TC3048 Compilers Design

Project Proposal

February 28, 2019

Kotoba

Isabela Escalante Campbell
A01193251

Carolina Galván Villarreal
A01192953

Vision/Purpose

To enhance text analysis worldwide and to be the go-to tool for online literacy statistics by visualizing the results that are obtained.

Main objective

Our objective is to create a language that simplifies the process and improves the scope of text analysis. It will be able to show visually the results of the analysis the program makes. The uses of this program are focused on basic mathematical statistics, text mining and user-made functions for analysis. It will be an easy to use language so that people working in the literacy area can implement it in their projects.

Language requirements

Basic elements

Statements:

- Assignment (=)
- Condition (if, else)
- Cycle (while, do while)
- Read / Write (kread, kprint)

Math expressions:

- Arithmetic (+, -, *, /)
- Logical (AND, OR)
- Relational (<,>,==, !=)

Modules:

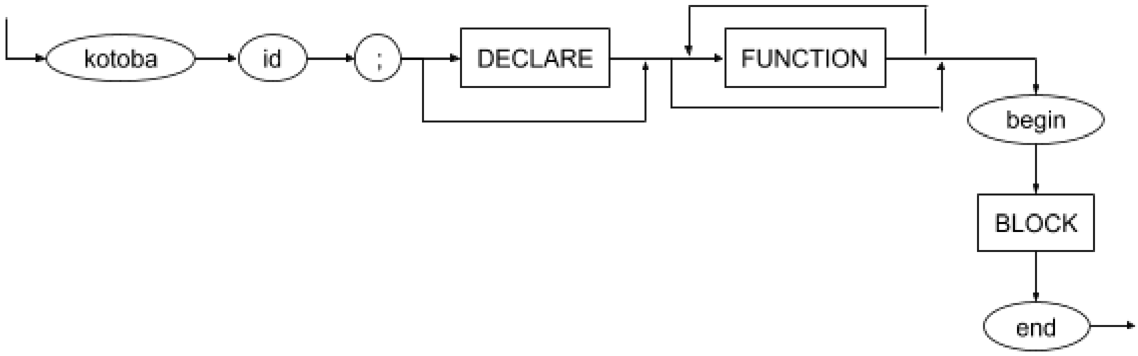
- Length
- Frequency
- Search
- Exists
- Mean
- Median
- Mode
- WordCount
- Tokenize
- Remove

Structured element:

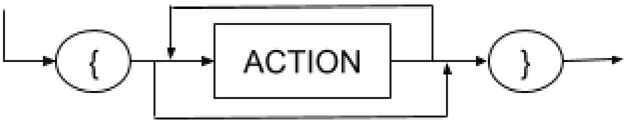
- Arrays

Syntax diagrams

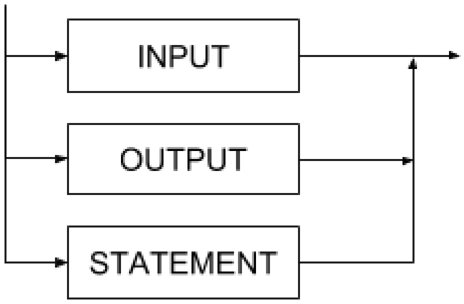
<START>



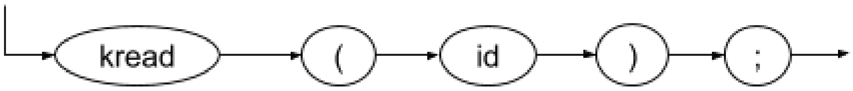
<BLOCK>



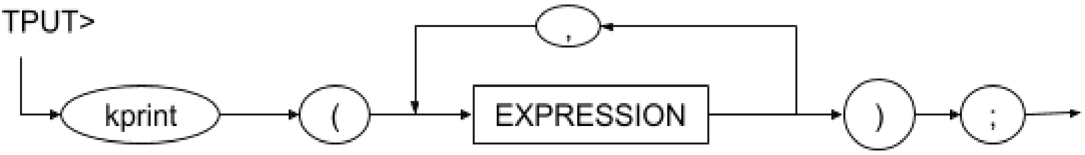
<ACTION>



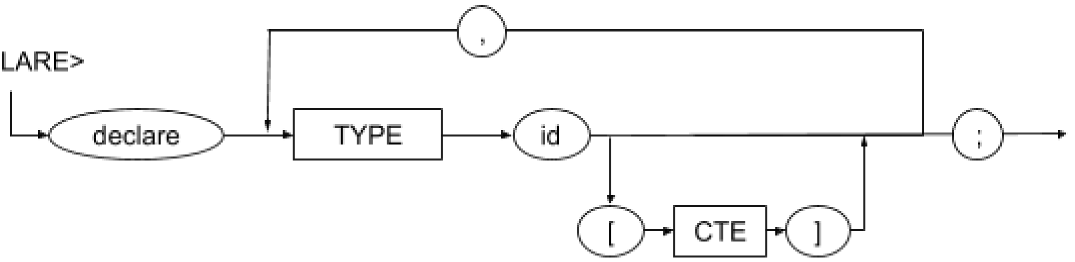
<INPUT>



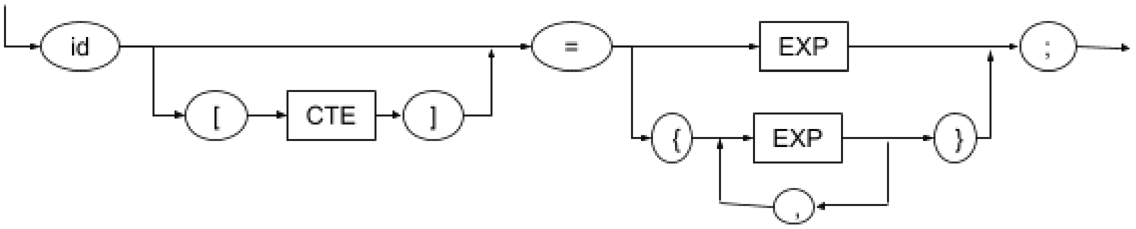
<OUTPUT>



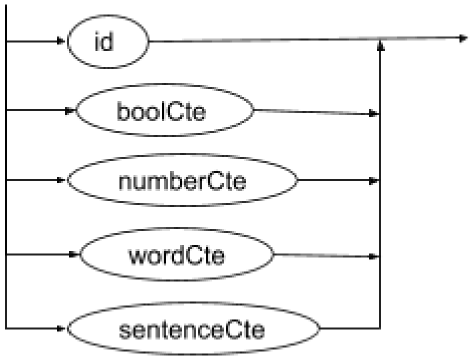
<DECLARE>


declare
TYPE
id
,
[
CTE
]
;

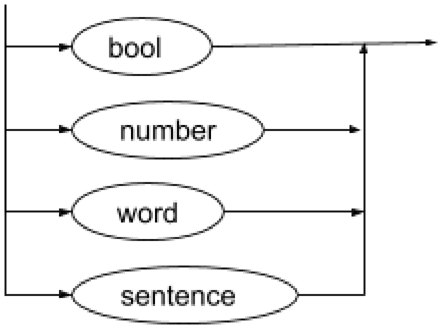
<ASSIGN>


id
[
CTE
]
=
EXP
{
EXP
,
}
;

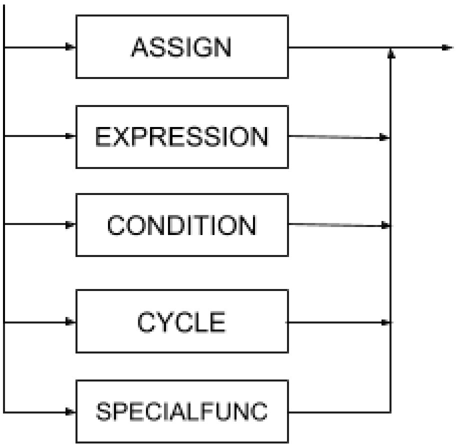
<CTE>


id
boolCte
numberCte
wordCte
sentenceCte

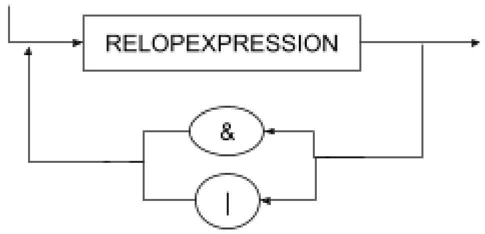
<TYPE>


bool
number
word
sentence

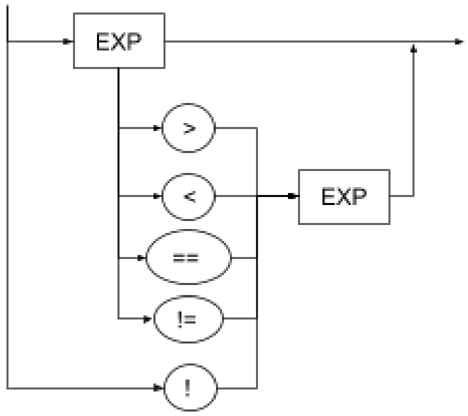
<STATEMENT>


ASSIGN
EXPRESSION
CONDITION
CYCLE
SPECIALFUNC

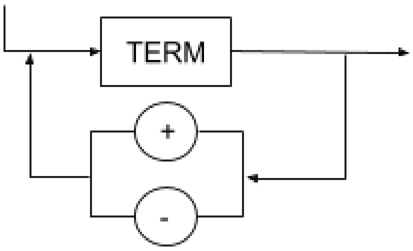
<EXPRESSION>


RELOPEXPRESSION
&
|

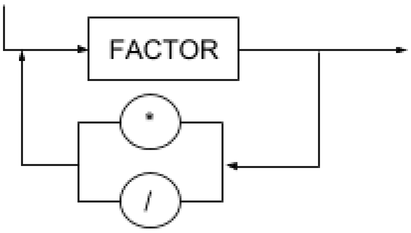
<RELOPEXPRESSION>


EXP
>
<
==
!=
!
EXP

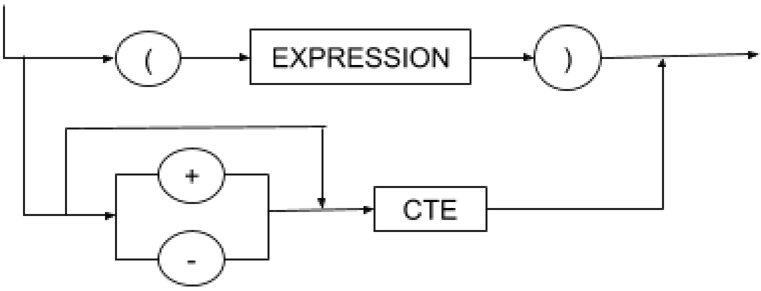
<EXP>


TERM
+
-

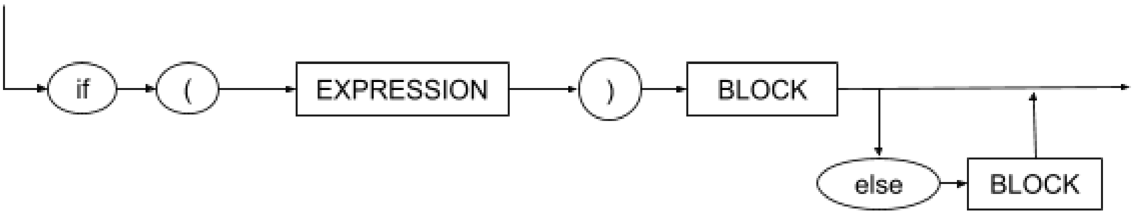
<TERM>


FACTOR
*
/

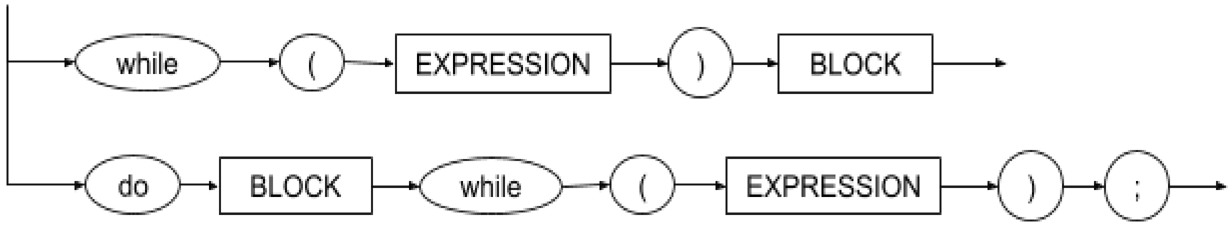
<FACTOR>


(
EXPRESSION
)
+
-
CTE

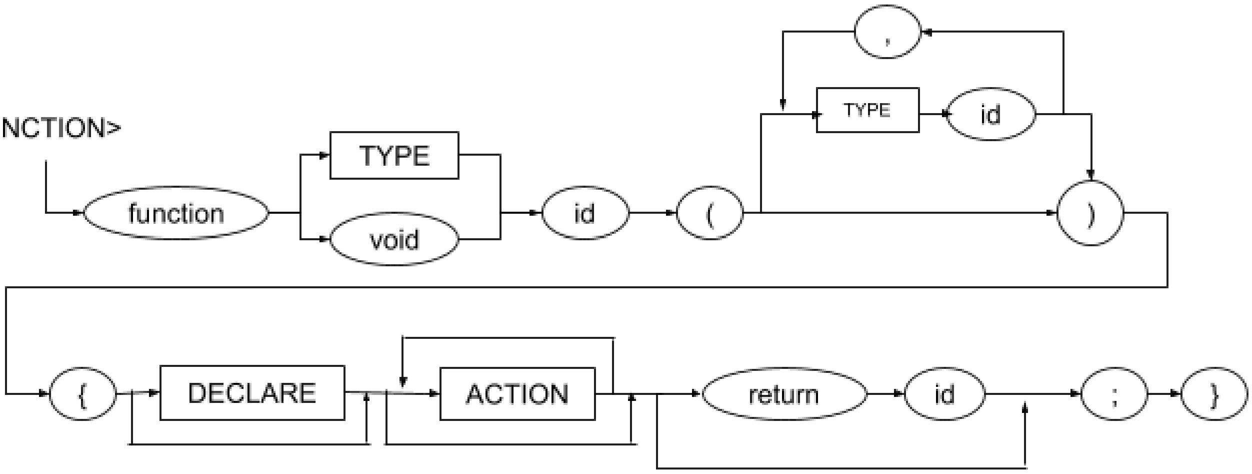
<CONDITION>


if
(
EXPRESSION
)
BLOCK
else
BLOCK

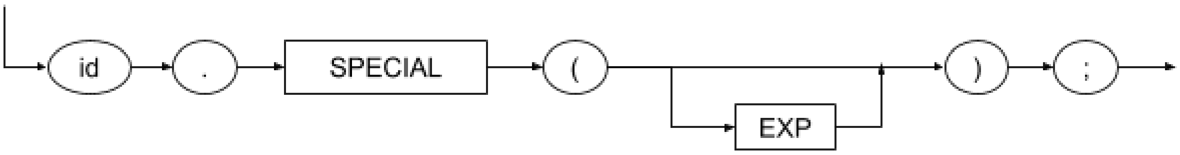
<CYCLE>

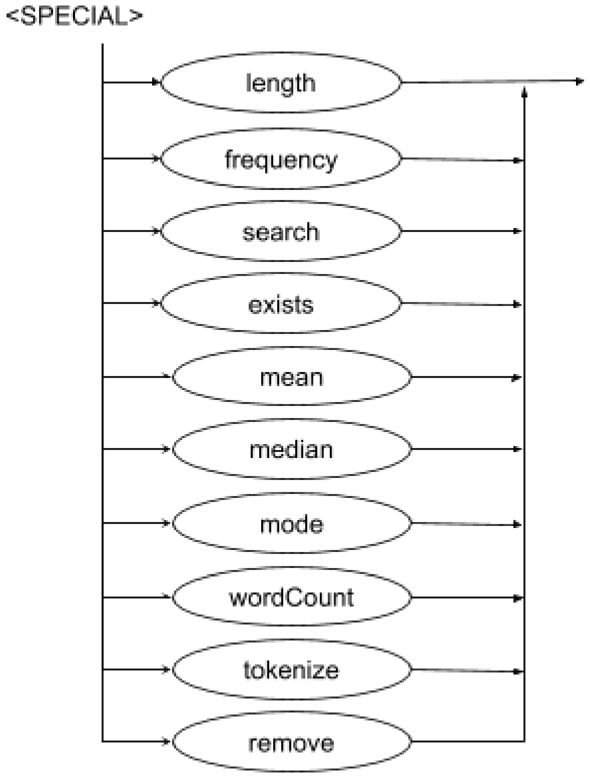

while
(
EXPRESSION
)
BLOCK
do
BLOCK
while
(
EXPRESSION
)
;

<FUNCTION>


,
TYPE
id
function
TYPE
void
id
(
)
{
DECLARE
ACTION
return
id
;
}

<SPECIALFUNCTION>


id
.
SPECIAL
(
EXP
)
;



Main semantic characteristics

- Operations will only be made between same data types.
- Relational and arithmetic expressions only work for *number* data type.
- Arrays will only accept *number, bool* or *word* data types.
- Array assignments (content) must match the data type with which it was created.
- Array assignment can only be done either by ‘id = {Exp, Exp, ... }’ or ‘id[Exp] = Exp’.
- Special functions may be used only by their assigned data type, for example, function “tokenize” will only work for *sentences.*
- Functions that retrieve a value must be assigned to a variable of the same data type as the result.

Description of functions

Functions for Words

- **Length**: function available for data type *word*. Retrieves the length of the word as a number. No parameters.
- **Frequency**: function available for data type *word.* Retrieves the number of times that a word appears in a certain array. One parameter (a word array).
- **Search**: function available for data type *word.* Retrieves the position of a word in a certain array. One parameter (a word array).
- **Exists**: function available for data type *word*. Retrieves a boolean expression that states if a word exists in a certain array. One parameter (a word array).

Functions for Numbers

- **Mean**: function available for number arrays. Retrieves the average of the numbers in the array. No parameters.
- **Median**: function available for number arrays. Retrieves the median number in a certain array. No parameters.
- **Mode**: function available for number arrays. Retrieves the number with the highest frequency in an array. No parameters.

Functions for Sentences

- **WordCount**: function available for data type *sentence.* Retrieves the number of words in the sentence. no parameters.
- **Tokenize:** function available for data type *sentence.* Retrieves an array containing the words in the sentence. No parameters.
- **Remove**: function available for data type *sentence*. Retrieves the same sentence without the character that was added as a parameter in the function. One parameter (a word).

Data types

Word: group of characters without any spaces or special characters. Just capital and lowercase letters.
Sentence: group of any type of characters.
Number: a float number
Bool: true or false

Language and OS

The language that will be used to create Kotoba is Python 3 using Ply for the Lex & Yacc. It will be created using macOS Mojave with the most recent software update.

Bibliography

PLY (Python Lex-Yacc). (n.d.). Retrieved January 26, 2019, from

https://www.dabeaz.com/ply/