

apl. Prof. Dr. Thomas Noll, Dipl.-Inform. Friedrich Gretz, Souymodip Chakraborty, M.Sc.

First Name: _____

Second Name: _____

Matriculation Number: _____

Degree Programme (please mark):

- ☐ CS Bachelor
- ☐ CS Master
- ☐ CS Lehramt
- ☐ SSE Master
- ☐ Other: _____

	Σ Points	Points obtained
Exercise 1	10	
Exercise 2	10	
Exercise 3	10	
Exercise 4	10	
Exercise 5	10	
Exercise 6	10	
Σ	60	

- Mark every sheet with your **matriculation number**.
- Check that your copy of the exam consists of **8 sheets**.
- Duration of exam: **120 minutes**.
- No helping materials (e.g. books, notes, slides) are permitted.
- Give your solution on the respective sheet. Also use the backside if necessary. If you need more paper, ask the assistants.
- Write with blue or black ink; do **not** use a pencil or red ink.
- Make sure all electronic devices are switched off and are nowhere near you.
- Any attempt at deception leads to failure for this exam, even if it is detected only later.

Question 1 (Multiple Choice):**(10 Points)**

1. Assume that the SLR(1) parser for a grammar G has n_1 states and the LALR(1) parser has n_2 states (where the number of states refers the number of rows in the parsing table):

- ☐ n_1 is necessarily less than n_2 .
☐ n_1 is necessarily equal to n_2 .
☐ n_1 is necessarily greater than n_2 .
☐ None of the above.

2. Given the following expression grammar:

$$E \rightarrow E * F \mid F + E \mid F \qquad F \rightarrow F - F \mid \text{id}$$

which of the following is true:

- ☐ $*$ has higher precedence than $+$.
☐ $-$ has higher precedence than $*$.
☐ $+$ and $-$ have the same precedence.
☐ $+$ has higher precedence than $*$.

3. Which of the following methods generally suffice to convert an arbitrary CFG to an LL(1) grammar?

- ☐ Removing left recursion.
☐ Left factoring.
☐ The combination of both of the above.
☐ None of the above.

4. Let H be any subset of natural numbers, possibly infinite. Define $L_H = \{a^i \mid i \in H\}$. Which of the following claims is generally true:

- ☐ L_H^* is regular.
☐ L_H^* is context free but not regular.
☐ L_H^* is not context free.
☐ None of the above.

5. Which one is correct?

- ☐ Given a CFG G there exists an algorithm to decide whether G is unambiguous.
☐ Every LL(k) grammar is also an LR(k) grammar.
☐ A CFL with prefix-free property always has a LR(0) grammar.
☐ The class of languages captured by LR(k) grammars is strictly larger than the class of languages captured by LR(k-1) for $k > 1$.

6. Grammar

$$S \rightarrow S S \mid (S) \mid [] \mid t$$

is:

- ☐ not a LL(1) grammar since it can be left factorized.
☐ not a LL(1) grammar as it is ambiguous.
☐ not a LL(1) grammar as it has left recursion.
☐ none of the above.

7. Let the program be in the state (l, d, p) , where $l = 10$ is the program counter, d is the data stack and $p = 5 : 4 : 7 : 4 : 5 : 4 : 3 : 5 : 9 : 8 : \dots$ is the procedure stack. What is the resulting state of the program if the next instruction is:

10 : call (20,2,1)

- ☐ (20, d, 4 : 3 : 10 : 0 : p).
- ☐ (20, d, 13 : 3 : 11 : 0 : p).
- ☐ (12, d, 13 : 3 : 20 : 0 : p).
- ☐ None of the above

8. Grammar

$$S' \rightarrow S \quad S \rightarrow A \mid b \quad A \rightarrow aA \mid c$$

is:

- ☐ LR(0) and hence SLR(1) and LR(1).
- ☐ not LR(0) but SLR(1) and hence LR(1).
- ☐ not LR(0) and not SLR but LR(1).
- ☐ none of the above.

9. Which of the following derivations does a top-down parser use while parsing an input string? The input is assumed to scan left to right.

- ☐ Left derivation.
- ☐ Leftmost derivation traced out in reverse.
- ☐ Rightmost derivation.
- ☐ Rightmost derivation traced in reverse.

10. Functions $f_1()$ and $f_2()$ are *declared* inside of a function $g()$. If $f_2()$ was called from $f_1()$ and $f_1()$ was called from $g()$ then:

- ☐ The *static link* of f_2 will point to f_1 in the procedure stack.
- ☐ f_1 cannot call f_2 as f_2 is not declared in f_1 .
- ☐ The *static link* of f_2 points to g in the procedure stack.
- ☐ The *dynamic link* of f_2 points to g in the procedure stack.

Question 2 (Lexical Analysis):**(10 Points)**

Let the alphabet Σ be given by $A \cup \{\backslash s\}$ where $A = \{a, b, \dots, z\}$ and $\backslash s$ is the space symbol. The language L is composed of *texts* with the following property:

- A text is a sequence of words that are separated exactly by one space.
- Texts must not start or end with space.
- Each word is a (non-empty) string of letters.
- Proper usage of article: words starting with a consonant can only have article 'a' and words starting with a vowel can only have article 'an'.

Define L by a regular expression.

Question 3 (LL(1) Grammar):**(10 Points)**

For the grammar given below,

$$\begin{aligned} S &\rightarrow a A b B \mid b A a B \mid \epsilon \\ A &\rightarrow S \\ B &\rightarrow S \end{aligned}$$

ϵ is the empty string.

- a)** Construct the LL(1) parsing table by computing the lookahead sets.
- b)** Is the grammar LL(1)?

Question 4 (LR(1) Grammar):**(10 Points)**

Consider the following $LR(1)$ grammar.

$$\begin{array}{lll} S' & \rightarrow & S \quad (0) \\ S & \rightarrow & C C \quad (1) \\ C & \rightarrow & c C \mid d \quad (2,3) \end{array}$$

- a) Construct the LR(1) parsing table for the above grammar.
- b) Parse the string $cdcd\$$.

Question 5 (Syntax-Directed Definitions):**(10 Points)**

Consider the following grammar for declaring variables:

$$\begin{aligned} D &\rightarrow T L ; \\ T &\rightarrow \text{int} \mid \text{char} \mid \text{real} \mid \text{array } T [N] \\ N &\rightarrow \text{num} \\ L &\rightarrow \text{id} , L \mid \text{id} \end{aligned}$$

The language of the grammar gives declarations of *types* of (variable) identifiers. For example,

array real [5] x;

defines a variable *x* whose type is *array of real*. Assume that the size of type *int* is 4 bytes, *char* is 2 bytes and *real* is 8 bytes. An *array* of *n* literals of type *T* needs 4 bytes to store the size of the array (book keeping) plus *n* times the size of type *T*. In the above example, the type of variable *x* is *array of real* and the memory needed to store it is $4 + 5 * 8 = 44$ bytes.

- a) Write down the semantic rules (equation system) that defines the type of a variable and memory needed to store it. The latter should be associated as inherited attributes *type* and *size*, respectively, with each occurrence of terminal **id**. Here you can assume that the numeric value of terminal **num** is provided by the scanner in attribute *value*.
- b) Draw the attributed tree for the following string: array array char [4] [2] a, b;

Question 6 (Code Generation):**(10 Points)**

Write a program in intermediate code that outputs the square values of the first n natural numbers, starting with 1. Here you can assume the following:

- The value of parameter n is provided at offset 1 in the I/O frame.
- Offset 2 can be used for handling intermediate values.
- To print a number, you can employ instruction `WRITE` which outputs (and removes) the top of the data stack.