# FLCD Seminar 1 – Programming Languages' Specification

## **<u>Notations (meta-languages)</u>**

### I.BNF (Backus-Naur Form)

Constructs:

1. Meta-linguistic variables (non-terminals) - written between < >
2. Language primitives (terminals) - written as they are, no special delimiters
3. Meta-linguistic connectors
   a. ::= equals by definition
   b. | alternative (OR)

General shape of a BNF definition:

<construct> ::= expr_1 | expr_2 | ... | expr_n, where expr_i is a combinaton of terminals and/or nonterminals, i=1,n

***Ex.1***: Specify, using BNF, all nonempty sequences of letters

<let_sequence> ::= <letter> | <letter><let_sequence>

<letter> ::= a | b | ... | z | A | B | .... | Z

***Ex.2***: Specify, using BNF, both signed and unsigned integers, with the following constraints:

- 0 does not have a sign
- numbers of at least two digits cannot start with 0

```
<integer> ::= 0 | <sign> <unsigned> | <unsigned>
<sign> ::= - | +
<unsigned> ::= <nonzerodigit> | <nonzerodigit> <digit_seq>
<digit_seq> ::= <digit> | <digit> <digit_seq>
<nonzerodigit> ::= 1 | 2 | 3 .. | 9
<digit> ::= 0 | <nonzerodigit>
```

### II.EBNF (Extended BNF)

Wirth's dialect

1. Changes to the concrete syntax of standard BNF

- Nonterminals lose <> => they are written without delimiters
- Terminals are written between " "
- ::= becomes =

2. New constructs

- {}      - repetition 0 or more times
- []      - optionality (0 or 1)
- ()      - math grouping
- (* *)   - comments
- rules end with .

***Ex.3***:  Ex. 2 reloaded, in EBNF

```
integer = "0" | [" + " | " - "] nonzerodigit { "0" | nonzerodigit }
nonzerodigit = "1" |  ...  | "9"
```