B. Language Compiler

Khaled Daoudieh

Compiler Construction - January 13, 2016

Lex B. Language

• Lex B. is language that was build using Lex , Bison and C programming language. and also it is an interpreted language so it runs in runtime , allocate and deallocate memory space with all generic types and variables , it is similar or in between C and python as it doesn't have main but it has functions that can be called at anytime excluding the main allow the user to execute functions and call them as soon as they are needed.

LANGUAGE SYNTAX :

❖ Literals :

1. The Language supports the following types:

A- Integers:

3 , -4

B- Real Numbers:

3.5 , -0.12

C. Strings:
“ B. is awesome ! ”

D. Variables:

Variables in B. Language are dynamic, they can be of any type and change their type on assignment.

Example:

```
a = 20;
print a;
a = 33.5;
print a;
a = “B. is awesome !
      print a;
```

The output would be “

20
33.5
B. is awesome !“

❖ Operators :

1. Unary operators

Unary operators : operators that are prefixed to an expression.

Following Operations are supported :

— Operand;

This causes the value of the operant to be negated (multiplied by -1). String operands can’t be negated.

Examples:

def b = 5;

print -b;

output :

-5

2- Normal Operators :

expr 1 + expr 2
sum of the two expression
expr. 1 - expr. 2
the value of expression 2 subtracted from expression 1

expr. 1 / expr. 2
the result of dividing expression. 1 by expression. 2.

expr. 1 * expr. 2

the multiplication of the two expression.

expr. 1 mod expr. 2

the remainder of dividing expression. 1 by expression. 2.

Examples:

print 50 - 30;

output:

20

print 2 - 6;

outputs

-4

print 3 mod 2;

outputs :

1;

3. Boolean operators :

operators that return either 1 (true) or 0 (false).

B. supports of boolean operators:

expr. 1 < expr. 2

returns 1 (true) if expression. 1 is less than expression. 2, else returns 0 (false).

expr. 1 > expr. 2

returns 1 (true) if expression. 1 is greater than expression. 2, else returns 0 (false).

expr. 1 <= expr. 2

returns 1 (true) if expression. 1 is less than or equal to expression. 2, else returns 0 (false).

expr. 1 >= expr. 2

returns 1 (true) if expression. 1 is greater than or equal to expression. 2, else returns 0 (false).

expr. 1 == expr. 2

returns 1 (true) if expression. 1 is equal to expression. 2, else returns 0 (false).

expr. 1 != expr. 2

returns 1 (true) if expression. 1 is not equal to expression. 2, else returns 0 (false).

Example:

print 34 >= 30;

This outputs 1 (true) because 34 is less than or equal (it is bigger than) 30 ;

print 50 != 20;

This outputs 1 (true) because 50 does not equal 20 ;

Strings

Strings in B. Language are similar to strings to Python and C.

Starting with a double quotation mark and ending with the same

example :

“ B. is Awesome “

1. Escape sequences

\n New line , Moves the cursor to the next line

\t Horizontal Tab , move to the next horizontal tabular position.

\" Produces double quotes.

Operators :

Statements

1. Declarations

A= 5;

Defines variable in the memory with the name A .

2. Control statements :

Statements that change the flow of statements .

3. Conditional statements.

```
    if ( Expr )
{
stmt1;
 }

else
 {

stmt2 ;
}
```

if the expression applies then statement 1 is excited other than that statement 2 is executed.

4. Loops

- while (expression1) statement1

while loop is a loop that executes statement1 as long as expression1 is true

5. For Loop :

A for loop is a loop that executes statement1 according to var times as long var is positive and the expression 1 is true .

The integer literal can also be a variable of integer value.

For (var ; expression1; increment)

{

statement1

}

6. Function

A function is a sequence of commands as you create and a function and it doesn't get excited until you call it when needed and

example :

```
def func_name (){

statement1

statement2

.

.

} // define the function

func_name() ;  // call the function .
```