University of Zagreb
Faculty of Electrical Engineering and Computing
Department of Electronics, Microelectronics, Computer and Intelligent Systems

INTRODUCTION TO COMPUTER THEORY

Ak. year 2022/2023

4. laboratory exercise

The task of the 4th laboratory exercise is to programmatically realize a parser using the recursive descent technique for the following grammar:

S → aAB | bBA
A → bC | And
B → ccSbc | ϵ
C → AA

As can be seen from the productions, the grammar has four non-final signs**WITH**,**AND**,**B**and**C**. The initial non-terminating character is **WITH**. Furthermore, the alphabet of grammar consists of signs**And**,**b**and**c**. The default grammar is the LL(1) grammar in an extra simplified form, similar to the textbook example on page 92.
Your program will receive exactly one line containing a string of characters on standard input**And**,**b**and**c**and ends with a newline character (**\n**). The input string will not be longer than 200 characters (not counting the end-of-line character).

The program should parse the string using the recursive descent technique and immediately after entering a function that processes a certain non-terminating character, print that non-terminating character to the standard output. All non-terminating characters are written to the first line of the output in turn. If the string is in the language defined by the default grammar, the program should print the word in a new line (the second line of the output).**THAT**. If the string is not in the language, the word should be printed on a new line**NOT**. When the string is not in the language, the parser should stop processing the string as soon as possible, i.e. exactly when it finds that the string is not in the language. In other words, the printing is unambiguously determined in both cases. The expected output is further explained with several examples at the end of the document.

The time limit on program execution for any input sequence is 10 seconds. A typical implementation for the most complex example will be performed instantaneously, ie in less than 100 milliseconds.

The entry point for Java solutions should be in the Parser class, and the entry point for Python solutions should be in the Parser.py file.

Example 1

Entrance:

ah

Expected output:

SAB
THAT

String parsing**ah**starts with a function call that handles the initial non-terminating character**WITH**. As the first character of the string**And**, the parser unambiguously decides on the application of the production**S → aAB**. The read head moves to the next character**And**and the function that handles the non-terminating character is called**AND**. The parser unequivocally determines that it should

apply production**A → a**and a function that processes the character**AND**moves the read head to the end of the string and finishes working. Finally, the function that processes the character is called**B**. As the read head is positioned at the end of string character, only ϵ-production can be applied**B → ϵ**. Finally, given that the entire string was read by calling the function that processes the initial non-terminating character S, i.e. the reading head is located at the end of the string, the parser concludes that the string is in the language and prints it on a new line**THAT**.

Example 2

Entrance:

ab

Expected output:

SACA
NOT

As in the previous example, the parser calls the character processing functions**WITH**and**AND**. When processing a non-terminating character**AND**, the read head is set to the end character**b**, so the production is applied**A → bC**. In other words, it calls the function that processes the character before**C,**the read head moves to the end of the string. A function that processes a character**C**it does not check anything, but immediately applies the only production**C → AA**, i.e. calls the character function**AND**. As the read head is at the end of the string at that moment, the parser detects that the string is not in the language, finishes processing and prints**NOT**.

Example 3

Entrance:

bccaabcbaa

Expected output:

SBSABACAA
THAT

Example 4

Entrance:

bbaab

Expected output:

SCACAA
NOT