Försättsblad till skriftlig tentamen vid Linköpings Universitet

Datum för tentamen	17/12 - 2010
Sal (1) Om tentan går i flera salar ska du bifoga ett försättsblad till varje sal och ringa in vilken sal som avses	TER4
Tid	14 - 18
Kurskod	TDDB44
Provkod	TDDB44/TEN1
Kursnamn/benämning **Provnamn/benämning**	Kompilatorkonstruktion
Institution	IDA
Antal uppgifter som ingår i tentamen	10
Jour/Kursansvarig Ange vem som besöker salen	Kristian Stavåker kristian.stavaker@liu.se
Telefon under skrivtiden	076-3361782
Besöker salen ca kl.	15.15
Kursadministratör/kontaktperson (namn + tfnr + mailaddress)	Peter Fritzson peter.fritzson@liu.se
Tillåtna hjälpmedel	Ordbok, räknare
Övrigt	Kontors nummer KS: 013-284093
Vilken typ av papper ska användas, rutigt eller linjerat	~~Blanko~~ Rutat
Antal exemplar i påsen	

Linköpings universitet
IDA – Department of Computer and Information Science
Professor Peter Fritzson

TENTAMEN / *EXAM*

TDDB44 Kompilatorkonstruktion / *Compiler Construction*

December 17, 2010, 14:00–18:00, TER4

Jour: Kristian Stavåker, 0763-361782 (Will come approx 15.15)

Hjälpmedel / *Allowed material:*

- Engelsk ordbok / Dictionary from/to English to/from your native language;
- Miniräknare / Pocket calculator

General Instructions

- This exam has 10 assignments and 5 pages, including this one.
- Read all assignments carefully and completely before you begin.
- It is recommended that you use a new sheet for each assignment. Number all your sheets, and mark each sheet on top with your name, personal number/personnummer, and the course code.
- You may answer in either English or Swedish.
- Write clearly. Unreadable text will be ignored.
- Be precise in your statements. Unprecise formulations may lead to a reduction of points.
- Motivate clearly all statements and reasoning.
- Explain calculations and solution procedures.
- The assignments are *not* ordered according to difficulty.
- The exam is designed for 43 points (including 3 points for on time finished laboratory exercises, see last assignment). You may thus plan about 6 minutes per point.
- Grading: U, 3, 4, 5.
- For exchange students (with a .in the personnummer) ECTS marks will be applied.
- The preliminary threshold for passing (grade 3) is 20 points.

1. (3p) Compiler Structure and Generators

(a) What are the advantages and disadvantages of a multi-pass compiler (compared to a one-pass compiler)? (1p)

(b) Most modern compilers have not just one but several intermediate representations.

i. Explain how these are, in general, related to each other, and what this organization means for the code generation process. (1p)

ii. What is the main advantage of having more than one IR, and what could be a drawback? (1p)

2. (6p) Top-Down Parsing

Given a grammar with nonterminals `<S>`, `<P>`, `<Q>`, `<R>` where `<S>` is the start symbol and the following productions:

```
1. <S> ::= <P> ^ <P>
2. <P> ::= <P> ! <Q>
3.       | <Q>
4. <Q> ::= <R> ? <Q>
5.       | <R>
6. <R> ::= m
7.       | n
```

Assume that `^`, `!` and `?` are operators.

(a) What is the associativity (left, right, or none) of the operators? (1p)

(b) What is the precedence (relative priority) between the operators? (1p)

(c) Can the given grammar be used directly for a recursive-descent parser? Motivate your answer. If not, rewrite the grammar so that the language generated from it can be analyzed using the recursive-desent method. (2p)

(d) Write a recursive-descent parser to analyze the language defined by the grammar. (2p)

3. (6p) LR Parsing

Given the following grammar G for strings over the alphabet `{a,b,c}` with nonterminals `S` and `A` where `S` is the start symbol:

```
S ::= A
A ::= aAa | bAb | c
```

Is the grammar G in SLR(1)? Is it LR(0)? Motivate with the LR-item sets.

Construct the characteristic LR-item NFA, the corresponding GOTO graph, the ACTION table and the GOTO table.

Show with tables and stack how the string:

`abcba`

is parsed.

4. (3p) Error Handling

Explain, define, and give examples of using the following concepts regarding error handling:

(a) Valid prefix property. (1p)

(b) Phrase level recovery. (1p)

(c) Global correction. (1p)

5. (3p) Symbol Table Management

The C language allows static nesting of scopes for identifiers determined by blocks enclosed in braces.

Given the following C program:

```
int data1;
int main( void )
{
  int data2, data1;
  // ... some statements omitted
  if (data2!=0) {
    int data3,data1;
    // ... some statements omitted
    for (data3=0; data3<100; data3++) {
      double data2,data3;
      // ... some statements omitted
      data2 = data1 * 2;
    }
  }
}
```

For the program point containing the assignment `data2 = data1 * 2`, show how the program variables are stored in the symbol table if the symbol table is to be realized as a hash table with chaining and block scope control. Assume that your hash function yields value 3 for `data3`, value 1 for `data2` and `data1`, and value 2 for `main`. (2p)

Show and explain how the right entry of the symbol table will be accessed when looking up identifier `data1` in the assignment `data2 = data1 * 2`. (0.5p)

When generating code for a block, one needs to allocate run-time space for all variables defined in the block. Given a hash table with chaining and block scope control as above, show how to enumerate all variables defined in the current block, without searching through the entire table. (0.5p)

6. (5p) Syntax-Directed Translation

The DO-WHILE statement in C++ and Java could be described using this rule:

```
<dowhilestmt> ::= DO <stmt-list> WHILE <expr>;
```

The semantics of the DO-WHILE statement is that `<stmt-list>` is executed and then repeated as long as expression `<expr>` evaluates to true.

Write the semantic rules - a syntax directed translation scheme – for translating the DO-WHILE statement to quadruples. Assume that the translation scheme is to be used in a bottom-up parsing environment using a semantic stack. Use the grammar rule above as a starting point, but maybe it has to be changed.

In order to obtain full points, you are not allowed to define and use symbolic labels, i.e., all jumps should have absolute quadruple addresses as their destinations. Explain all the attributes, functions, and instructions that you introduce. State all your assumptions.

(For an otherwise correct solution that uses symbolic labels up to 3p can be given.)

7. (6p) Intermediate Code Generation and Optimization

(a) Given the following code segment:

```
for var1:=1 to 20 do
if k1+12>var1
then var2:=var2*3
else var3:=var3-var1;
```

Translate the code segment into abstract syntax trees, quadruples, and postfix code. (3p)

(b) Divide the following code segment into basic blocks, draw a control flow graph, and show as well as motivate the existing loops: (3p)

```
goto L2
L1: p1:=p1+1
L2: p1:=p1-1
L3: p1:=p1+1
if p1=1 then goto L1
if p1=2 then goto L5
p1:=p1+2
L4: p1:=p1+1
if p1=3 then goto L3
L5: p1:=p1+1
if p1=4 then goto L4
```

8. (2p) Memory management

What property of programming languages requires the static link in the procedure's activation record? What is the purpose of the static link, i.e., how and when is it used? What is a display and how is it used)?

9. (6p) Code Generation for RISC, etc.

(a) Given the following medium-level intermediate representation of a program fragment (derived from a for-loop):

```
1: c = 3;
2: k = 20;
3: if k<=0 goto 9;
4: a = c / 2;
5: b = a + c;
6: c = a * b;
7: k = k - 1;
8: goto 3;
9: d = b * c
```

Identify the live ranges of program variables, and draw the live range interference graph for the entire code fragment. Assign registers to all live ranges by coloring the live range interference graph. How many registers do you need at least, and why? (4p)

(b) Register allocation and instruction scheduling are often performed separately (in different phases). Explain the advantages and problems of this separation. (1p)

(c) Explain briefly the concept of software pipelining. Show it with a simple example. (1p)

10. (3p) Compiler Laboratory Exercises

Correct and complete laboratory exercises from the 2010 TDDB44 laboratory course handed in at the latest Dec 15, 2010, will give 3 points. State if you think that you have fulfilled these conditions and you should receive these points.

Good luck!