form av makron, ungefär som Bison gör när man har en Bison-grammatik med **%token**-deklarationer.

Uppgift 3 (4 p)

Skriv en grammatik för språket. Startsymbolen ska vara *inmatning*, som representerar en fullständig inmatning enligt scenariot ovan.

Uppgift 4 (3 p)

Ett parse-träd (ibland kallat "konkret syntaxträd") innehåller noder för alla icke-terminaler. Rita upp parse-trädet för den här inmatningen, enligt din grammatik i uppgiften ovan:

```
Gott Nytt År!
Gott Nytt År!
Gott Nytt År!
Klart!
```

Uppgift 5 (9 p)

Skriv en prediktiv recursive-descent-parser för språket. i ett språk som åtminstone liknar C, C++, C# eller Java. Du behöver inte skriva exakt korrekt programkod, men det ska framgå vilka procedurer som finns, hur de anropar varandra, och vilka jämförelser med tokentyper som görs.

Vi ska inte bara skriva själva parsern, utan ett komplett program. Programmet ska, när inmatningen är avslutad, skriva ut antalet förekomster av varje hälsningsfras. För exempelinmatningen i scenariot kan utskriften se ut så här:

```
3 God Jul!
0 God Helg!
0 Gott Nytt År!
1 Glad Påsk!
1 Trevlig Midsommar!
```

Låt parsern anropa scanner-funktionen från uppgift 2 ovan. Som i uppgift 2 kan du anta att token-koderna finns definierade. Du kan också anta att det finns en funktion som heter **error**, som man kan anropa när något gått fel, och som skriver ut ett felmeddelande och avslutar programmet.

Uppgift 6 (5 p)

Programmet som vi skrev i uppgiften ovan har en inmatning, som det översätter till en utmatning. På det viset liknar det en kompilator. Beskriv, för var och en av faserna som brukar finnas i en kompilator, om den finns med i programmet och vad den i så fall gör. Om den inte finns med, hur skulle den fasen kunna läggas till, och vad skulle den i så fall göra? Om det skulle vara helt orimligt att ha med den, förklara varför!

Uppgift 7 (3 p)

Här är några olika felaktiga inmatningar:

a) `Glad Jul! Klart!`

b) `God Nationaldag! Klart!`

c) `Klart! Klart! Klart!`

Om vi betraktar programmet från uppgift 5 som en kompilator, vilka faser upptäcks vart och ett av felen i? Motivera svaren!

Thomas Padron-McCarthy (thomas.padron-mccarthy@oru.se), 2 januari 2021