Report on

“Mini Python Compiler”

Submitted in partial fulfillment of the requirements for ***Sem VI***

Compiler Design Laboratory

Bachelor of Technology
in
Computer Science & Engineering

Submitted by:

Chirag N Vijay	**01FB16ECS099**
Dhanush Ravi	**01FB16ECS112**
Bharath Chandra	**01FB16ECS087**

Under the guidance of

Madhura V

PES University, Bengaluru

January – May 2019

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING
FACULTY OF ENGINEERING
PES UNIVERSITY
(Established under Karnataka Act No. 16 of 2013)
100ft Ring Road, Bengaluru – 560 085, Karnataka, India

TABLE OF CONTENTS

INTRODUCTION

Main focus is to build a mini compiler for Python which is able to handle most of the selection statements and loops.
Sample input - The respective python file which needs to be compiled
Sample output - Symbol Table,Abstract Syntax Tree, 3 address code and the final optimised code.

ARCHITECTURE

The following things have been handled
-> User defined functions
->For loop
->while loop
->try and except clauses
->If and else and elif statements
->lists
->Binary operations and arithmetic operations
->Import statements
->Lambda functions

REFERENCES

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

https://www.google.com/url?sa=t&rct=j&q=&esrc=s&source=web&cd=2&cad=rja&uact=8&ved=2ahUKEwjx1qiy6-vhAhXGX30KHZLfCyQQFjABegQIARAC&url=https%3A%2F%2Fwww.dabeaz.com%2Fply%2FPLYTalk.pdf&usg=AOvVaw2R2Znb78j2Z84r7XbVKzfx

https://github.com/dabeaz/ply
http://compileroptimizations.com/category/constant_folding.html

http://compileroptimizations.com/category/dead_code_elimination.htm

http://jsonviewer.stack.hu/

https://github.com/pgbovine/python-parse-to-json

CONTEXT FREE GRAMMAR

```
single_input: NEWLINE | simple_stmt | compound_stmt NEWLINE
file_input: (NEWLINE | stmt)* ENDMARKER
eval_input: testlist NEWLINE* ENDMARKER

decorator: '@' dotted_name [ '(' [arglist] ')' ] NEWLINE
decorators: decorator+
decorated: decorators (classdef | funcdef | async_funcdef)

async_funcdef: 'async' funcdef
funcdef: 'def' NAME parameters ['->' test] ':' suite

parameters: '(' [typedargslist] ')'
typedargslist: (tfpdef ['=' test] (',' tfpdef ['=' test])* [',' [
        '*' [tfpdef] (',' tfpdef ['=' test])* [',' ['**' tfpdef [',']]]
      | '**' tfpdef [',']]]
  | '*' [tfpdef] (',' tfpdef ['=' test])* [',' ['**' tfpdef [',']]]
  | '**' tfpdef [','])
tfpdef: NAME [':' test]
varargslist: (vfpdef ['=' test] (',' vfpdef ['=' test])* [',' [
        '*' [vfpdef] (',' vfpdef ['=' test])* [',' ['**' vfpdef [',']]]
      | '**' vfpdef [',']]]
  | '*' [vfpdef] (',' vfpdef ['=' test])* [',' ['**' vfpdef [',']]]
  | '**' vfpdef [',']
)
vfpdef: NAME

stmt: simple_stmt | compound_stmt
simple_stmt: small_stmt (';' small_stmt)* [';'] NEWLINE
```

```
small_stmt: (expr_stmt | del_stmt | pass_stmt | flow_stmt |
             import_stmt | global_stmt | nonlocal_stmt | assert_stmt)
expr_stmt: testlist_star_expr (annassign | augassign (yield_expr|testlist)
|
                     ('=' (yield_expr|testlist_star_expr))*)
annassign: ':' test ['=' test]
testlist_star_expr: (test|star_expr) (',' (test|star_expr))* [',']
augassign: ('+=' | '-=' | '*=' | '@=' | '/=' | '%=' | '&=' | '|=' | '^=' |
            '<<=' | '>>=' | '**=' | '//=')

del_stmt: 'del' exprlist
pass_stmt: 'pass'
flow_stmt: break_stmt | continue_stmt | return_stmt | raise_stmt |
yield_stmt
break_stmt: 'break'
continue_stmt: 'continue'
return_stmt: 'return' [testlist]
yield_stmt: yield_expr
raise_stmt: 'raise' [test ['from' test]]
import_stmt: import_name | import_from
import_name: 'import' dotted_as_names

import_from: ('from' (('.' | '...')* dotted_name | ('.' | '...')+)
              'import' ('*' | '(' import_as_names ')' | import_as_names))
import_as_name: NAME ['as' NAME]
dotted_as_name: dotted_name ['as' NAME]
import_as_names: import_as_name (',' import_as_name)* [',']
dotted_as_names: dotted_as_name (',' dotted_as_name)*
dotted_name: NAME ('.' NAME)*
global_stmt: 'global' NAME (',' NAME)*
nonlocal_stmt: 'nonlocal' NAME (',' NAME)*
assert_stmt: 'assert' test [',' test]

compound_stmt: if_stmt | while_stmt | for_stmt | try_stmt | with_stmt |
funcdef | classdef | decorated | async_stmt
async_stmt: 'async' (funcdef | with_stmt | for_stmt)
if_stmt: 'if' test ':' suite ('elif' test ':' suite)* ['else' ':' suite]
while_stmt: 'while' test ':' suite ['else' ':' suite]
```

```
for_stmt: 'for' exprlist 'in' testlist ':' suite ['else' ':' suite]
try_stmt: ('try' ':' suite
           ((except_clause ':' suite)+
            ['else' ':' suite]
            ['finally' ':' suite] |
           'finally' ':' suite))
with_stmt: 'with' with_item (',' with_item)*  ':' suite
with_item: test ['as' expr]

except_clause: 'except' [test ['as' NAME]]
suite: simple_stmt | NEWLINE INDENT stmt+ DEDENT

test: or_test ['if' or_test 'else' test] | lambdef
test_nocond: or_test | lambdef_nocond
lambdef: 'lambda' [varargslist] ':' test
lambdef_nocond: 'lambda' [varargslist] ':' test_nocond
or_test: and_test ('or' and_test)*
and_test: not_test ('and' not_test)*
not_test: 'not' not_test | comparison
comparison: expr (comp_op expr)*

comp_op: '<'|'>'|'=='|'>='|'<='|'<>'|'!='|'in'|'not' 'in'|'is'|'is' 'not'
star_expr: '*' expr
expr: xor_expr ('|' xor_expr)*
xor_expr: and_expr ('^' and_expr)*
and_expr: shift_expr ('&' shift_expr)*
shift_expr: arith_expr (('<<'|'>>') arith_expr)*
arith_expr: term (('+'|'-') term)*
term: factor (('*'|'@'|'/'|'%'|'//') factor)*
factor: ('+'|'-'|'~') factor | power
power: atom_expr ['**' factor]
atom_expr: ['await'] atom trailer*
atom: ('(' [yield_expr|testlist_comp] ')' |
       '[' [testlist_comp] ']' |
       '{' [dictorsetmaker] '}' |
       NAME | NUMBER | STRING+ | '...' | 'None' | 'True' | 'False')
testlist_comp: (test|star_expr) ( comp_for | (',' (test|star_expr))* [','] )
trailer: '(' [arglist] ')' | '[' subscriptlist ']' | '.' NAME
```

```
subscriptlist: subscript (',' subscript)* [',']
subscript: test | [test] ':' [test] [sliceop]
sliceop: ':' [test]
exprlist: (expr|star_expr) (',' (expr|star_expr))* [',']
testlist: test (',' test)* [',']
dictorsetmaker: ( ((test ':' test | '**' expr)
                  (comp_for | (',' (test ':' test | '**' expr))* [','])) |
                 ((test | star_expr)
                  (comp_for | (',' (test | star_expr))* [','])) )

classdef: 'class' NAME ['(' [arglist] ')'] ':' suite

arglist: argument (',' argument)*  [',']

argument: ( test [comp_for] |
            test '=' test |
            '**' test |
            '*' test )

comp_iter: comp_for | comp_if
sync_comp_for: 'for' exprlist 'in' or_test [comp_iter]
comp_for: ['async'] sync_comp_for
comp_if: 'if' test_nocond [comp_iter]


encoding_decl: NAME

yield_expr: 'yield' [yield_arg]
yield_arg: 'from' test | testlist
```

DESIGN STRATEGY

SYMBOL TABLE

Here the construction of symbol table takes scope and scopestack into consideration for assigning values to variables, assigning attributes and declaring identifiers.
It also takes care of the function default parameters.It is also responsible for assigning the width for different identifier types.

ABSTRACT SYNTAX TREE

Here we are retrieving the parse tree from the parser and then we are converting the parse tree entries to json objects which then are fed to an online tree generator which gives us the constructed abstract syntax tree

INTERMEDIATE CODE GENERATION

We are storing the intermediate code in 3 address format - the quadraple format using registers.We are using a list of lists for storing the intermediate code.
We are handling the icg also for user defined function calls .

OPTIMISATIONS

Here the two optimisations which we are performing are
->Constant Folding
-> Dead Code Elimination.
The constructs being used here are lists, we are opening and storing the icg code in a list of lines and then we apply regex to classify identifiers and numbers/constants seperately .

IMPLEMENTATIONS

SYMBOL TABLE

The following functions are implemented to build the symbol table
->lookup,lookupScopeStack--- Looks for an identifier with the help of the scope numbers we generated in the lexical phase
->getcurrentscope
->addscope -> for user defined functions
->addidentifier
->addattribute,getattribute
->getattributefrom currentscope
->addattributeto currentscope
->getattributefromfunction list
->printst
->getwidthfromtype
->printsymboltablehistory

Structure

Entry : Scopename
Type
Return type

INTERMEDIATE CODE GENERATION

Here we are using an arraylist or a list of lists to implement the intermediate code, we are using quadruple format to store the three address code and storing the quadraple of each instruction in a seperate list, then we concatenate the induvidual lists to a master list which gives us the whole code of the program.

Some functions being implemented are
->incrementQuad
->getnextQuad
->emit
->createnewFunctionCode
->printCode

OPTIMIZATIONS

The optimisations being implemented are Constant Folding and Dead-Code Elimination.
Main functions being implemented are
evalwrap() -> Evaluates each instruction.
fold_constant() -> Does constant folding ,takes list_of_lines as function arguments and outputs the optimized code.
remove_dead_code() -> Removes dead code ,takes list_of_lines as function arguments and outputs the optimized code.

ERRORS BEING HANDLED

Syntax errors,parsing errors and value and name errors are being handled.

RESULTS

Following are the results we obtained

TEST CODE 1

```
a = 100
if a < 200:
    c = 2000
    d = 200*10
    isEqual = (c==d)
    if isEqual==True:
        print("Hello")
    else:
        d = 1000
```

TEST CODE 2

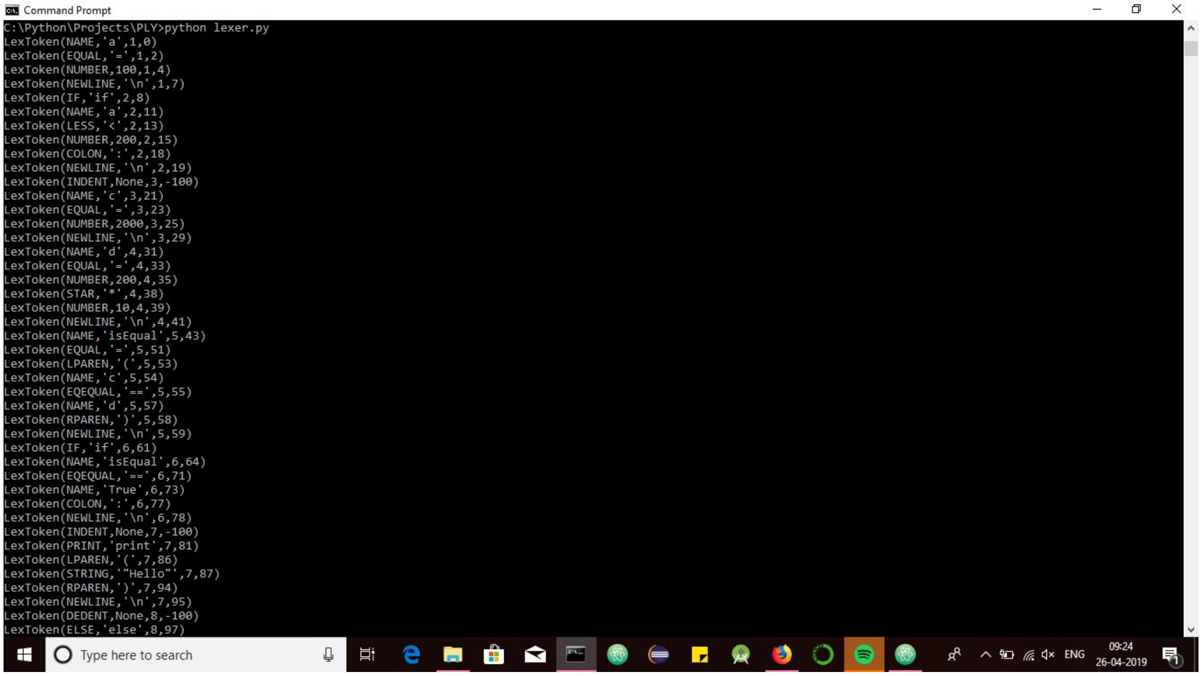
```
a=8
```

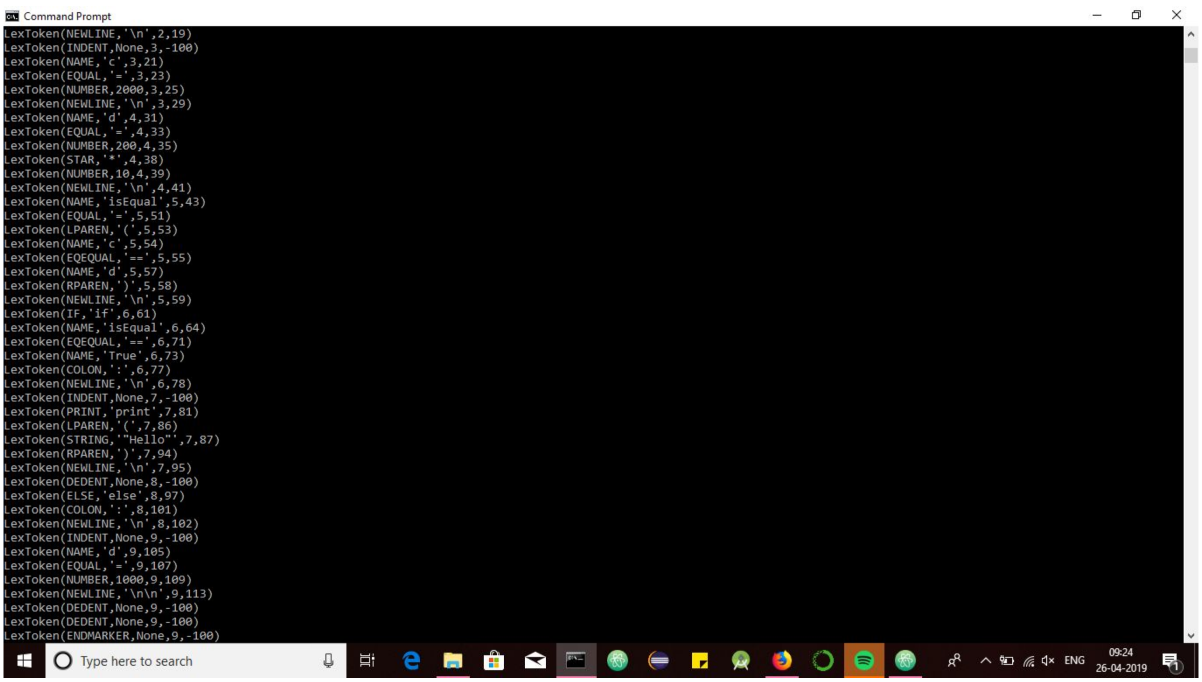
```
i=10

def  fun():
    j=10
print("hi HEllo")
```

LEXICAL ANALYSIS

TEST CODE 1


Command Prompt
C:\Python\Projects\PLY>python lexer.py
LexToken(NAME,'a',1,0)
LexToken(EQUAL,'=',1,2)
LexToken(NUMBER,100,1,4)
LexToken(NEWLINE,'\n',1,7)
LexToken(IF,'if',2,8)
LexToken(NAME,'a',2,11)
LexToken(LESS,'<',2,13)
LexToken(NUMBER,200,2,15)
LexToken(COLON,':',2,18)
LexToken(NEWLINE,'\n',2,19)
LexToken(INDENT,None,3,-100)
LexToken(NAME,'c',3,21)
LexToken(EQUAL,'=',3,23)
LexToken(NUMBER,2000,3,25)
LexToken(NEWLINE,'\n',3,29)
LexToken(NAME,'d',4,31)
LexToken(EQUAL,'=',4,33)
LexToken(NUMBER,200,4,35)
LexToken(STAR,'*',4,38)
LexToken(NUMBER,10,4,39)
LexToken(NEWLINE,'\n',4,41)
LexToken(NAME,'isEqual',5,43)
LexToken(EQUAL,'=',5,51)
LexToken(LPAREN,'(',5,53)
LexToken(NAME,'c',5,54)
LexToken(EQEQUAL,'==',5,55)
LexToken(NAME,'d',5,57)
LexToken(RPAREN,')',5,58)
LexToken(NEWLINE,'\n',5,59)
LexToken(IF,'if',6,61)
LexToken(NAME,'isEqual',6,64)
LexToken(EQEQUAL,'==',6,71)
LexToken(NAME,'True',6,73)
LexToken(COLON,':',6,77)
LexToken(NEWLINE,'\n',6,78)
LexToken(INDENT,None,7,-100)
LexToken(PRINT,'print',7,81)
LexToken(LPAREN,'(',7,86)
LexToken(STRING,'"Hello"',7,87)
LexToken(RPAREN,')',7,94)
LexToken(NEWLINE,'\n',7,95)
LexToken(DEDENT,None,8,-100)
LexToken(ELSE,'else',8,97)
Type here to search
ENG
09:24
26-04-2019


Command Prompt
LexToken(NEWLINE,'\n',2,19)
LexToken(INDENT,None,3,-100)
LexToken(NAME,'c',3,21)
LexToken(EQUAL,'=',3,23)
LexToken(NUMBER,2000,3,25)
LexToken(NEWLINE,'\n',3,29)
LexToken(NAME,'d',4,31)
LexToken(EQUAL,'=',4,33)
LexToken(NUMBER,200,4,35)
LexToken(STAR,'*',4,38)
LexToken(NUMBER,10,4,39)
LexToken(NEWLINE,'\n',4,41)
LexToken(NAME,'isEqual',5,43)
LexToken(EQUAL,'=',5,51)
LexToken(LPAREN,'(',5,53)
LexToken(NAME,'c',5,54)
LexToken(EQEQUAL,'==',5,55)
LexToken(NAME,'d',5,57)
LexToken(RPAREN,')',5,58)
LexToken(NEWLINE,'\n',5,59)
LexToken(IF,'if',6,61)
LexToken(NAME,'isEqual',6,64)
LexToken(EQEQUAL,'==',6,71)
LexToken(NAME,'True',6,73)
LexToken(COLON,':',6,77)
LexToken(NEWLINE,'\n',6,78)
LexToken(INDENT,None,7,-100)
LexToken(PRINT,'print',7,81)
LexToken(LPAREN,'(',7,86)
LexToken(STRING,'"Hello"',7,87)
LexToken(RPAREN,')',7,94)
LexToken(NEWLINE,'\n',7,95)
LexToken(DEDENT,None,8,-100)
LexToken(ELSE,'else',8,97)
LexToken(COLON,':',8,101)
LexToken(NEWLINE,'\n',8,102)
LexToken(INDENT,None,9,-100)
LexToken(NAME,'d',9,105)
LexToken(EQUAL,'=',9,107)
LexToken(NUMBER,1000,9,109)
LexToken(NEWLINE,'\n\n',9,113)
LexToken(DEDENT,None,9,-100)
LexToken(DEDENT,None,9,-100)
LexToken(ENDMARKER,None,9,-100)
Type here to search
ENG
09:24
26-04-2019

TEST CODE 2

Command Prompt

```
C:\Python\Projects\PLY>python irgen.py
LexToken(NAME,'a',1,0)
LexToken(EQUAL,'=',1,1)
LexToken(NUMBER,8,1,2)
LexToken(NEWLINE,'\n\n',1,3)
LexToken(NAME,'i',3,5)
LexToken(EQUAL,'=',3,6)
LexToken(NUMBER,10,3,7)
LexToken(NEWLINE,'\n\n\n\n\n\n\n\n\n\n',3,11)
LexToken(DEF,'def',13,21)
LexToken(NAME,'fun',13,26)
LexToken(LPAREN,'(',13,29)
LexToken(RPAREN,')',13,30)
LexToken(COLON,':',13,31)
LexToken(NEWLINE,'\n\n',13,32)
LexToken(INDENT,None,15,-100)
LexToken(NAME,'j',15,36)
LexToken(EQUAL,'=',15,37)
LexToken(NUMBER,10,15,38)
LexToken(NEWLINE,'\n\n',15,40)
LexToken(DEDENT,None,17,-100)
LexToken(PRINT,'print',17,42)
LexToken(LPAREN,'(',17,47)
LexToken(STRING,'"hi HEllo"',17,48)
LexToken(RPAREN,')',17,58)
LexToken(NEWLINE,'\n\n\n',17,59)
LexToken(ENDMARKER,None,17,-100)
```

SYMBOL TABLE

TEST CODE 1

```
 SYMBOL TABLE
--------------

SCOPE: program
-------------------------
{'False': {'offset': 1, 'program': 0, 'type': 'BOOLEAN', 'width': 1},
 'True': {'offset': 0, 'program': 1, 'type': 'BOOLEAN', 'width': 1},
 'a': {'offset': 2, 'program': 'var1', 'type': 'NUMBER', 'width': 4},
 'c': {'offset': 6, 'program': 'var3', 'type': 'NUMBER', 'width': 4},
 'd': {'offset': 10, 'program': 'var5', 'type': 'NUMBER', 'width': 4},
 'isEqual': {'offset': 14, 'program': 'var7', 'type': 'BOOLEAN', 'width': 1},
 'numParam': 0,
 'returnType': 'UNDEFINED',
 'scopeName': 'program',
 'type': 'FUNCTION',
 'width': 15}
```

TEST CODE 2

```
 SYMBOL TABLE
--------------

SCOPE: fun
-----------------------
{'j': {'fun': 'var4', 'offset': 0, 'type': 'NUMBER', 'width': 4},
 'numParam': 0,
 'parentName': 'program',
 'returnType': 'UNDEFINED',
 'scopeName': 'fun',
 'type': 'FUNCTION',
 'width': 4}
-----------------------


SCOPE: program
-----------------------
{'False': {'fun': 0, 'offset': 15, 'program': 0, 'type': 'BOOLEAN', 'width': 1},
 'True': {'fun': 1, 'offset': 14, 'program': 1, 'type': 'BOOLEAN', 'width': 1},
 'a': {'offset': 2, 'program': 'var1', 'type': 'NUMBER', 'width': 4},
 'fun': {'j': {'fun': 'var4', 'offset': 0, 'type': 'NUMBER', 'width': 4},
         'numParam': 0,
         'parentName': 'program',
         'returnType': 'UNDEFINED',
         'scopeName': 'fun',
         'type': 'FUNCTION',
         'width': 4},
 'i': {'offset': 6, 'program': 'var2', 'type': 'NUMBER', 'width': 4},
 'numParam': 0,
 'returnType': 'UNDEFINED',
 'scopeName': 'program',
 'type': 'FUNCTION',
 'width': 16}
-----------------------
```

INTERMEDIATE CODE GENERATION

TEST CODE 1

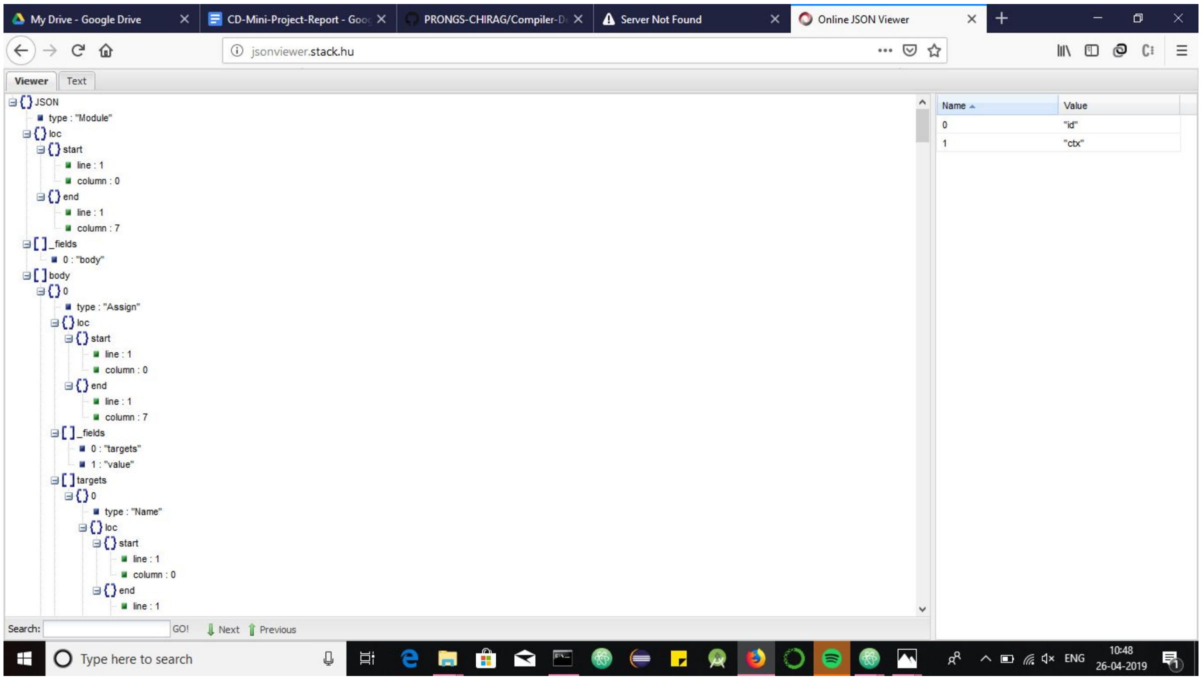
```
program :
    0:   ['var1', 100, '', '=']
    1:   ['var2', 'var1', 200, '<']
    2:   ['var2', 0, 14, 'COND_GOTO']
    3:   ['var3', 2000, '', '=']
    4:   ['var4', 200, 10, '*']
    5:   ['var5', 'var4', '', '=']
    6:   ['var6', 'var3', 'var5', '==']
    7:   ['var7', 'var6', '', '=']
    8:   ['var8', 'var7', 1, '==']
    9:   ['var8', 0, 13, 'COND_GOTO']
   10:   ['"Hello"', '', 'STRING', 'PRINT']
   11:   ['"\n"', '', 'STRING', 'PRINT']
   12:   ['', '', 14, 'GOTO']
   13:   ['var5', 1000, '', '=']
   14:   ['', '', -1, 'HALT']
```

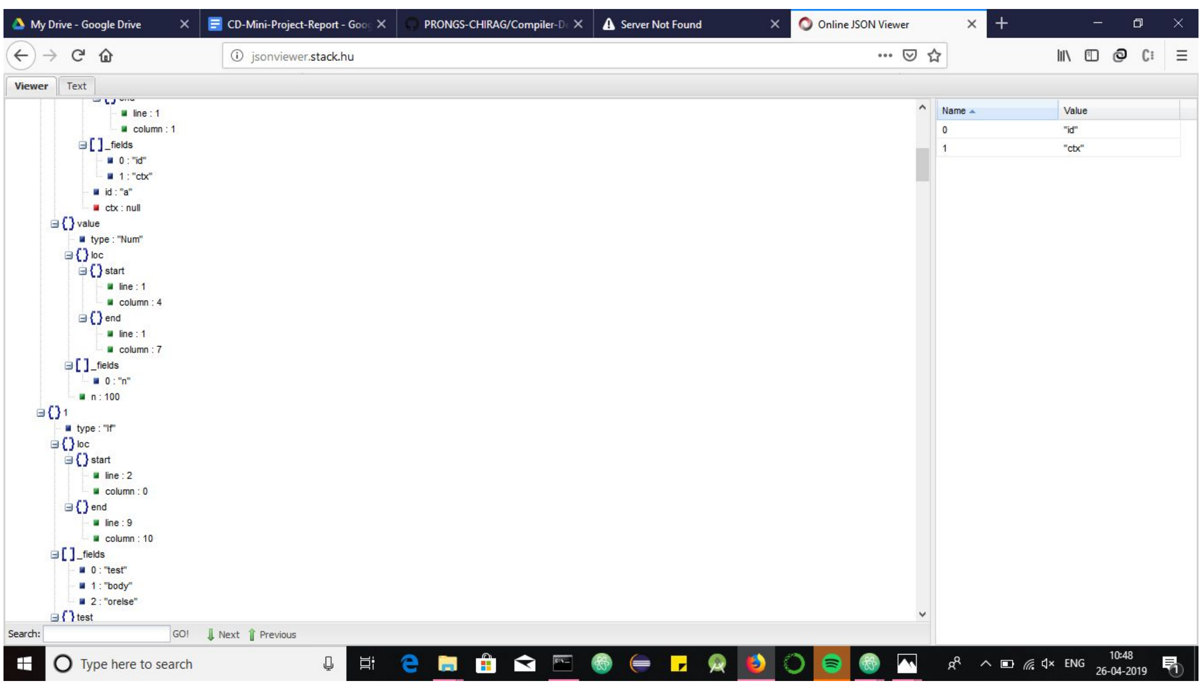
TEST CODE 2

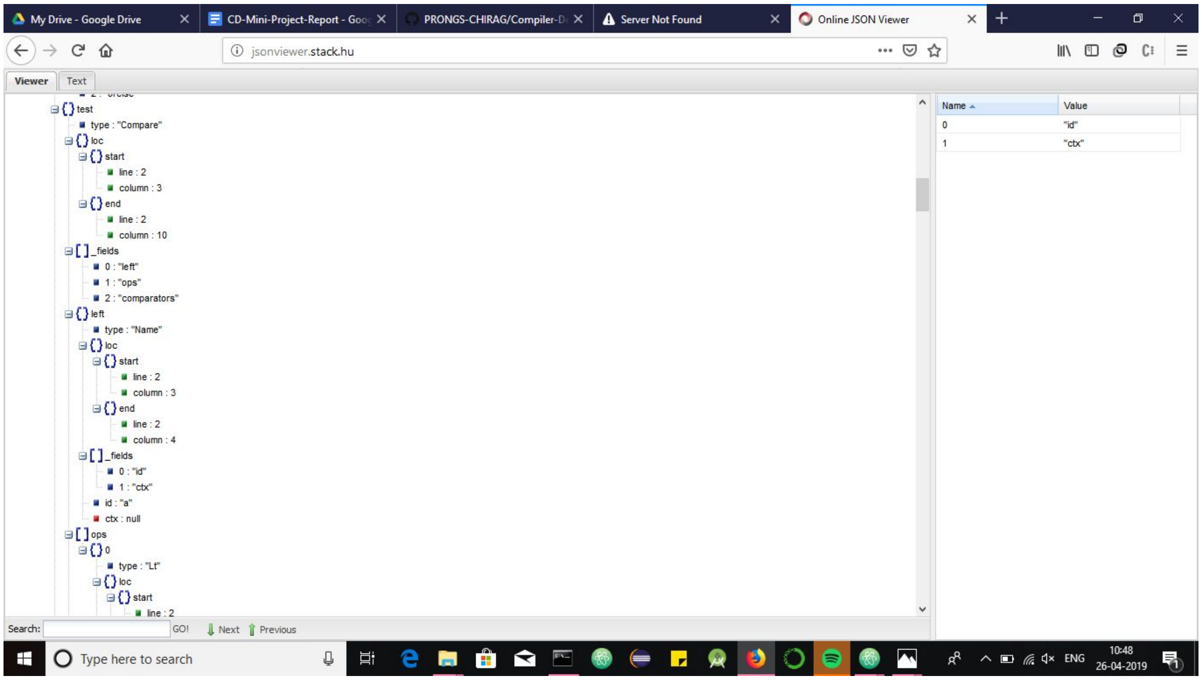
```
program :
    0:   ['var1', 8, '', '=']
    1:   ['var2', 10, '', '=']
    2:   ['"hi HEllo"', '', 'STRING', 'PRINT']
    3:   ['"\n"', '', 'STRING', 'PRINT']
    4:   ['', '', -1, 'HALT']
fun :
    0:   ['var4', 10, '', '=']
    1:   ['', '', '', 'JUMP_RETURN']
```

ABSTRACT SYNTAX TREE

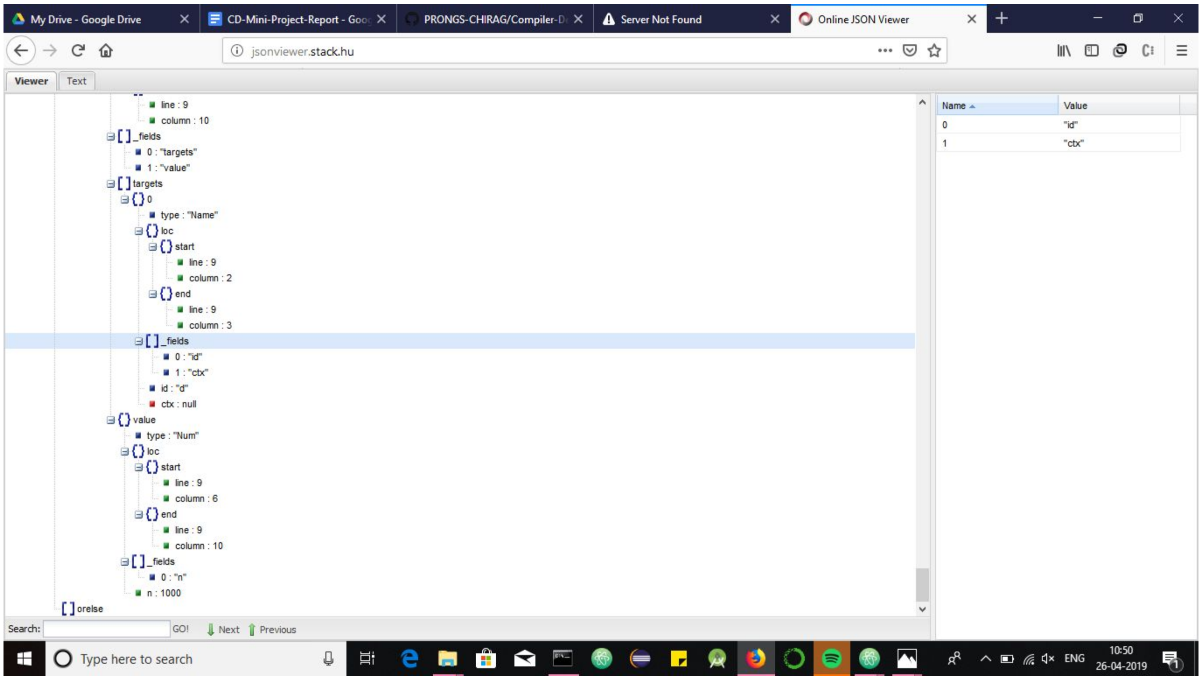
TEST CODE 1


My Drive - Google Drive
CD-Mini-Project-Report - Goo
PRONGS-CHIRAG/Compiler-D
Server Not Found
Online JSON Viewer
jsonviewer.stack.hu
Viewer
Text
JSON
type : "Module"
loc
start
line : 1
column : 0
end
line : 1
column : 7
_fields
0 : "body"
body
0
type : "Assign"
loc
start
line : 1
column : 0
end
line : 1
column : 7
_fields
0 : "targets"
1 : "value"
targets
0
type : "Name"
loc
start
line : 1
column : 0
end
line : 1
Name
Value
0
"id"
1
"ctx"
Search:
GO!
Next
Previous
Type here to search
ENG
10:48
26-04-2019

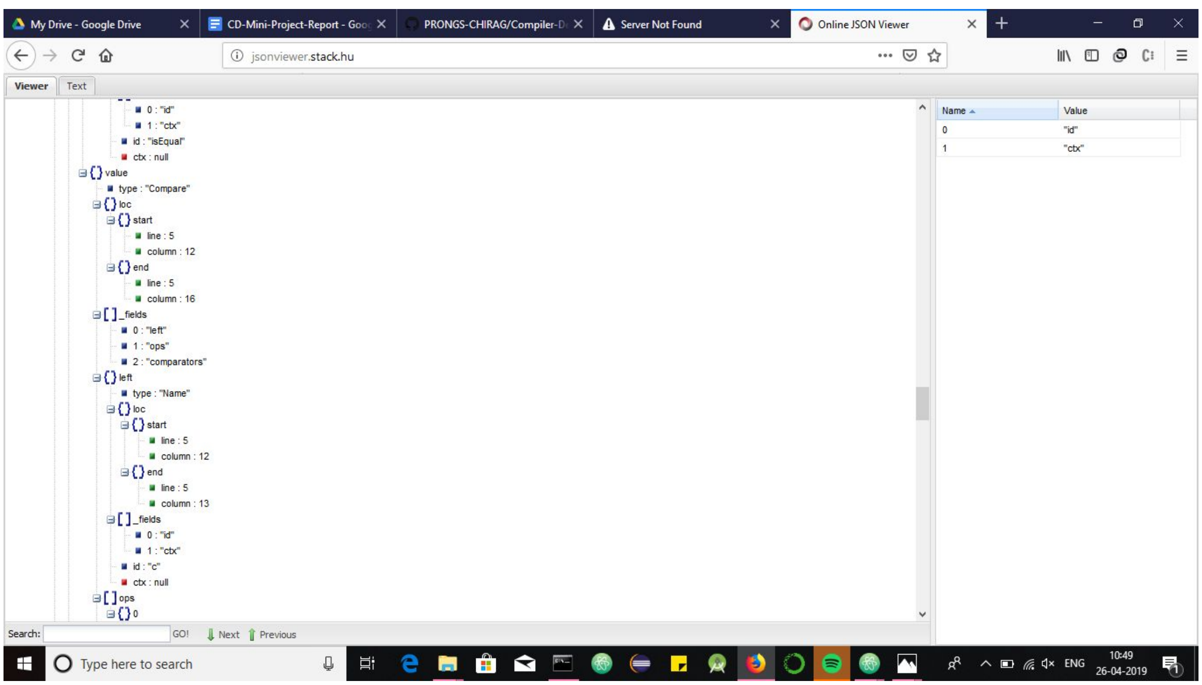

My Drive - Google Drive
CD-Mini-Project-Report - Goo
PRONGS-CHIRAG/Compiler-D
Server Not Found
Online JSON Viewer
jsonviewer.stack.hu
Viewer
Text
line : 1
column : 1
_fields
0 : "id"
1 : "ctx"
id : "a"
ctx : null
value
type : "Num"
loc
start
line : 1
column : 4
end
line : 1
column : 7
_fields
0 : "n"
n : 100
1
type : "If"
loc
start
line : 2
column : 0
end
line : 9
column : 10
_fields
0 : "test"
1 : "body"
2 : "orelse"
test
Name
Value
0
"id"
1
"ctx"
Search:
GO!
Next
Previous
Type here to search
ENG
10:48
26-04-2019



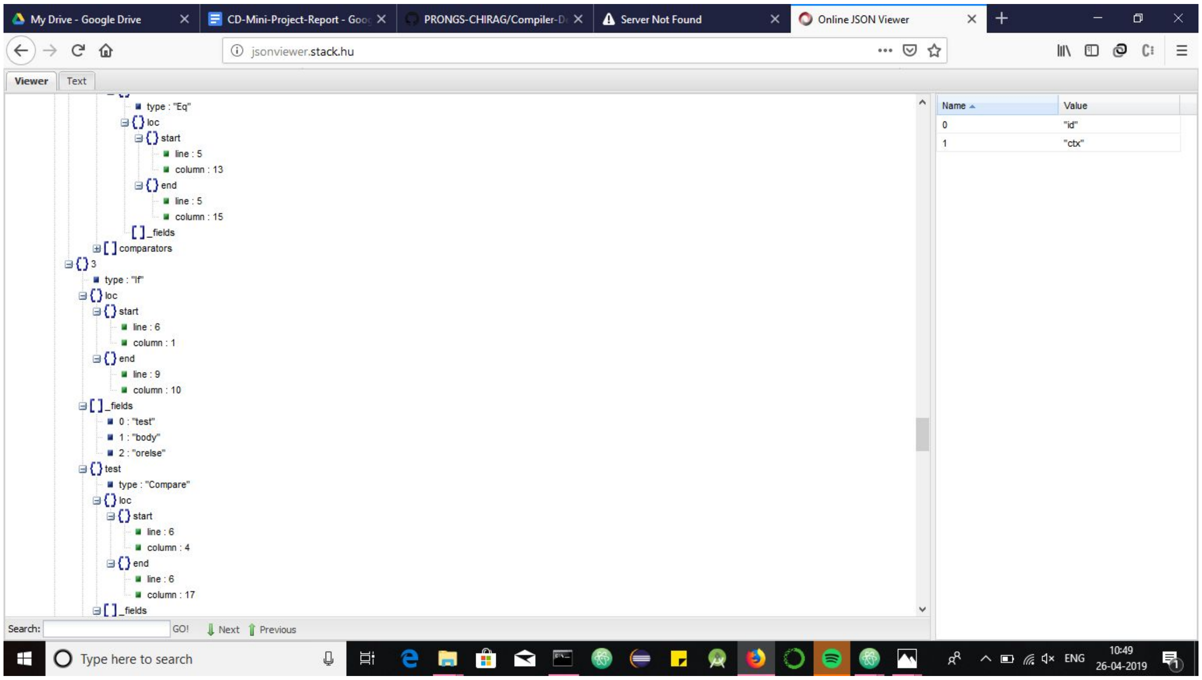
My Drive - Google Drive
CD-Mini-Project-Report - Goo
PRONGS-CHIRAG/Compiler-D
Server Not Found
Online JSON Viewer
jsonviewer.stack.hu
Viewer
Text
test
type : "Compare"
loc
start
line : 2
column : 3
end
line : 2
column : 10
_fields
0 : "left"
1 : "ops"
2 : "comparators"
left
type : "Name"
loc
start
line : 2
column : 3
end
line : 2
column : 4
_fields
0 : "id"
1 : "ctx"
id : "a"
ctx : null
ops
0
type : "Lt"
loc
start
line : 2
Name
Value
0
"id"
1
"ctx"
Search:
GO!
Next
Previous
Type here to search
ENG
10:48
26-04-2019



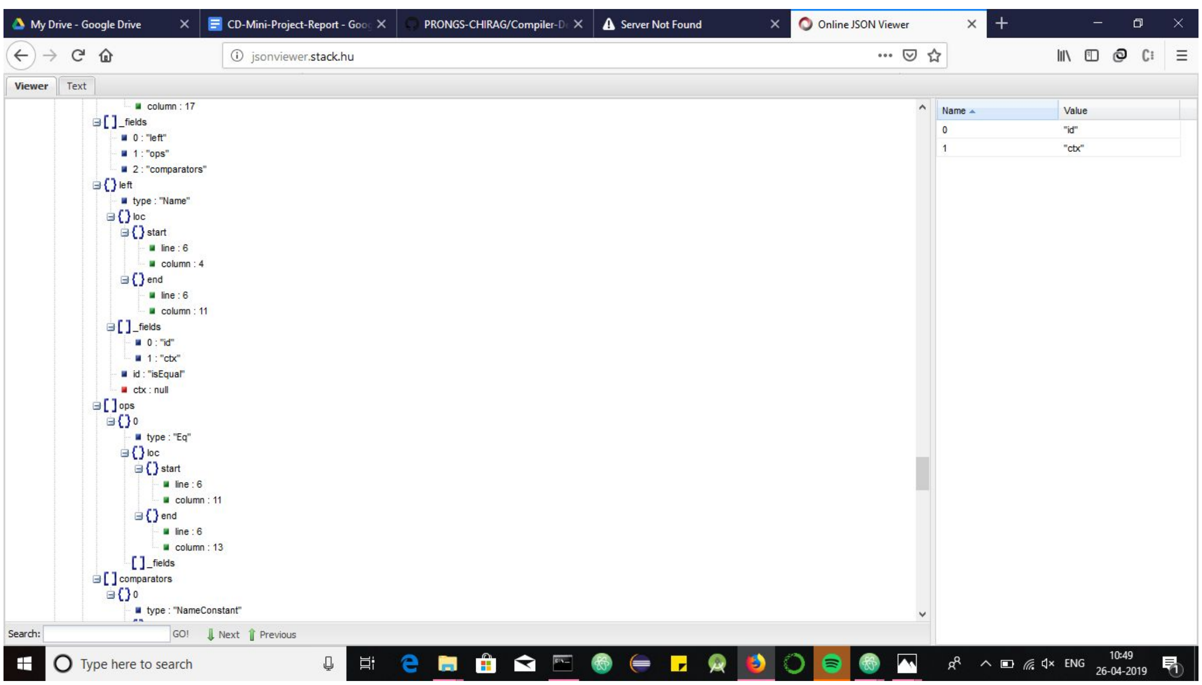
My Drive - Google Drive
CD-Mini-Project-Report - Goo
PRONGS-CHIRAG/Compiler-D
Server Not Found
Online JSON Viewer
jsonviewer.stack.hu
Viewer
Text
line : 9
column : 10
[ ] _fields
0 : "targets"
1 : "value"
[ ] targets
{ } 0
type : "Name"
{ } loc
{ } start
line : 9
column : 2
{ } end
line : 9
column : 3
[ ] _fields
0 : "id"
1 : "ctx"
id : "d"
ctx : null
{ } value
type : "Num"
{ } loc
{ } start
line : 9
column : 6
{ } end
line : 9
column : 10
[ ] _fields
0 : "n"
n : 1000
[ ] orelse
Name
Value
0
"id"
1
"ctx"
Search:
GO!
Next
Previous
Type here to search
ENG
10:50
26-04-2019



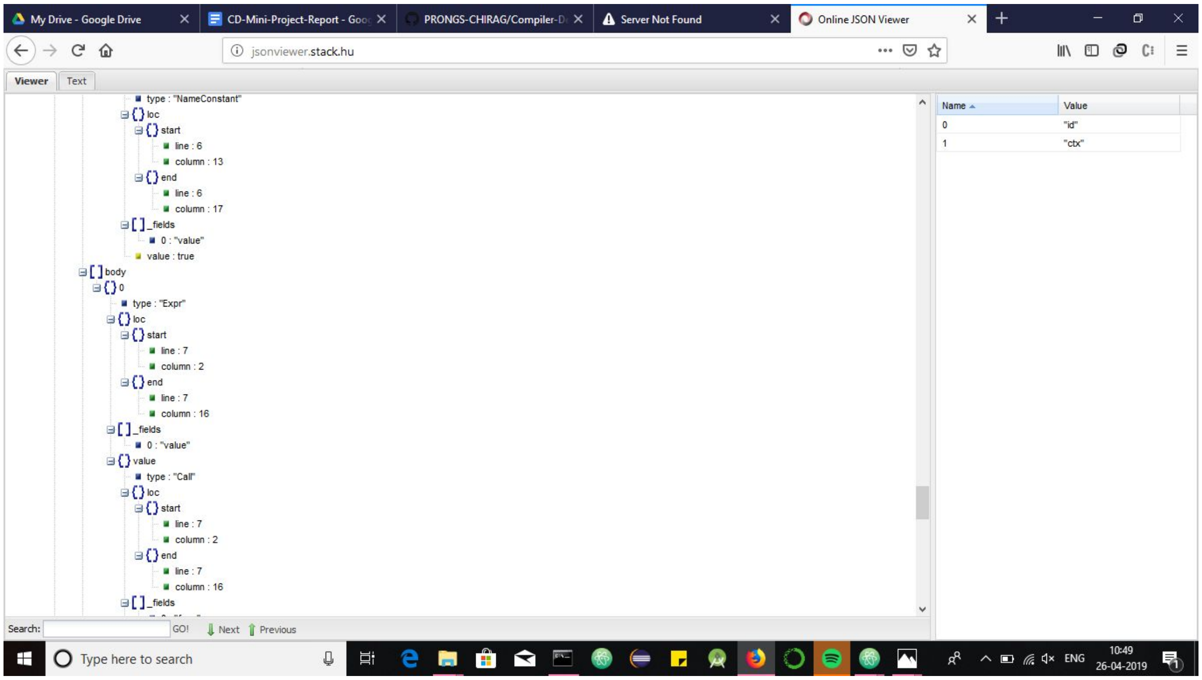
My Drive - Google Drive
CD-Mini-Project-Report - Goo
PRONGS-CHIRAG/Compiler-D
Server Not Found
Online JSON Viewer
jsonviewer.stack.hu
Viewer
Text
0 : "id"
1 : "ctx"
id : "isEqual"
ctx : null
{ } value
type : "Compare"
{ } loc
{ } start
line : 5
column : 12
{ } end
line : 5
column : 16
[ ] _fields
0 : "left"
1 : "ops"
2 : "comparators"
{ } left
type : "Name"
{ } loc
{ } start
line : 5
column : 12
{ } end
line : 5
column : 13
[ ] _fields
0 : "id"
1 : "ctx"
id : "c"
ctx : null
[ ] ops
{ } 0
Name
Value
0
"id"
1
"ctx"
Search:
GO!
Next
Previous
Type here to search
ENG
10:49
26-04-2019

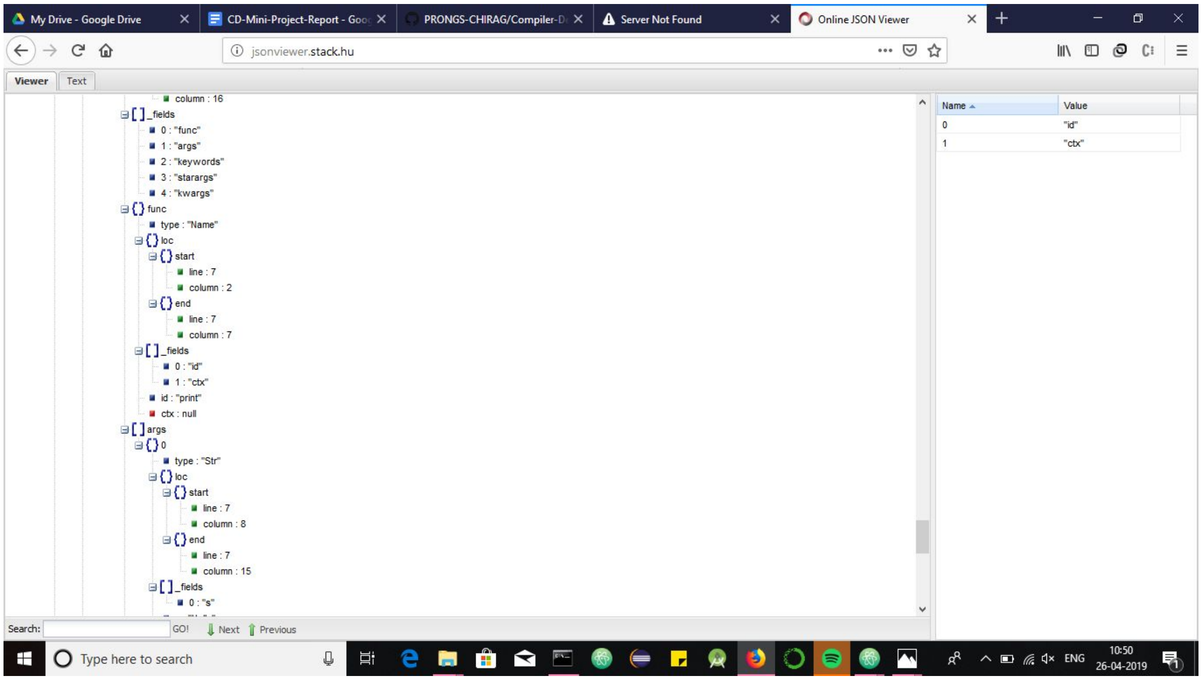


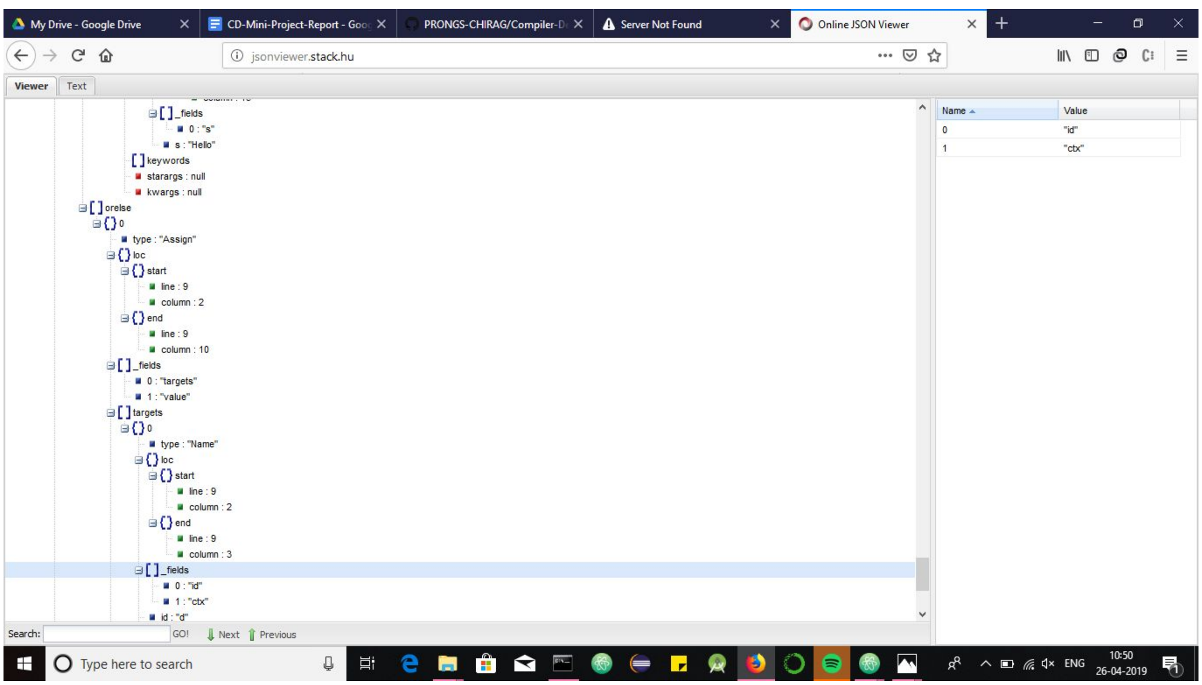
My Drive - Google Drive
CD-Mini-Project-Report - Goo
PRONGS-CHIRAG/Compiler-D
Server Not Found
Online JSON Viewer
jsonviewer.stack.hu
Viewer
Text
type : "Eq"
loc
start
line : 5
column : 13
end
line : 5
column : 15
_fields
comparators
3
type : "If"
loc
start
line : 6
column : 1
end
line : 9
column : 10
_fields
0 : "test"
1 : "body"
2 : "orelse"
test
type : "Compare"
loc
start
line : 6
column : 4
end
line : 6
column : 17
_fields
Name
Value
0
"id"
1
"ctx"
Search:
GO!
Next
Previous
Type here to search
ENG
10:49
26-04-2019

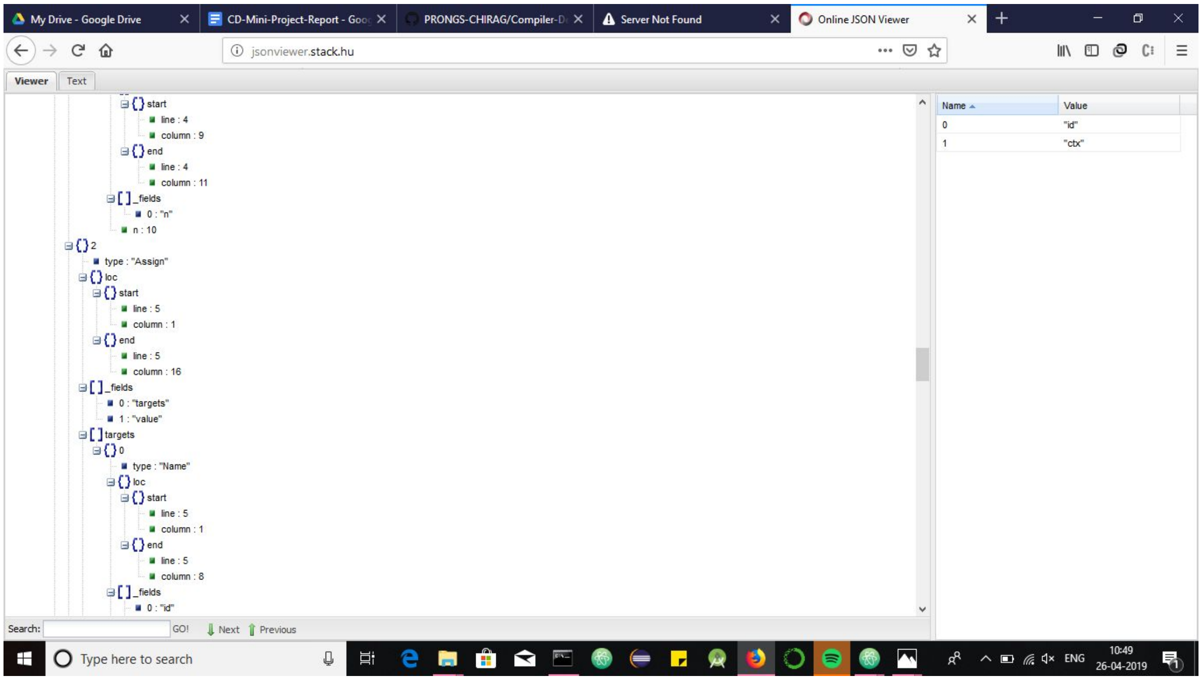


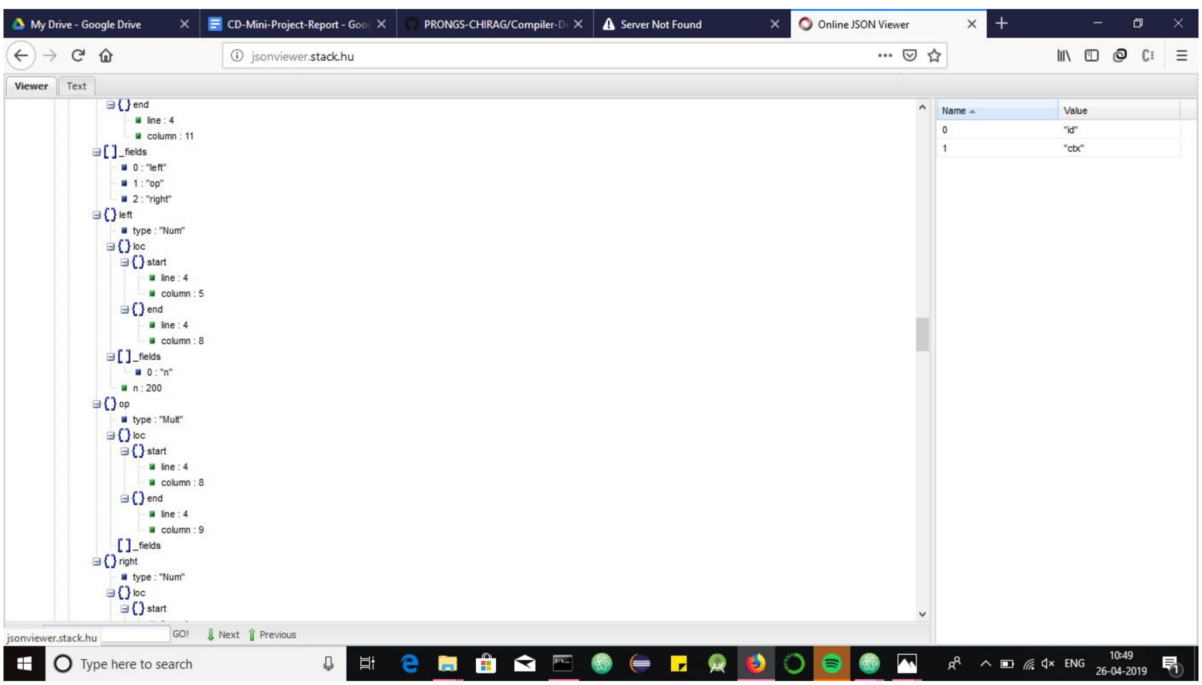
My Drive - Google Drive
CD-Mini-Project-Report - Goo
PRONGS-CHIRAG/Compiler-D
Server Not Found
Online JSON Viewer
jsonviewer.stack.hu
Viewer
Text
column : 17
_fields
0 : "left"
1 : "ops"
2 : "comparators"
left
type : "Name"
loc
start
line : 6
column : 4
end
line : 6
column : 11
_fields
0 : "id"
1 : "ctx"
id : "isEqual"
ctx : null
ops
0
type : "Eq"
loc
start
line : 6
column : 11
end
line : 6
column : 13
_fields
comparators
0
type : "NameConstant"
Name
Value
0
"id"
1
"ctx"
Search:
GO!
Next
Previous
Type here to search
ENG
10:49
26-04-2019

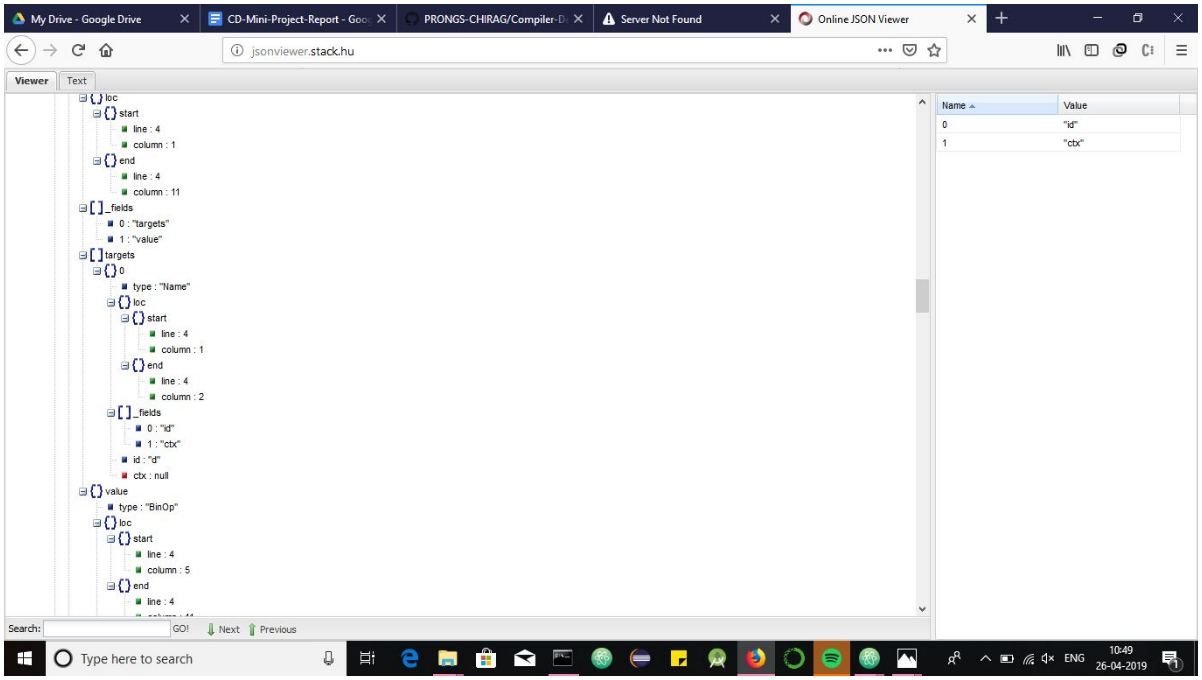

My Drive - Google Drive
CD-Mini-Project-Report - Goo
PRONGS-CHIRAG/Compiler-D
Server Not Found
Online JSON Viewer
jsonviewer.stack.hu
Viewer
Text
type : "NameConstant"
loc
start
line : 6
column : 13
end
line : 6
column : 17
_fields
0 : "value"
value : true
body
0
type : "Expr"
loc
start
line : 7
column : 2
end
line : 7
column : 16
_fields
0 : "value"
value
type : "Call"
loc
start
line : 7
column : 2
end
line : 7
column : 16
_fields
Name
Value
0
"id"
1
"ctx"
Search:
GO!
Next
Previous
Type here to search
ENG
10:49
26-04-2019

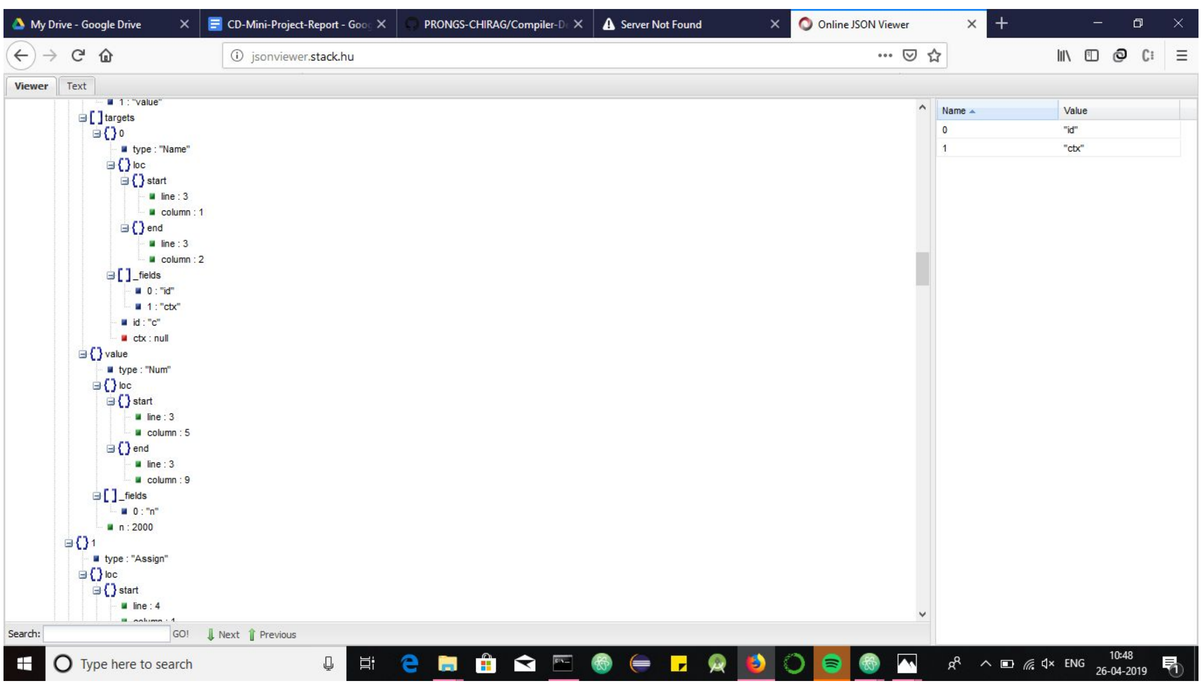

My Drive - Google Drive
CD-Mini-Project-Report - Goo
PRONGS-CHIRAG/Compiler-D
Server Not Found
Online JSON Viewer
jsonviewer.stack.hu
Viewer
Text
column : 16
[ ] _fields
0 : "func"
1 : "args"
2 : "keywords"
3 : "starargs"
4 : "kwargs"
{ } func
type : "Name"
{ } loc
{ } start
line : 7
column : 2
{ } end
line : 7
column : 7
[ ] _fields
0 : "id"
1 : "ctx"
id : "print"
ctx : null
[ ] args
{ } 0
type : "Str"
{ } loc
{ } start
line : 7
column : 8
{ } end
line : 7
column : 15
[ ] _fields
0 : "s"
Name
Value
0
"id"
1
"ctx"
Search:
GO!
Next
Previous
Type here to search
ENG
10:50
26-04-2019

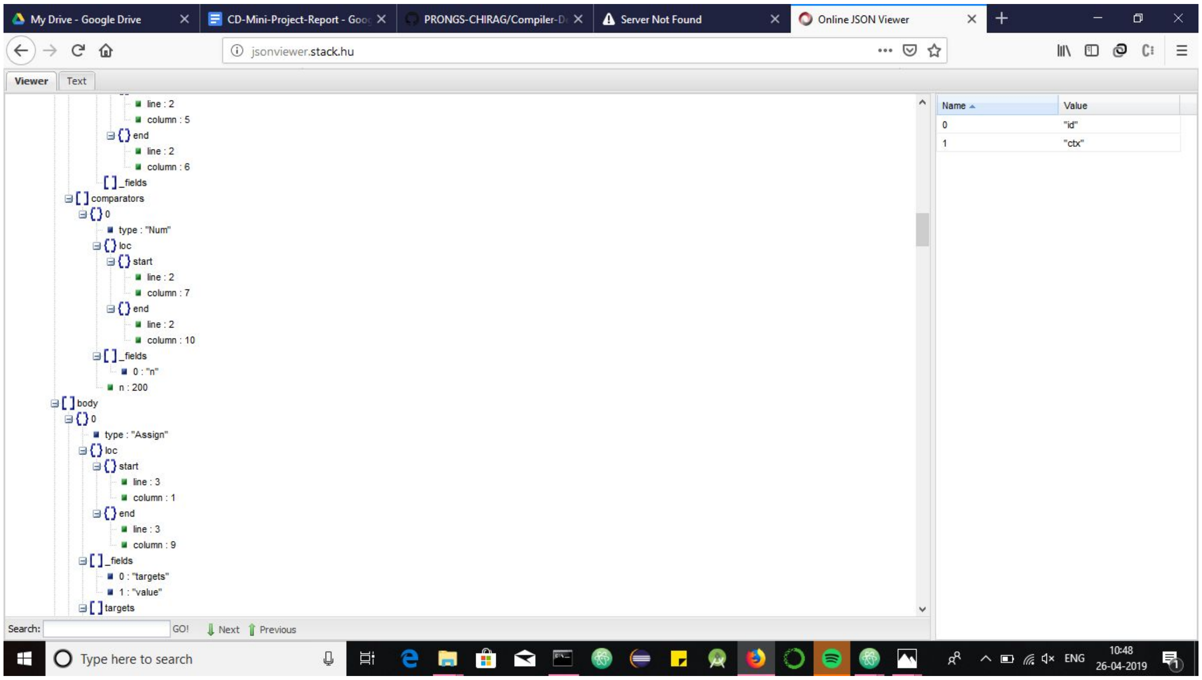

My Drive - Google Drive
CD-Mini-Project-Report - Goo
PRONGS-CHIRAG/Compiler-D
Server Not Found
Online JSON Viewer
jsonviewer.stack.hu
Viewer
Text
[ ] _fields
0 : "s"
s : "Hello"
[ ] keywords
starargs : null
kwargs : null
[ ] orelse
{ } 0
type : "Assign"
{ } loc
{ } start
line : 9
column : 2
{ } end
line : 9
column : 10
[ ] _fields
0 : "targets"
1 : "value"
[ ] targets
{ } 0
type : "Name"
{ } loc
{ } start
line : 9
column : 2
{ } end
line : 9
column : 3
[ ] _fields
0 : "id"
1 : "ctx"
id : "d"
Name
Value
0
"id"
1
"ctx"
Search:
GO!
Next
Previous
Type here to search
ENG
10:50
26-04-2019


My Drive - Google Drive
CD-Mini-Project-Report - Goo
PRONGS-CHIRAG/Compiler-D
Server Not Found
Online JSON Viewer
jsonviewer.stack.hu
Viewer
Text
{} start
line : 4
column : 9
{} end
line : 4
column : 11
[] _fields
0 : "n"
n : 10
{} 2
type : "Assign"
{} loc
{} start
line : 5
column : 1
{} end
line : 5
column : 16
[] _fields
0 : "targets"
1 : "value"
[] targets
{} 0
type : "Name"
{} loc
{} start
line : 5
column : 1
{} end
line : 5
column : 8
[] _fields
0 : "id"
Search:
GO!
Next
Previous
Name
Value
0
"id"
1
"ctx"
Type here to search
ENG
10:49
26-04-2019

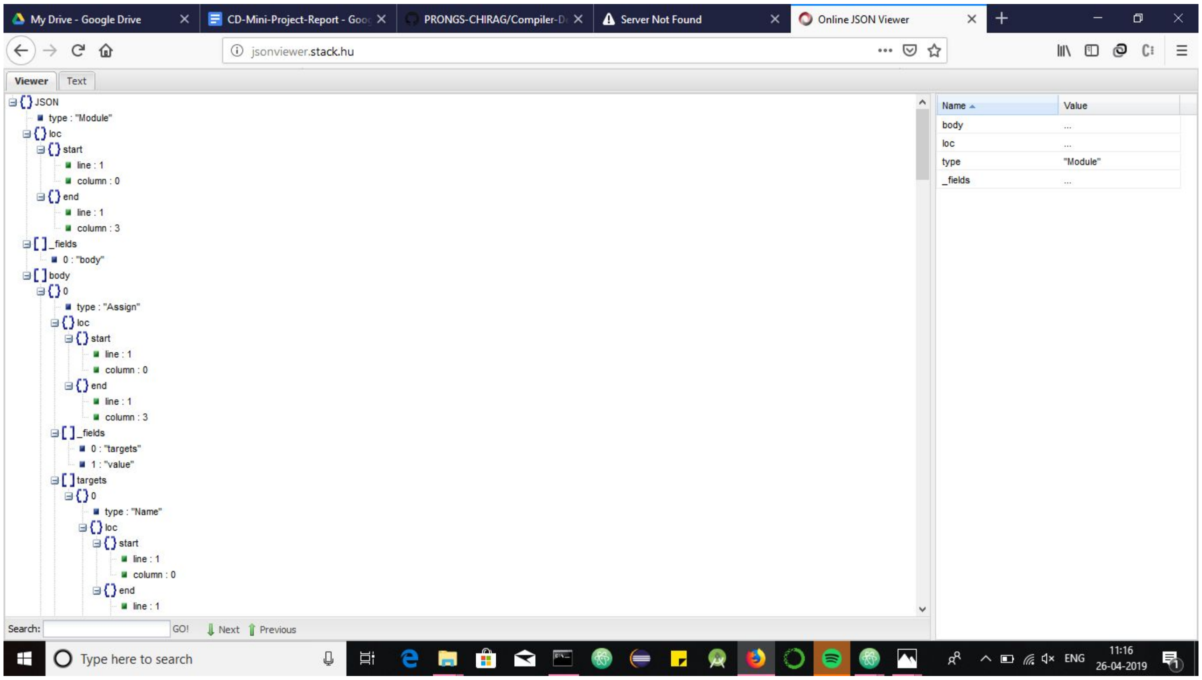

My Drive - Google Drive
CD-Mini-Project-Report - Goo
PRONGS-CHIRAG/Compiler-D
Server Not Found
Online JSON Viewer
jsonviewer.stack.hu
Viewer
Text
{} end
line : 4
column : 11
[] _fields
0 : "left"
1 : "op"
2 : "right"
{} left
type : "Num"
{} loc
{} start
line : 4
column : 5
{} end
line : 4
column : 8
[] _fields
0 : "n"
n : 200
{} op
type : "Mult"
{} loc
{} start
line : 4
column : 8
{} end
line : 4
column : 9
[] _fields
{} right
type : "Num"
{} loc
{} start
jsonviewer.stack.hu
GO!
Next
Previous
Name
Value
0
"id"
1
"ctx"
Type here to search
ENG
10:49
26-04-2019

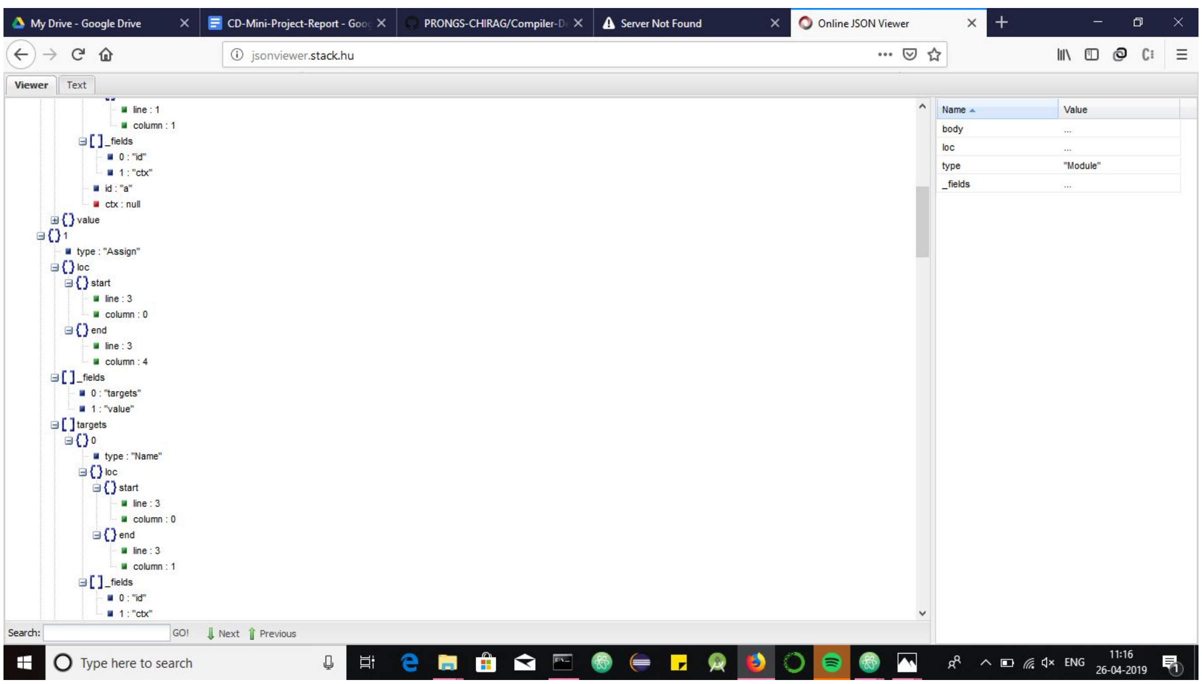

My Drive - Google Drive
CD-Mini-Project-Report - Goo
PRONGS-CHIRAG/Compiler-D
Server Not Found
Online JSON Viewer
jsonviewer.stack.hu
Viewer
Text
loc
start
line : 4
column : 1
end
line : 4
column : 11
_fields
0 : "targets"
1 : "value"
targets
0
type : "Name"
loc
start
line : 4
column : 1
end
line : 4
column : 2
_fields
0 : "id"
1 : "ctx"
id : "d"
ctx : null
value
type : "BinOp"
loc
start
line : 4
column : 5
end
line : 4
Name
Value
0
"id"
1
"ctx"
Search:
GO!
Next
Previous
Type here to search
ENG
10:49
26-04-2019

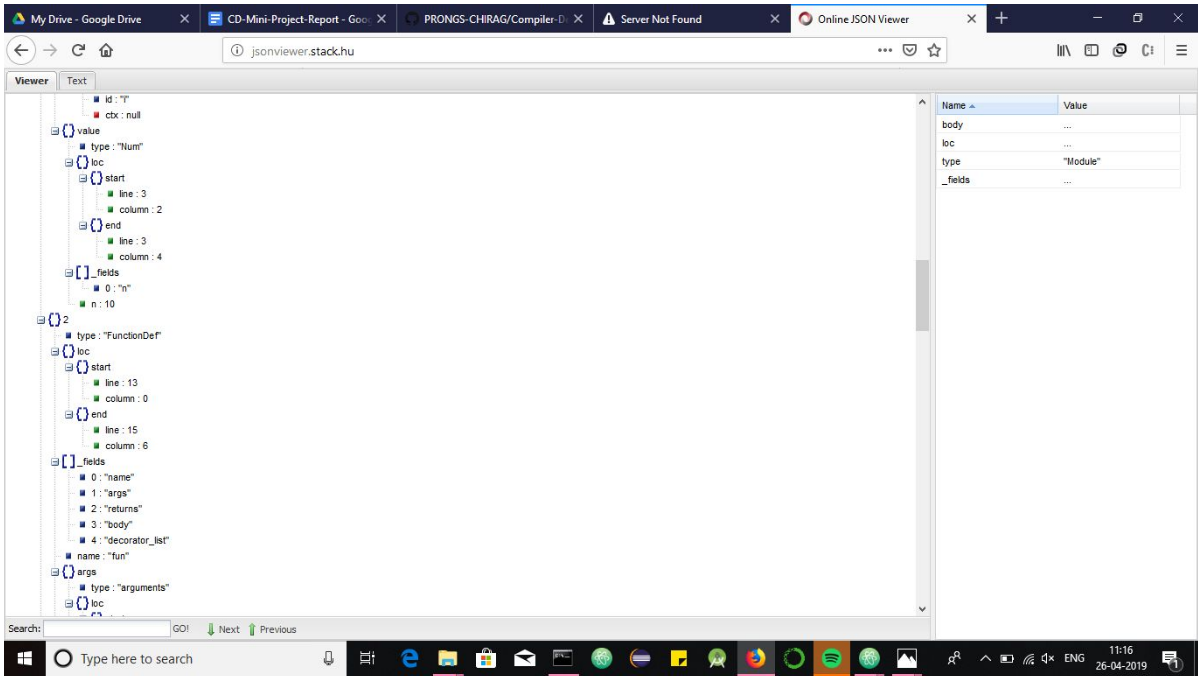

My Drive - Google Drive
CD-Mini-Project-Report - Goo
PRONGS-CHIRAG/Compiler-D
Server Not Found
Online JSON Viewer
jsonviewer.stack.hu
Viewer
Text
1 : "value"
targets
0
type : "Name"
loc
start
line : 3
column : 1
end
line : 3
column : 2
_fields
0 : "id"
1 : "ctx"
id : "c"
ctx : null
value
type : "Num"
loc
start
line : 3
column : 5
end
line : 3
column : 9
_fields
0 : "n"
n : 2000
1
type : "Assign"
loc
start
line : 4
Name
Value
0
"id"
1
"ctx"
Search:
GO!
Next
Previous
Type here to search
ENG
10:48
26-04-2019

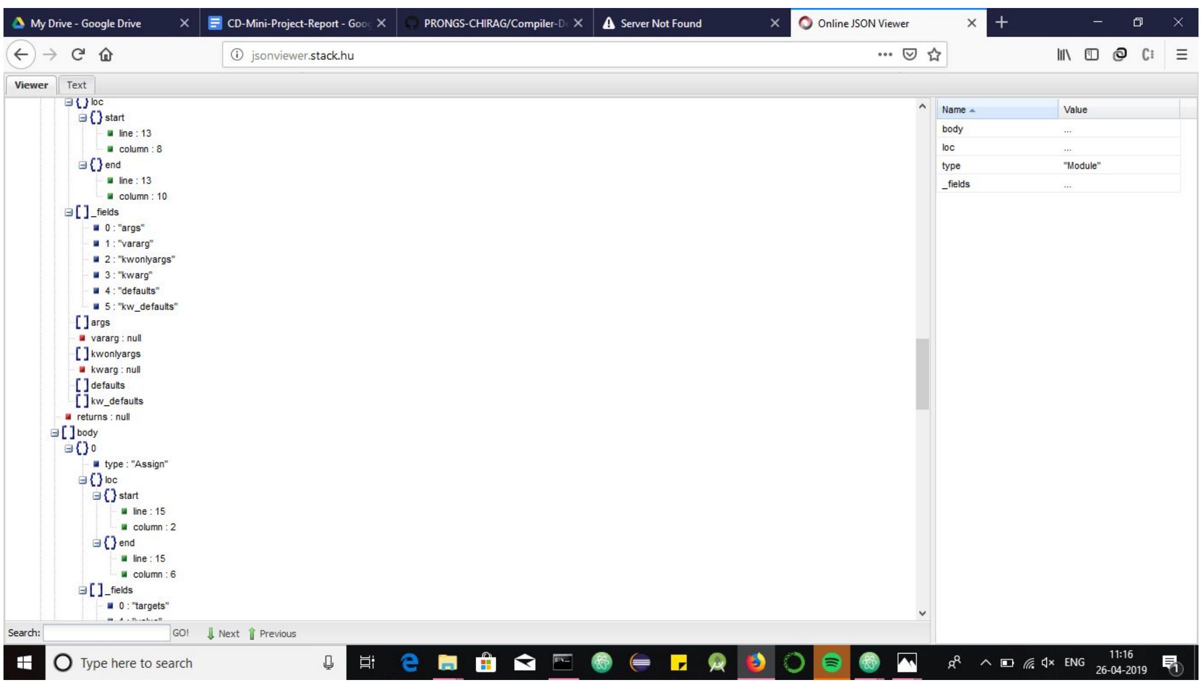

My Drive - Google Drive
CD-Mini-Project-Report - Goo
PRONGS-CHIRAG/Compiler-D
Server Not Found
Online JSON Viewer
jsonviewer.stack.hu
Viewer
Text
line : 2
column : 5
end
line : 2
column : 6
_fields
comparators
0
type : "Num"
loc
start
line : 2
column : 7
end
line : 2
column : 10
_fields
0 : "n"
n : 200
body
0
type : "Assign"
loc
start
line : 3
column : 1
end
line : 3
column : 9
_fields
0 : "targets"
1 : "value"
targets
Name
Value
0
"id"
1
"ctx"
Search:
GO!
Next
Previous
Type here to search
ENG
10:48
26-04-2019

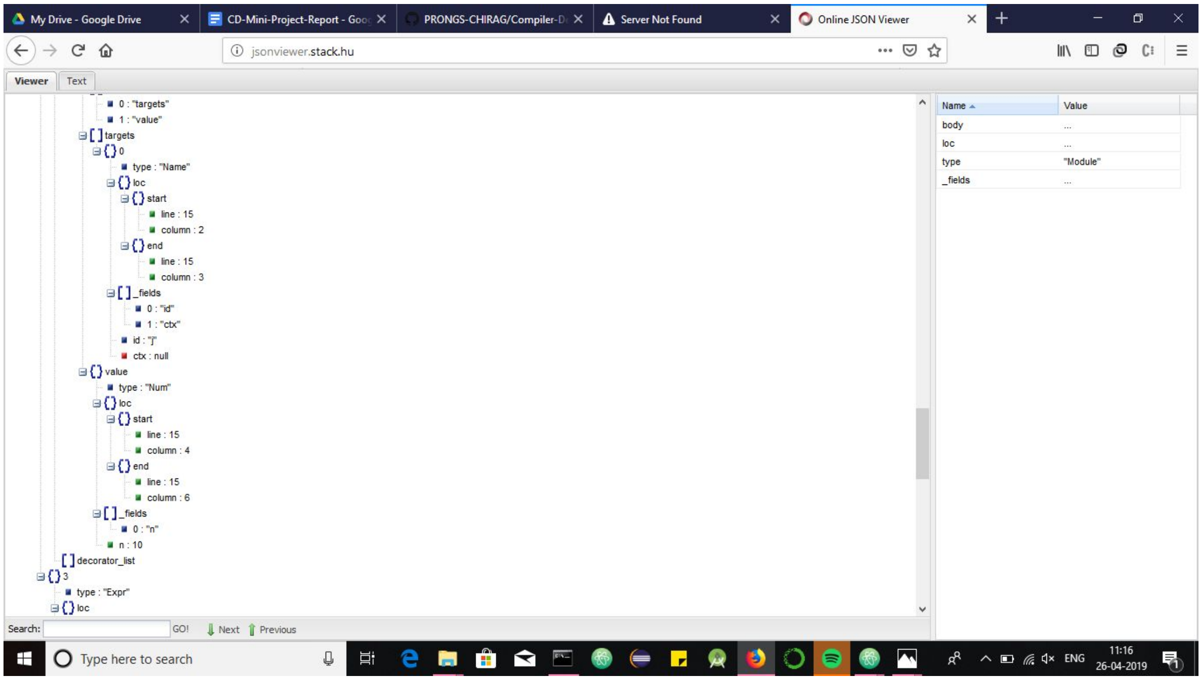
TEST CODE 2

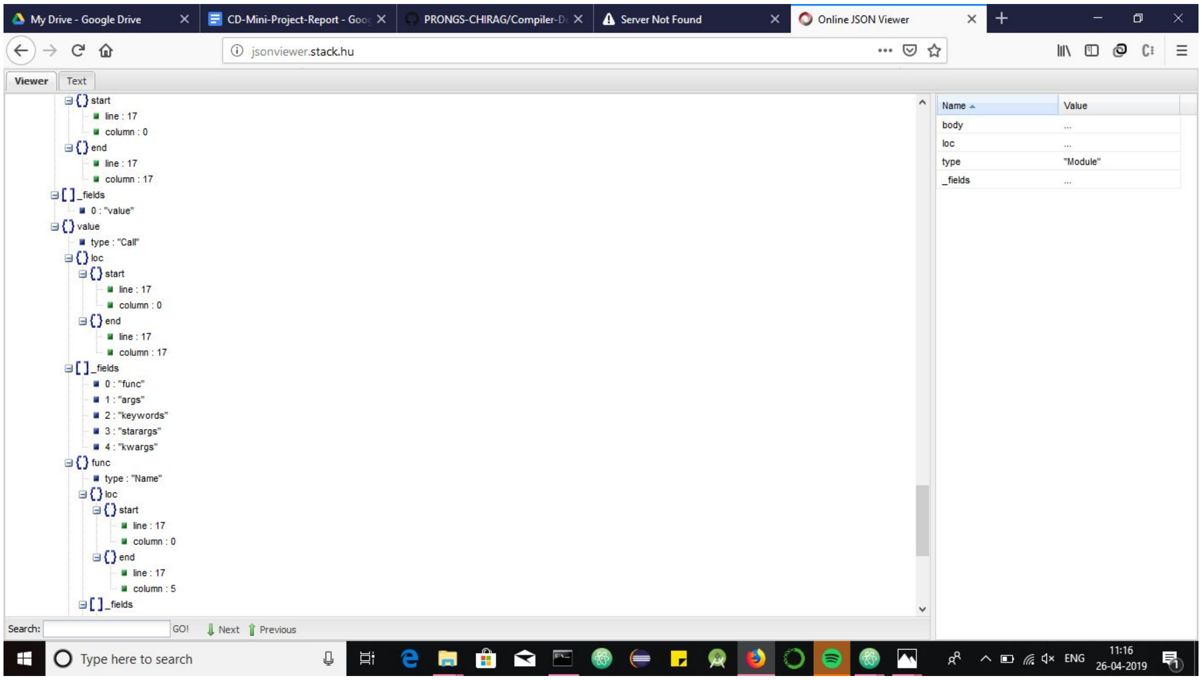

My Drive - Google Drive
CD-Mini-Project-Report - Goo
PRONGS-CHIRAG/Compiler-D
Server Not Found
Online JSON Viewer
jsonviewer.stack.hu
Viewer
Text
JSON
type : "Module"
loc
start
line : 1
column : 0
end
line : 1
column : 3
_fields
0 : "body"
body
0
type : "Assign"
loc
start
line : 1
column : 0
end
line : 1
column : 3
_fields
0 : "targets"
1 : "value"
targets
0
type : "Name"
loc
start
line : 1
column : 0
end
line : 1
Name
Value
body
...
loc
...
type
"Module"
_fields
...
Search:
GO!
Next
Previous
Type here to search
ENG
11:16
26-04-2019

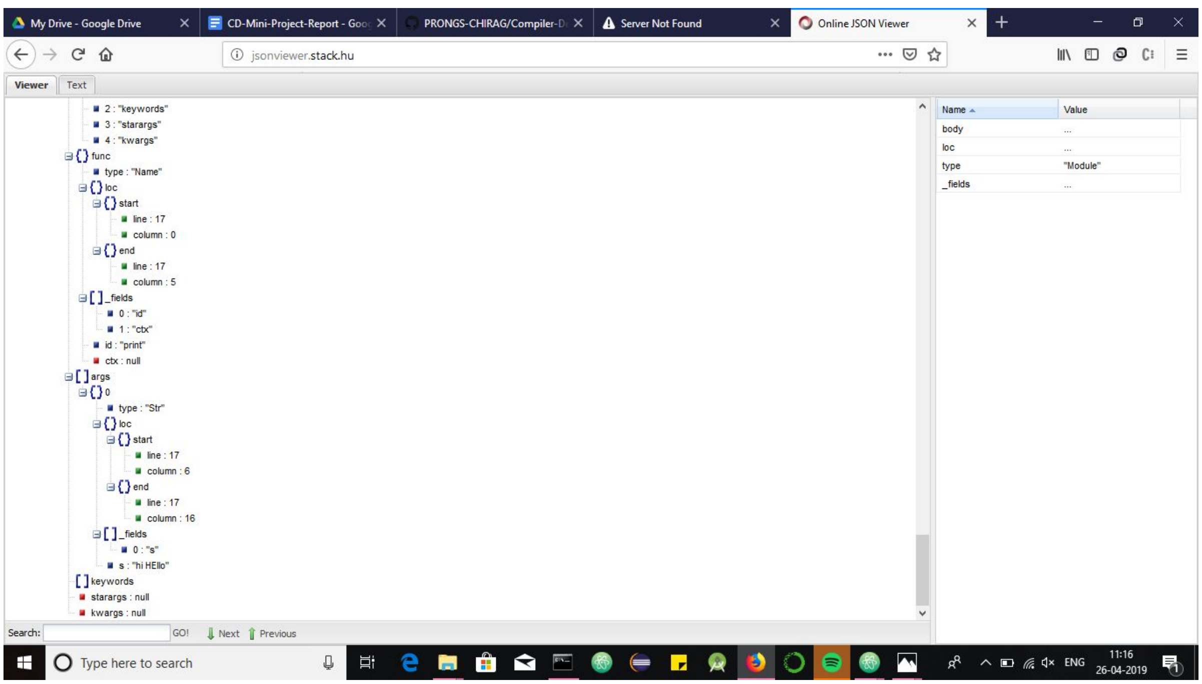

My Drive - Google Drive
CD-Mini-Project-Report - Goo
PRONGS-CHIRAG/Compiler-D
Server Not Found
Online JSON Viewer
jsonviewer.stack.hu
Viewer
Text
line : 1
column : 1
_fields
0 : "id"
1 : "ctx"
id : "a"
ctx : null
value
1
type : "Assign"
loc
start
line : 3
column : 0
end
line : 3
column : 4
_fields
0 : "targets"
1 : "value"
targets
0
type : "Name"
loc
start
line : 3
column : 0
end
line : 3
column : 1
_fields
0 : "id"
1 : "ctx"
Name
Value
body
...
loc
...
type
"Module"
_fields
...
Search:
GO!
Next
Previous
Type here to search
ENG
11:16
26-04-2019


My Drive - Google Drive
CD-Mini-Project-Report - Goo
PRONGS-CHIRAG/Compiler-D
Server Not Found
Online JSON Viewer
jsonviewer.stack.hu
Viewer
Text
id : "i"
ctx : null
value
type : "Num"
loc
start
line : 3
column : 2
end
line : 3
column : 4
_fields
0 : "n"
n : 10
2
type : "FunctionDef"
loc
start
line : 13
column : 0
end
line : 15
column : 6
_fields
0 : "name"
1 : "args"
2 : "returns"
3 : "body"
4 : "decorator_list"
name : "fun"
args
type : "arguments"
loc
Name
Value
body
...
loc
...
type
"Module"
_fields
...
Search:
GO!
Next
Previous
Type here to search
ENG
11:16
26-04-2019

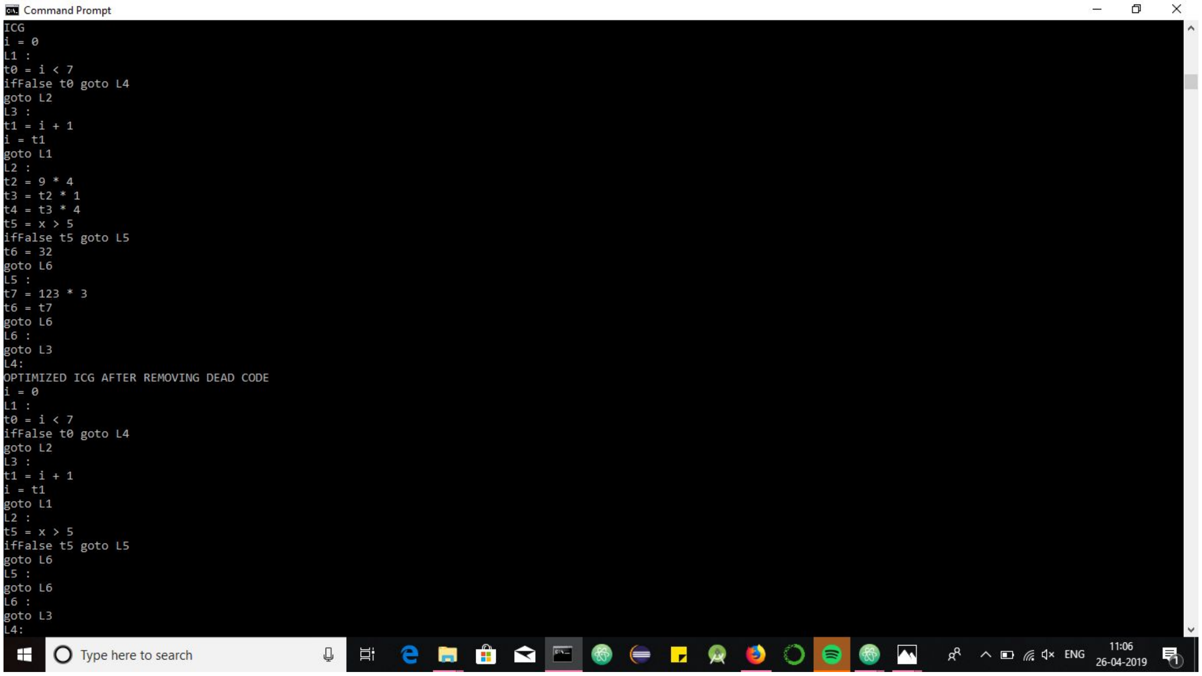

My Drive - Google Drive
CD-Mini-Project-Report - Goo
PRONGS-CHIRAG/Compiler-D
Server Not Found
Online JSON Viewer
jsonviewer.stack.hu
Viewer
Text
loc
start
line : 13
column : 8
end
line : 13
column : 10
_fields
0 : "args"
1 : "vararg"
2 : "kwonlyargs"
3 : "kwarg"
4 : "defaults"
5 : "kw_defaults"
args
vararg : null
kwonlyargs
kwarg : null
defaults
kw_defaults
returns : null
body
0
type : "Assign"
loc
start
line : 15
column : 2
end
line : 15
column : 6
_fields
0 : "targets"
Name
Value
body
...
loc
...
type
"Module"
_fields
...
Search:
GO!
Next
Previous
Type here to search
ENG
11:16
26-04-2019

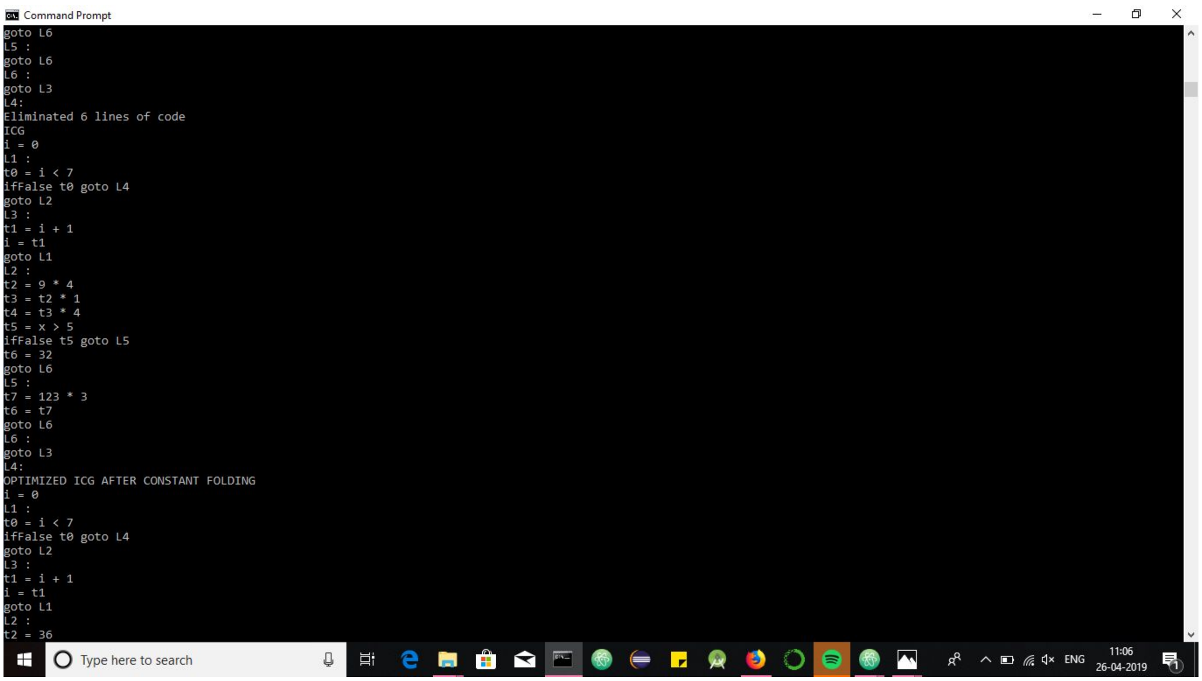

My Drive - Google Drive
CD-Mini-Project-Report - Goo
PRONGS-CHIRAG/Compiler-D
Server Not Found
Online JSON Viewer
jsonviewer.stack.hu
Viewer
Text
0 : "targets"
1 : "value"
targets
0
type : "Name"
loc
start
line : 15
column : 2
end
line : 15
column : 3
_fields
0 : "id"
1 : "ctx"
id : "j"
ctx : null
value
type : "Num"
loc
start
line : 15
column : 4
end
line : 15
column : 6
_fields
0 : "n"
n : 10
decorator_list
3
type : "Expr"
loc
Name
Value
body
...
loc
...
type
"Module"
_fields
...
Search:
GO!
Next
Previous
Type here to search
ENG
11:16
26-04-2019

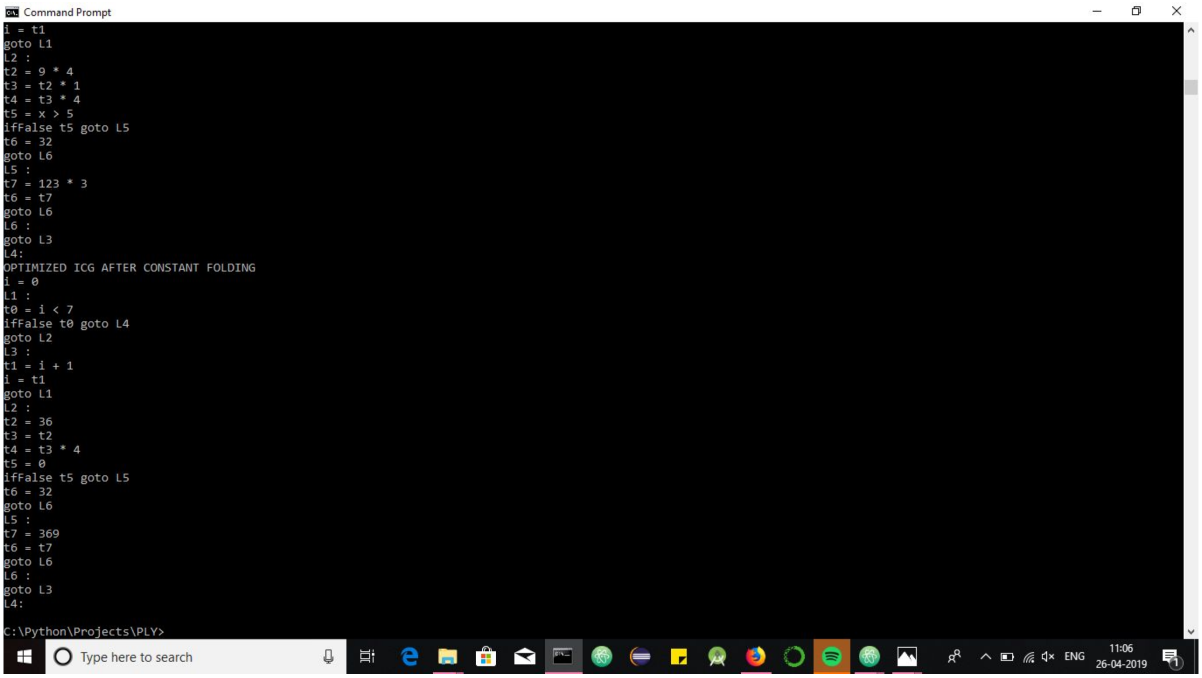




CODE OPTIMIZATIONS


Command Prompt
ICG
i = 0
L1 :
t0 = i < 7
ifFalse t0 goto L4
goto L2
L3 :
t1 = i + 1
i = t1
goto L1
L2 :
t2 = 9 * 4
t3 = t2 * 1
t4 = t3 * 4
t5 = x > 5
ifFalse t5 goto L5
t6 = 32
goto L6
L5 :
t7 = 123 * 3
t6 = t7
goto L6
L6 :
goto L3
L4:
OPTIMIZED ICG AFTER REMOVING DEAD CODE
i = 0
L1 :
t0 = i < 7
ifFalse t0 goto L4
goto L2
L3 :
t1 = i + 1
i = t1
goto L1
L2 :
t5 = x > 5
ifFalse t5 goto L5
goto L6
L5 :
goto L6
L6 :
goto L3
L4:
Type here to search
ENG
11:06
26-04-2019


Command Prompt
goto L6
L5 :
goto L6
L6 :
goto L3
L4:
Eliminated 6 lines of code
ICG
i = 0
L1 :
t0 = i < 7
ifFalse t0 goto L4
goto L2
L3 :
t1 = i + 1
i = t1
goto L1
L2 :
t2 = 9 * 4
t3 = t2 * 1
t4 = t3 * 4
t5 = x > 5
ifFalse t5 goto L5
t6 = 32
goto L6
L5 :
t7 = 123 * 3
t6 = t7
goto L6
L6 :
goto L3
L4:
OPTIMIZED ICG AFTER CONSTANT FOLDING
i = 0
L1 :
t0 = i < 7
ifFalse t0 goto L4
goto L2
L3 :
t1 = i + 1
i = t1
goto L1
L2 :
t2 = 36
Type here to search
ENG
11:06
26-04-2019



CONCLUSIONS

A mini-compiler for python is successfully built. It is built via 4 phases :- Lexical Analysis,Parsing,Intermediate Code Generation,Syntax Analysis and Code Optimisation.
Symbol Table, 3 address code and Abstract Syntax Tree has been generated.
Our compiler is able to identify and handle
ValueError,Keyerror,Indentation Error,SyntaxError

FURTHER ENHANCEMENTS

We can enhance it further by handling list comprehensions and classes. And we could include more optimization techniques to make it more accurate.