Gebze Technical University

Programming Languages
CSE 341

Homework 3
Report

Elif Akgün

1801042251

- I implemented *and, or, not, equal, less, nil, list, append, concat, set, deffun, defvar, for, while, if, exit, load, true, false, +(plus), -(minus), /(div), *(mult) ,**(dblmult),* operations. There is KW_DISP keyword in GppSyntax.pdf file but there is no example with this keyword. I could not find what disp operation does so I did not implement it.
- For list operations, you should use '(…). You can use (list …) format only a single operation like (list 1 2 3). (concat (list 1 2) (list 3 4)) does not run.
- To program exit, you should use empty input as previous homework. So, in your input file, there must be an empty line end of the file to program exits.
- When incorrect line read, program writes an error message then exits.

1) G++ Language Interpreter using Flex and Yacc

You can run this part with using makefile and following commands:

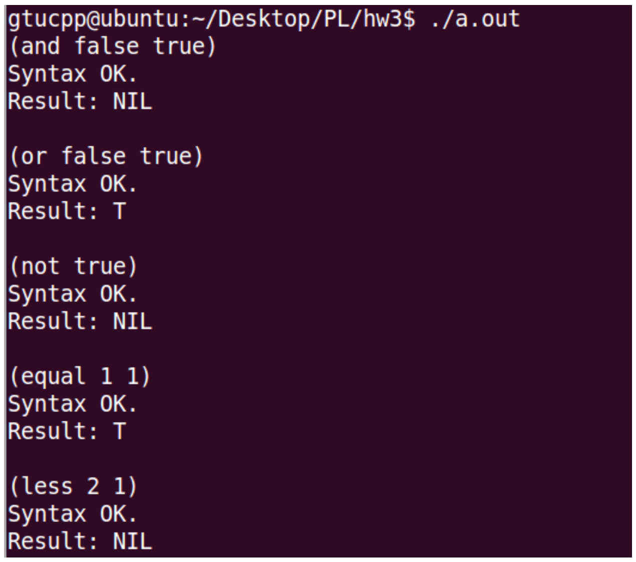
- make
- ./gpp_interpreter.out

In this part, output is written in "parsed_cpp.txt" file. To write output file, you should enter empty line as input otherwise, it is not written to the file.

If entered line is correct, it writes "Syntax OK". According to expressions, Result changes. There are sample inputs and outputs below.

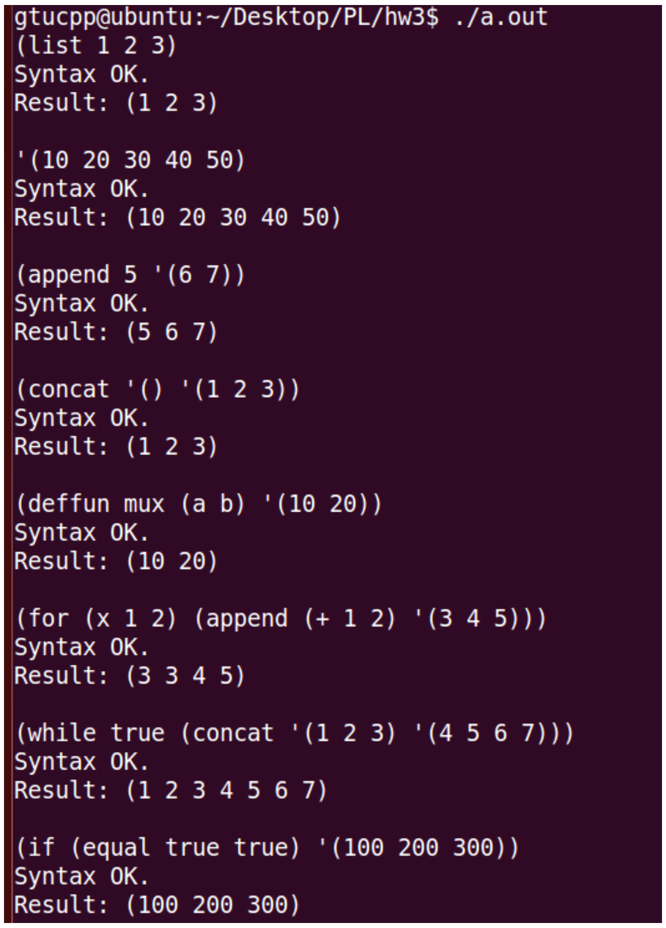
In homework file, the following sentence is included: "You are also expected to submit the corresponding C file generated by your lexer and syntax analyzer along with any other .h or .c file that is needed to compile "gpp_interpreter.c" using GCC on Ubuntu (16 or later). ". There is only gpp_interpreter.l and gpp_interpreter.y files in template zip file that shared in announcements in moodle, so there is only gpp_interpreter.l and gpp_interpreter.y file in my zip file that i ulpoaded.

- If operation is one of *and, or, not, equal, or less*; it writes *Result: NIL* or *Result: T* .



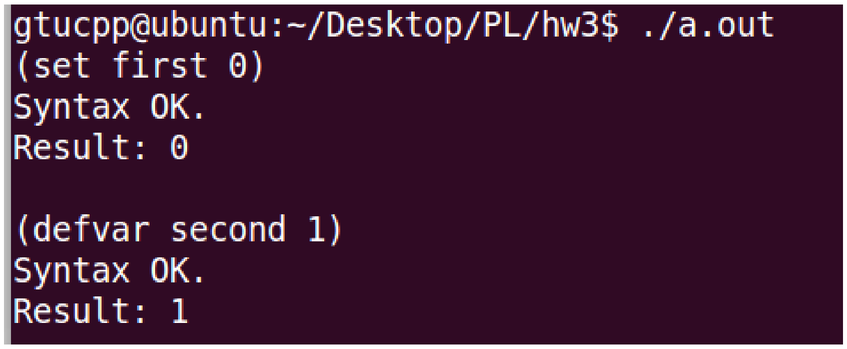
```
gtucpp@ubuntu:~/Desktop/PL/hw3$ ./a.out
(and false true)
Syntax OK.
Result: NIL

(or false true)
Syntax OK.
Result: T

(not true)
Syntax OK.
Result: NIL

(equal 1 1)
Syntax OK.
Result: T

(less 2 1)
Syntax OK.
Result: NIL
```

- If operation is one of *list, append, concat, deffun, for, while, if*; result will be a list.



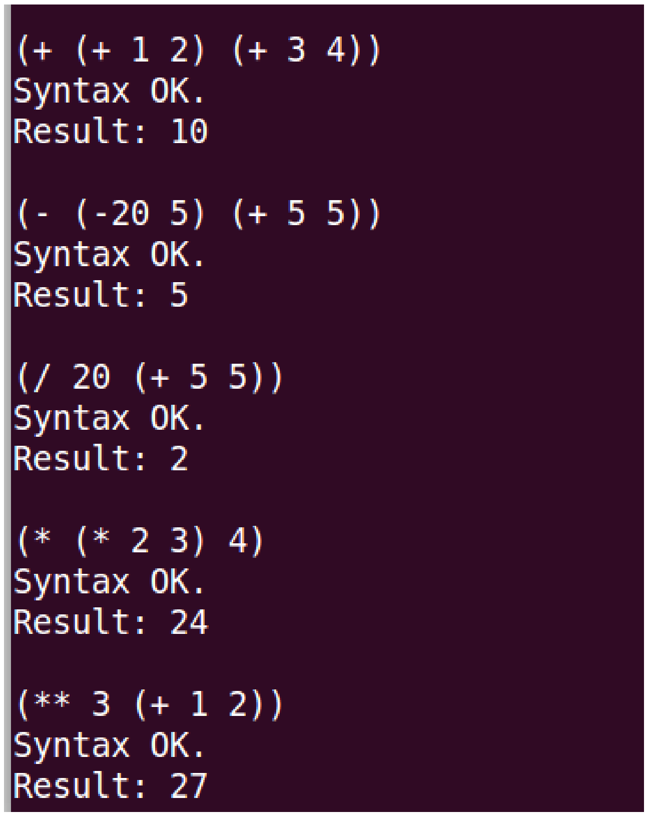
```
gtucpp@ubuntu:~/Desktop/PL/hw3$ ./a.out
(list 1 2 3)
Syntax OK.
Result: (1 2 3)

'(10 20 30 40 50)
Syntax OK.
Result: (10 20 30 40 50)

(append 5 '(6 7))
Syntax OK.
Result: (5 6 7)

(concat '() '(1 2 3))
Syntax OK.
Result: (1 2 3)

(deffun mux (a b) '(10 20))
Syntax OK.
Result: (10 20)

(for (x 1 2) (append (+ 1 2) '(3 4 5)))
Syntax OK.
Result: (3 3 4 5)

(while true (concat '(1 2 3) '(4 5 6 7)))
Syntax OK.
Result: (1 2 3 4 5 6 7)

(if (equal true true) '(100 200 300))
Syntax OK.
Result: (100 200 300)
```

- If operation is one of *set and defvar, if*; result will be assigned value.



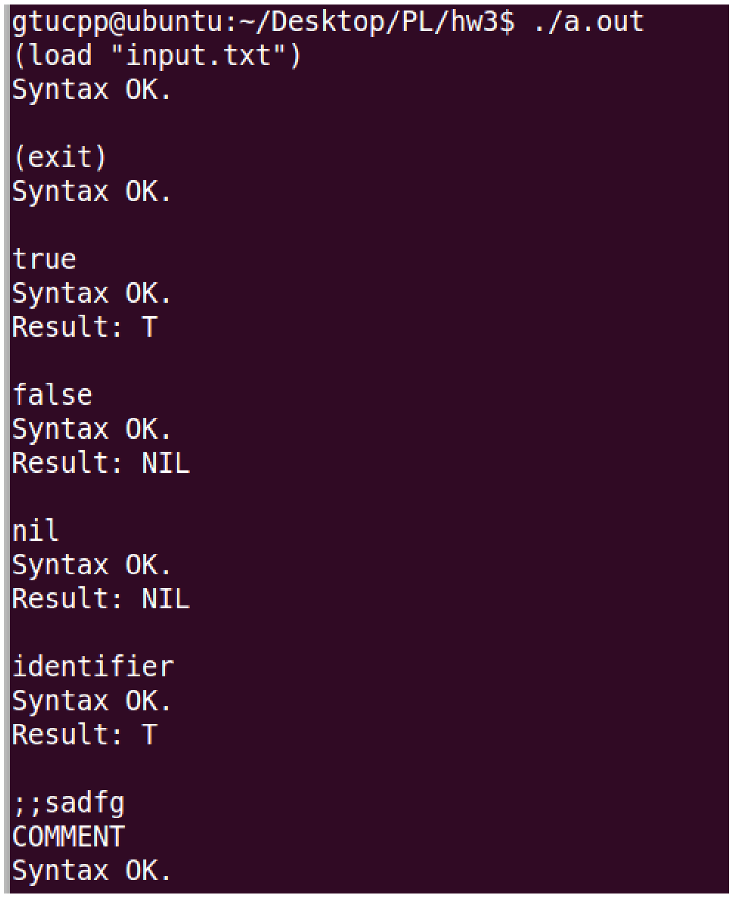
```
gtucpp@ubuntu:~/Desktop/PL/hw3$ ./a.out
(set first 0)
Syntax OK.
Result: 0

(defvar second 1)
Syntax OK.
Result: 1
```

- If operation is one of *+, -, /, *, and ***; result will be result of operation.



```

(+ (+ 1 2) (+ 3 4))
Syntax OK.
Result: 10

(- (-20 5) (+ 5 5))
Syntax OK.
Result: 5

(/ 20 (+ 5 5))
Syntax OK.
Result: 2

(* (* 2 3) 4)
Syntax OK.
Result: 24

(** 3 (+ 1 2))
Syntax OK.
Result: 27
```

- In case of *exit* and *load* operations, it writes only SYNTAX OK.



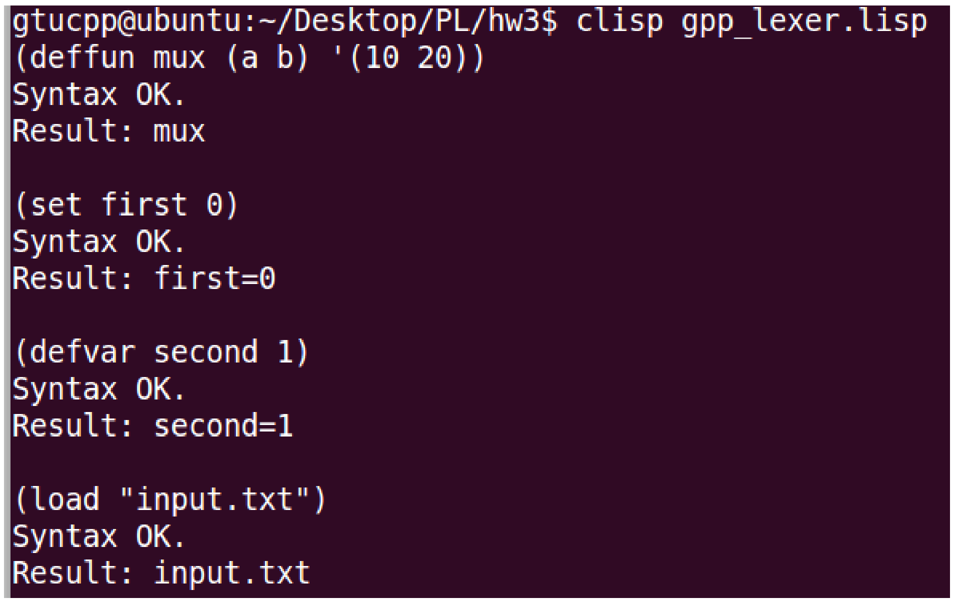
```
gtucpp@ubuntu:~/Desktop/PL/hw3$ ./a.out
(load "input.txt")
Syntax OK.

(exit)
Syntax OK.

true
Syntax OK.
Result: T

false
Syntax OK.
Result: NIL

nil
Syntax OK.
Result: NIL

identifier
Syntax OK.
Result: T

;;sadfg
COMMENT
Syntax OK.
```

2) G++ Language Interpreter in Lisp

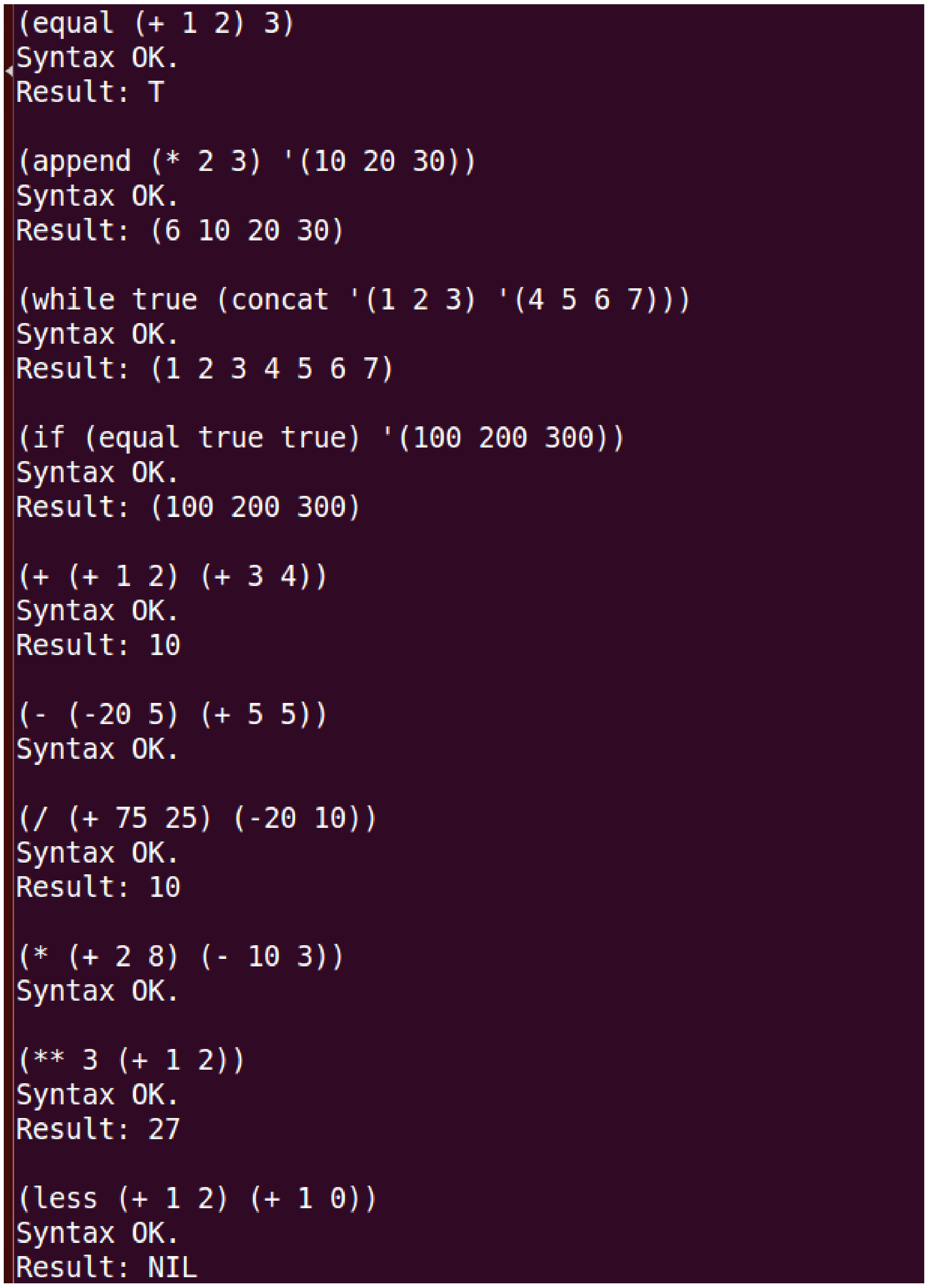
You can run this part following commands: clisp gpp_interpreter.lisp or clisp gpp_interpreter.lisp <file_name>

In this part, output is written in “parsed_lisp.txt” file.

If entered line is correct, it writes “Syntax OK”. According to expressions, Result changes. Except deffun, set, defvar, load operations; all results are the same with part a. Unlike part a, if operation is deffun, set, defvar, load; result includes variable’s name. There are some inputs and outputs below.



```
gtucpp@ubuntu:~/Desktop/PL/hw3$ clisp gpp_lexer.lisp
(deffun mux (a b) '(10 20))
Syntax OK.
Result: mux

(set first 0)
Syntax OK.
Result: first=0

(defvar second 1)
Syntax OK.
Result: second=1

(load "input.txt")
Syntax OK.
Result: input.txt
```



```
(equal (+ 1 2) 3)
Syntax OK.
Result: T

(append (* 2 3) '(10 20 30))
Syntax OK.
Result: (6 10 20 30)

(while true (concat '(1 2 3) '(4 5 6 7)))
Syntax OK.
Result: (1 2 3 4 5 6 7)

(if (equal true true) '(100 200 300))
Syntax OK.
Result: (100 200 300)

(+ (+ 1 2) (+ 3 4))
Syntax OK.
Result: 10

(- (-20 5) (+ 5 5))
Syntax OK.

(/ (+ 75 25) (-20 10))
Syntax OK.
Result: 10

(* (+ 2 8) (- 10 3))
Syntax OK.

(** 3 (+ 1 2))
Syntax OK.
Result: 27

(less (+ 1 2) (+ 1 0))
Syntax OK.
Result: NIL
```