Faculteit der Exacte Wetenschappen	Tentamen Compilerbouw
Vrije Universiteit	20-12-2004

Questions can be answered in Dutch or English.

1. General knowledge: Explain the following terms:

a. symbol table

b. attribute grammar

c. cloning/specialization of a routine

2. Lexical analysis: A lexical analyser is constructed to recognise two patterns, *a* and *a*b*. It is given the input *aaa$* in which *$* signals the end of the input.

The lexical analyser will have to read to the end of the input to see that the input does not match the pattern *a*b*. How can it still yield the first *a* of the input as the first recognised token?

3. LR parsing: Construct the $LR(0)$ automaton for the grammar

$$S \rightarrow x\,x\,S \mid a$$

where x and a are terminal symbols.

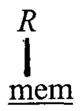
4. Context handling: Suppose dataflow equations are used to track the initialization status of a variable *x*.

a. What information should be recorded for *x* between each node pair?

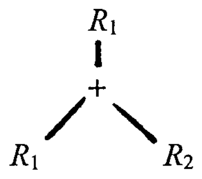
b. Give the KILL and GEN sets for a node containing $x :=$ expression.

5. Code generation: Given a machine with 3 machine instructions:

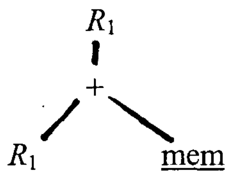
(1) R := mem



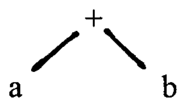
(2) $R_1 := R_1 + R_2$



(3) $R_1 := R_1 +$ mem



where mem denotes a memory location, and given the input tree



where a and b are memory locations. The instructions and the tree are presented to a bottom-up tree-rewriting code generator (BURS code generator).

a. Show the sets the BURS code generator builds at the nodes of the imput tree, and explain why is does so.

b. Show the tree or trees that result from the rewriting process.

6. Code post-processing, peephole optimization:

a. What is a replacement pattern?

b. How are the left-hand sides of replacement patterns found efficiently in the input stream?

7. Garbage collection:

a. Explain how two-space copying garbage collection works.

b. Name an advantage and a disadvantage of this method.

8. Routines: How is partial parametrization ("currying") implemented?

9. Logic programs: In the Prolog rule

grandparent (X, Z) :– parent (X, Y), parent (Y, Z).

the goal *parent (X, Y)* may match for more than one *Y*. How are these multiple values transferred to the second goal *parent (Y, Z)* ?

Assessment:

	1:		2:		3:		4:		5:		6:		7:		8:		9:		
a:	4		8		10		5		7		2		10		10		8		
b:	4						5		5		5		3						
c:	4																		
d:																			
	12	+	8	+	10	+	10	+	12	+	7	+	13	+	10	+	8	=	90