CSE 341 Programming Languages Documentation

FIRST PART!

Lex (Flex) Part:

Defines regular expressions for tokens like keywords (true, false, and, etc.), operators (+, -, *, /, etc.), comments, identifiers, and values.

Specifies actions to be taken when a token is recognized, such as returning a specific token type or performing additional processing.

Yacc Part:

Defines a set of grammar rules for the language.

Specifies actions associated with each grammar rule, indicating what to do when a specific syntactic structure is recognized.

Uses tokens generated by Flex as input.

Keywords and Operators:

Keywords include boolean values (true, false), logical operators (and, or, not), comparison operators (equal, less), and control flow keywords (if, for, load, exit, etc.).

Mathematical operators include addition (+), subtraction (-), multiplication (*), division (/), and parentheses for grouping.

Functions and Expressions:

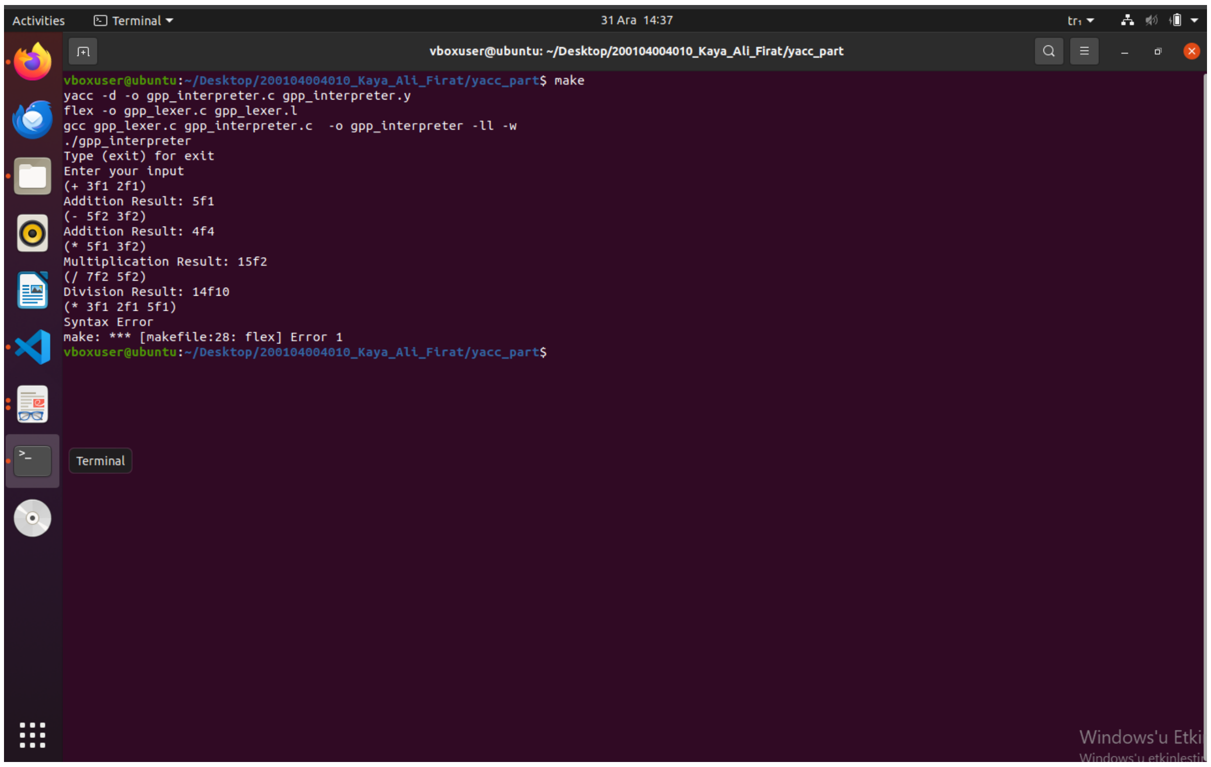
The code handles mathematical operations on fractions, with functions like operateFractions for addition, subtraction, multiplication, and division.

Defines a basic set of mathematical operations using Yacc rules for expressions.

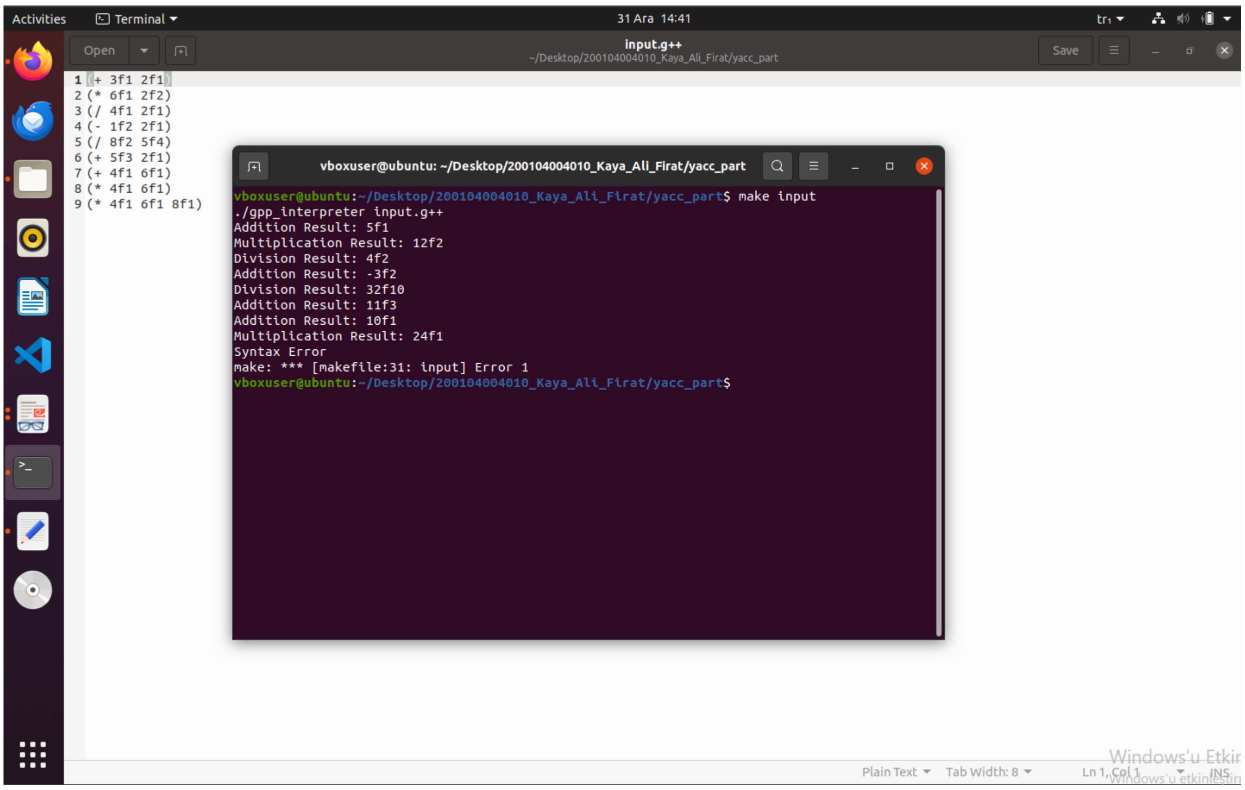
Variables and Identifiers:

The code supports the definition and manipulation of identifiers and variables.

The addIdentifier function is used to add or update identifiers and their corresponding values.


Activities
Terminal
31 Ara 14:37
vboxuser@ubuntu: ~/Desktop/200104004010_Kaya_Ali_Firat/yacc_part
Terminal

```
vboxuser@ubuntu:~/Desktop/200104004010_Kaya_Ali_Firat/yacc_part$ make
yacc -d -o gpp_interpreter.c gpp_interpreter.y
flex -o gpp_lexer.c gpp_lexer.l
gcc gpp_lexer.c gpp_interpreter.c  -o gpp_interpreter -ll -w
./gpp_interpreter
Type (exit) for exit
Enter your input
(+ 3f1 2f1)
Addition Result: 5f1
(- 5f2 3f2)
Addition Result: 4f4
(* 5f1 3f2)
Multiplication Result: 15f2
(/ 7f2 5f2)
Division Result: 14f10
(* 3f1 2f1 5f1)
Syntax Error
make: *** [makefile:28: flex] Error 1
vboxuser@ubuntu:~/Desktop/200104004010_Kaya_Ali_Firat/yacc_part$
```


Activities
Terminal
31 Ara 14:41
Open
input.g++
~/Desktop/200104004010_Kaya_Ali_Firat/yacc_part
Save
vboxuser@ubuntu: ~/Desktop/200104004010_Kaya_Ali_Firat/yacc_part
Plain Text
Tab Width: 8
Ln 1, Col 1
INS

```
(+ 3f1 2f1)
(* 6f1 2f2)
(/ 4f1 2f1)
(- 1f2 2f1)
(/ 8f2 5f4)
(+ 5f3 2f1)
(+ 4f1 6f1)
(* 4f1 6f1)
(* 4f1 6f1 8f1)
```

```
vboxuser@ubuntu:~/Desktop/200104004010_Kaya_Ali_Firat/yacc_part$ make input
./gpp_interpreter input.g++
Addition Result: 5f1
Multiplication Result: 12f2
Division Result: 4f2
Addition Result: -3f2
Division Result: 32f10
Addition Result: 11f3
Addition Result: 10f1
Multiplication Result: 24f1
Syntax Error
make: *** [makefile:31: input] Error 1
vboxuser@ubuntu:~/Desktop/200104004010_Kaya_Ali_Firat/yacc_part$
```

SECOND PART!

NOTE: My Lisp code accepts 4b1 style inputs, not 4f1 style.

Tokenization:

Reads input strings and tokenizes them into categories like keywords (KW_AND, KW_OR, KW_DEF), identifiers, numbers, and comments.

Parentheses Balance:

Checks for balanced parentheses in the input strings.

Mathematical Operations:

Processes basic mathematical operations such as addition, subtraction, multiplication, and division.

Special handling for division, replacing slashes with 'b' in the result.

Global Array Management:

Maintains a global array (global-array) and updates it with information from each processed string.

Function Name Detection:

Identifies function names based on the processed input.

Error Handling:

Provides error messages for syntax errors and invalid input.

File Processing:

Includes functions to process the contents of a file and tokenize them.

User Interaction:

Allows user input, and typing "exit" terminates the program.

List Operations:

Performs various mathematical operations on lists.

Fraction Handling:

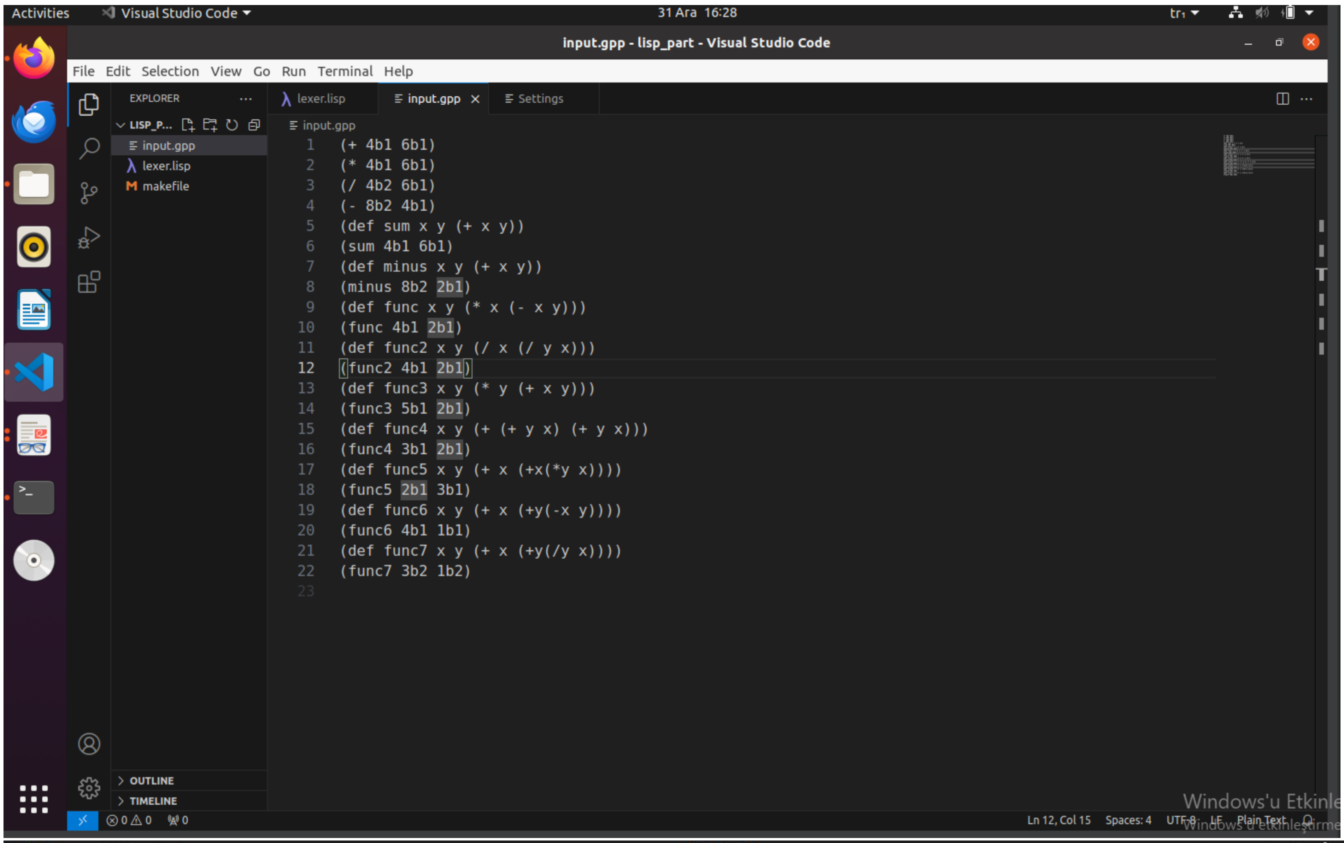
Recognizes and processes unsigned fractions in the format of 'NUMBER-b-NUMBER'.

Input Verification:

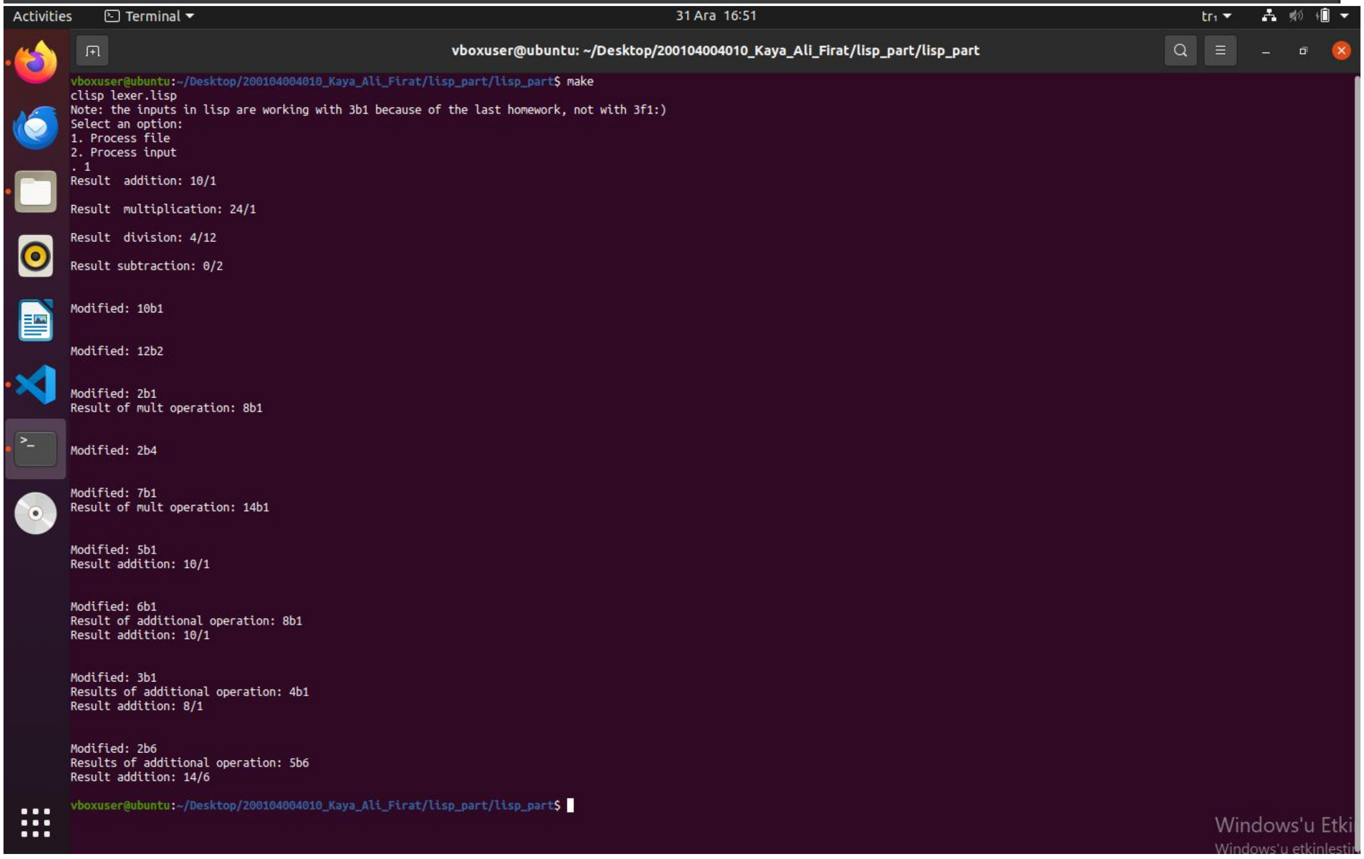
Checks if a string is a number, an unsigned fraction, or has invalid syntax.

Printing Tokens:

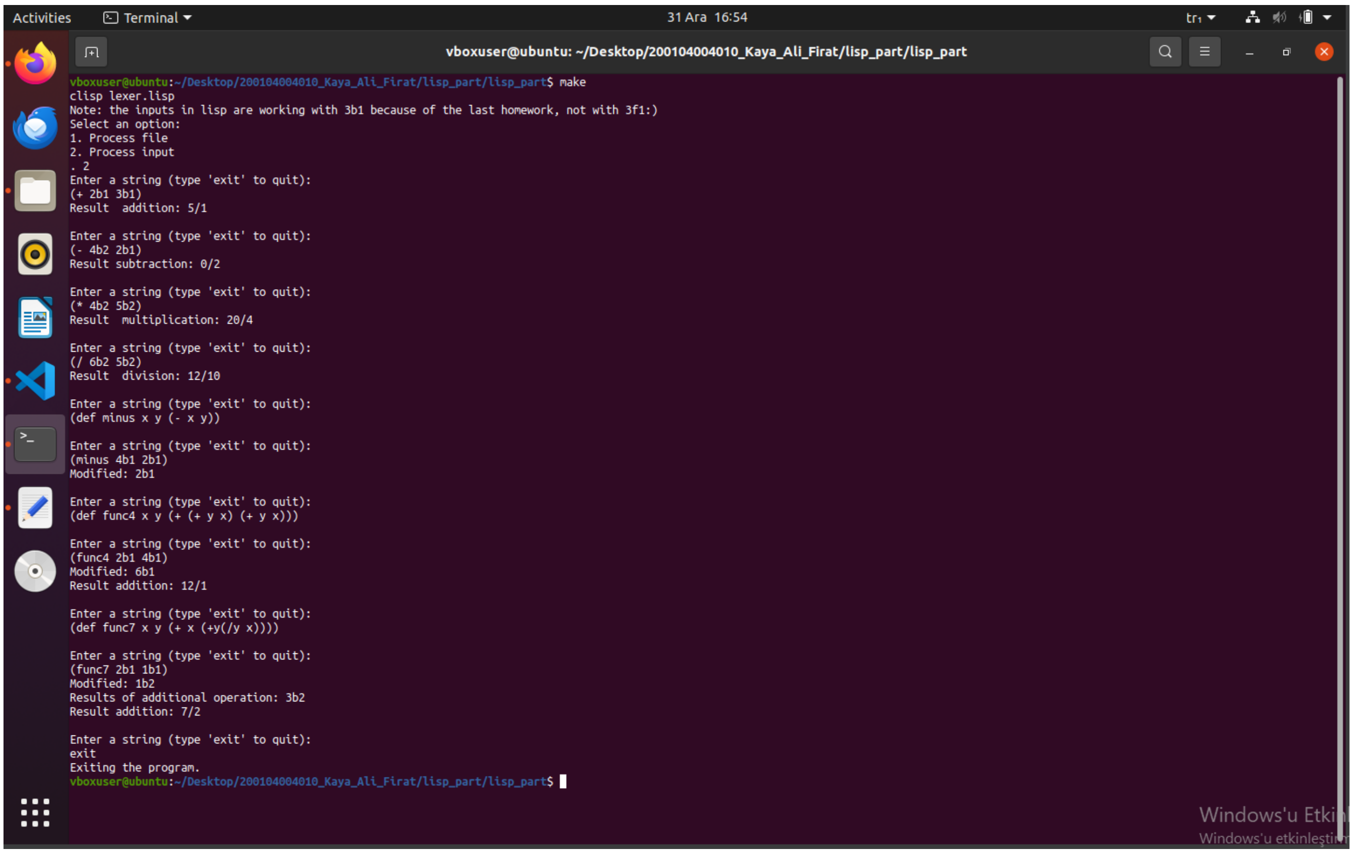
Prints tokens based on their types or keywords.


Activities
Visual Studio Code
31 Ara 16:28
input.gpp - lisp_part - Visual Studio Code
File Edit Selection View Go Run Terminal Help
EXPLORER
LISP_P...
input.gpp
lexer.lisp
makefile
lexer.lisp
input.gpp
Settings
OUTLINE
TIMELINE
Ln 12, Col 15 Spaces: 4 UTF-8 LF Plain Text

```
(+ 4b1 6b1)
(* 4b1 6b1)
(/ 4b2 6b1)
(- 8b2 4b1)
(def sum x y (+ x y))
(sum 4b1 6b1)
(def minus x y (+ x y))
(minus 8b2 2b1)
(def func x y (* x (- x y)))
(func 4b1 2b1)
(def func2 x y (/ x (/ y x)))
(func2 4b1 2b1)
(def func3 x y (* y (+ x y)))
(func3 5b1 2b1)
(def func4 x y (+ (+ y x) (+ y x)))
(func4 3b1 2b1)
(def func5 x y (+ x (+x(*y x))))
(func5 2b1 3b1)
(def func6 x y (+ x (+y(-x y))))
(func6 4b1 1b1)
(def func7 x y (+ x (+y(/y x))))
(func7 3b2 1b2)
```


Activities
Terminal
31 Ara 16:51
vboxuser@ubuntu: ~/Desktop/200104004010_Kaya_Ali_Firat/lisp_part/lisp_part

```
vboxuser@ubuntu:~/Desktop/200104004010_Kaya_Ali_Firat/lisp_part/lisp_part$ make
clisp lexer.lisp
Note: the inputs in lisp are working with 3b1 because of the last homework, not with 3f1:)
Select an option:
1. Process file
2. Process input
. 1
Result  addition: 10/1

Result  multiplication: 24/1

Result  division: 4/12

Result subtraction: 0/2


Modified: 10b1


Modified: 12b2


Modified: 2b1
Result of mult operation: 8b1


Modified: 2b4


Modified: 7b1
Result of mult operation: 14b1


Modified: 5b1
Result addition: 10/1


Modified: 6b1
Result of additional operation: 8b1
Result addition: 10/1


Modified: 3b1
Results of additional operation: 4b1
Result addition: 8/1


Modified: 2b6
Results of additional operation: 5b6
Result addition: 14/6

vboxuser@ubuntu:~/Desktop/200104004010_Kaya_Ali_Firat/lisp_part/lisp_part$
```


Activities
Terminal
31 Ara 16:54
tr
vboxuser@ubuntu: ~/Desktop/200104004010_Kaya_Ali_Firat/lisp_part/lisp_part

```
vboxuser@ubuntu:~/Desktop/200104004010_Kaya_Ali_Firat/lisp_part/lisp_part$ make
clisp lexer.lisp
Note: the inputs in lisp are working with 3b1 because of the last homework, not with 3f1:)
Select an option:
1. Process file
2. Process input
. 2
Enter a string (type 'exit' to quit):
(+ 2b1 3b1)
Result  addition: 5/1

Enter a string (type 'exit' to quit):
(- 4b2 2b1)
Result subtraction: 0/2

Enter a string (type 'exit' to quit):
(* 4b2 5b2)
Result  multiplication: 20/4

Enter a string (type 'exit' to quit):
(/ 6b2 5b2)
Result  division: 12/10

Enter a string (type 'exit' to quit):
(def minus x y (- x y))

Enter a string (type 'exit' to quit):
(minus 4b1 2b1)
Modified: 2b1

Enter a string (type 'exit' to quit):
(def func4 x y (+ (+ y x) (+ y x)))

Enter a string (type 'exit' to quit):
(func4 2b1 4b1)
Modified: 6b1
Result addition: 12/1

Enter a string (type 'exit' to quit):
(def func7 x y (+ x (+y(/y x))))

Enter a string (type 'exit' to quit):
(func7 2b1 1b1)
Modified: 1b2
Results of additional operation: 3b2
Result addition: 7/2

Enter a string (type 'exit' to quit):
exit
Exiting the program.
vboxuser@ubuntu:~/Desktop/200104004010_Kaya_Ali_Firat/lisp_part/lisp_part$
```