COS**341**, 2021

Practical **1: "Lexer"**

SPECIFICATION

Preliminaries

- Throughout this specification, **bold blue** will be used to indicate the **Σ-**symbols *which your lexer software must deal with*.
- Text in normal black font describes and explains the tasks of this assignment.
- Throughout this specification document, the two following symbols will be used for the ***invisible characters*** on the keyboard:
  - □ for the *blank_space* key,
  - # for the *return_enter* key (at the end of a text line).

Students' Programming Language: ***SPL***

- In this project we will deal with ***SPL***, which your professor has "made" for you, for educational purposes.
  - In P1 (this specification) we will deal with the ***lexical analysis*** of ***SPL***.
  - In P2 (later) we will deal with the syntax analysis (grammar and parsing) of ***SPL***.
- On the following slides,
  - The *admissible words* of ***SPL*** will be given,
  - and your assignment task will be stipulated.

Admissible Words (vocabulary)

of ***SPL***

- **Comparison** symbols: **eq** , **<** , **>**
  - Note: **eq** may *not be used as variable name*
- **Boolean** operators: **and** , **or** , **not**
  - Note: may *not be used as variable names*
- **Number** operators: **add** , **sub** , **mult**
  - Note: may *not be used as variable names*
- **String** indicators: **“** , **”**
- **Separator** symbols: □ , **#**
  - Note: see slide 2 for their explanation
- **Grouping** symbols: **(** , **)** , **{** , **}** , **,** , **;**
- **Assignment** operator: **=**

- **Control structure**

  keywords: **if**, **then**, **else**, **while**, **for**
  - Note: may *not be used as variable names*
- **I/O** commands: **input** , **output**
  - Note: may *not be used as variable names*
- Special command: **halt**
  - Note: may *not be used as variable name*
- **Procedure definition** keyword: **proc**
  - Note: may *not be used as variable name*

- **Integer** Numbers in ***SPL*** are characterised by the following regular expression:

$$\boxed{(\mathbf{0} \mid ((D_{pos})\bullet(D_{null})^{*})) \mid (-\bullet((D_{pos})\bullet(D_{null})^{*}))}$$

- Whereby the usual Digits are used:

$$D_{pos} := (\mathbf{1} \mid \mathbf{2} \mid \mathbf{3} \mid \mathbf{4} \mid \mathbf{5} \mid \mathbf{6} \mid \mathbf{7} \mid \mathbf{8} \mid \mathbf{9})$$

$$D_{null} := (\mathbf{0} \mid D_{pos})$$

- **User-defined names** (e.g.: variable names) in ***SPL*** are generally defined by the **regular expression**:

$$\boxed{\mathrm{Lett}_{rom} \bullet (\mathrm{Lett}_{rom} \mid \mathrm{D}_{null})^*}$$

whereby

Lett_{rom} := (**a** | **b** | **c** | **d** | ... | **x** | **y** | **z**)

are the usual small roman letters

- Note, however, that the **special keywords** (defined on the previous slides) **must be excluded** from this general definition**!**

- **Short Strings** in ***SPL*** are defined by the regular expression

$$\text{“} \bullet (\square \mid \text{Lett}_{rom} \mid \text{D}_{null})^{\{0,1,2,3,4,5,6,7,8\}} \bullet \text{”}$$

which indicates that the *string-length* is

minimally 0 (**“”**) and

maximally 8 characters *between* the **“”** marks

Explanation:

{0,1,2,3,4,5,6,7,8} is a bounded subset of the un-bounded regular expression star operator *****

Examples: **“hello□jo”** , **“cos341”** , **“error”** , **“my□house”** , **“□□”**

Your TASK

Implement the Lexer with the "Strategy": LONGEST MATCH

Note: **Software Demo Date will be announced** on the COS341 Web Page

Construction Strategy

- **ON PAPER**:
  - Regular Expressions to NFA
- **ON PAPER**:
  - NFA to DFA
- **ON PAPER**:
  - DFA to Min-DFA
- **IN SOFTWARE** (*any Programming Lang*.):
  - *Implementation* of Min-DFA *embedded* into the Longest-Match-Lexer

Hint: **exploit the space_symbols wisely** for longest matching!

INPUT

- You will be given a plain text file (*test**.txt***) which contains a long and un-structured sequence of text.
  - *For pre-testing your scanner software before the Assessment-Day, use your own home-made **.txt** files as input*
- This input text file must be scanned, one character after another, by the Lexer tool which you must implement

along the lines of what you have learned from the book and in the lectures.

OUTPUT

- IF the input text file contains any word or character that does NOT belong to the vocabulary of ***SPL***,
  - then your tool <u>must</u> output the message: "**Lexical Error:**"
  - and <u>must</u> then indicate the detected input text snippet which caused the error
  - and thereafter <u>abort</u> the process of scanning.

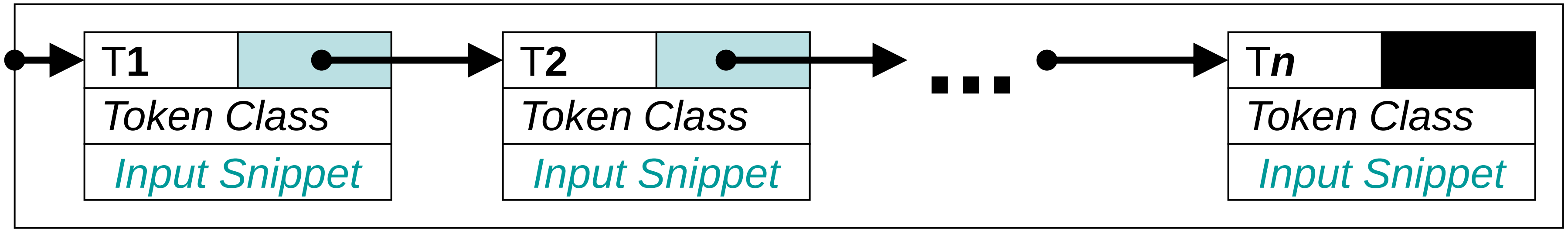
Output Examples:

Lexical Error: @ = *illegal character*. Scanning aborted.

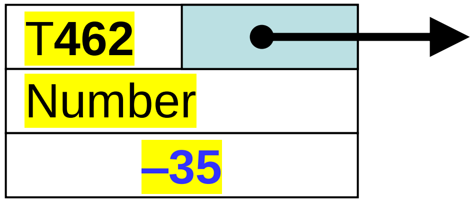
Lexical Error: **"universit** = *string too long*. Scanning aborted.

OUTPUT

- OTHERWISE your tool must create an **output file** which contains a finite LINKED LIST of the following data for the correctly identified **Tokens**:

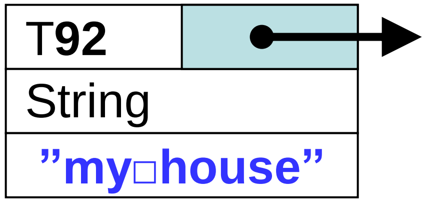


Example:



OUTPUT

- **Special treatment** of the two “**invisible**” input symbols □ and # from the keyboard:
  - For # you do NOT create any data block in the linked list. If you have correctly scanned a # symbol, you simply continue with scanning.
  - For □ :
    - IF □ appears **inside a string**, it will be part of its STRING token data block.
    - IF □ appears **outside a string**, then proceed as in the case of # .



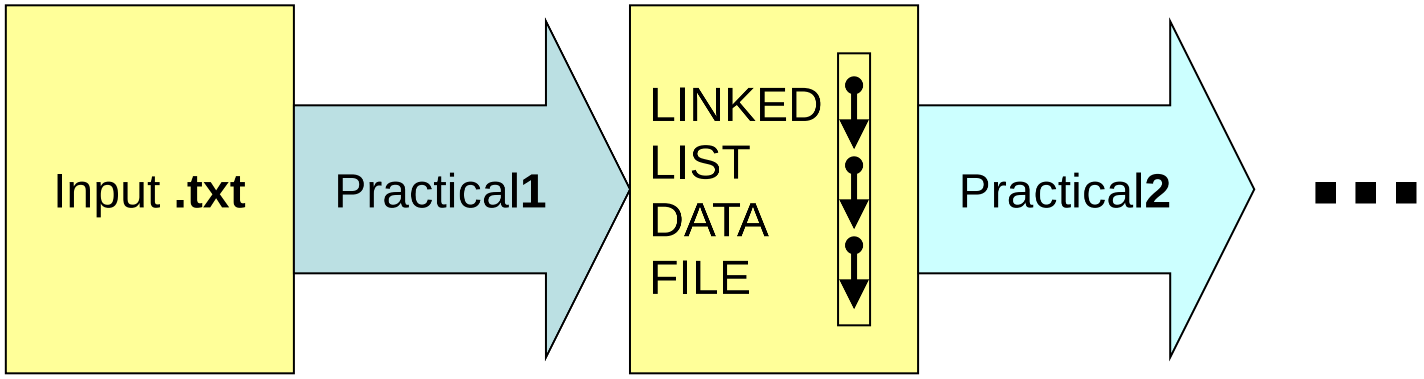
Explanation: Because of the “➔” in the linked list, we do not need the separator symbols any longer

Web-based Presentation

- The output file, which your Lexer has created at server-side, must then be transmitted either as plain *.txt file, or as a very simple *.HTML file, back to the client-side (for viewing), as it was already explained in the previous (preparatory) practical **P0**.
- **Attention: Marks will be given ONLY for what is visible at the Client-Side!**

Additional Remark

- The persistent **output file** (containing the linked list data structure) created by your Lexer in this Practical**1** will later be used **as input** file (for parsing) in Practical2:



And **now**…

HAPPY CODING!

☺

Note: Plagiarism is *forbidden!*
Code sharing with other students
is also *not allowed*