Report on

Python mini-compiler for 'for' and 'while' constructs

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

Compiler Design Laboratory

Bachelor of Technology
in
Computer Science & Engineering

Submitted by:

Sree Pranavi G	**PES1201800368**
Nikita Joanne Raj	**PES1201800808**
Sanriya S	**PES1201801233**

Under the guidance of

Madhura V
Asst. Professor
Dept. of CSE
PES University, Bengaluru

January – May 2021

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

1 : INTRODUCTION

This report discusses a mini compiler constructed for python language's for and while constructs using lex and yacc tools. The compiler is designed to begin with token generation followed by parsing based on the Context Free Grammar rules in section 4. The parser outputs the symbol tree as well which is then used for Intermediate Code Generation in the form of Three Address Code and represented in Quadruples. The Three Address Code is further optimized starting with strength reduction, then common subexpression elimination, followed by constant propagation and constant folding finally with dead code elimination. The optimized code is also represented in Quadruple format.

SAMPLE INPUT (detailed input snapshots attached in section 8)

```
1 x=2
2 y=3
3 c=x*y
4 while(x==True):                                    #p==True
5                 c=0
6                 for i in range(x,y):
7                         hello="world"
8                 z=10
9         #sdbfkdsjbgdjhbgdkf
```

SAMPLE OUTPUT (detailed output snapshots attached in section 8)

Terminal

```
pranavi@pranavi-VirtualBox:~/CD/Updated$ ./Test.out<ip1.py
-----------------------------------------------------------------------------------------------------------------------
Valid Python Syntax!
-----------------------------------------------------------------------------------------------------------------------
pranavi@pranavi-VirtualBox:~/CD/Updated$
```

Tokens.txt

```
 1
 2 ------------------------------------------------------------------Token Sequence----------------------------
 3 1 T_x T_EQL T_2 T_NL
 4 2 T_y T_EQL T_3 T_NL
 5 3 T_c T_EQL T_x T_ML T_y T_NL
 6 4 T_While T_OP T_x T_EQ T_True T_CP T_Cln T_NL
 7 5 T_ID T_c T_EQL T_0 T_NL
 8 6 T_ND T_For T_i T_In T_Range T_OP T_x T_Comma T_y T_CP T_Cln T_NL
 9 7 T_ID T_hello T_EQL T_"world" T_NL
10 8 T_DD T_z T_EQL T_10 T_NL
11 9 T_NL
12 10 T_EOF
13
14
```

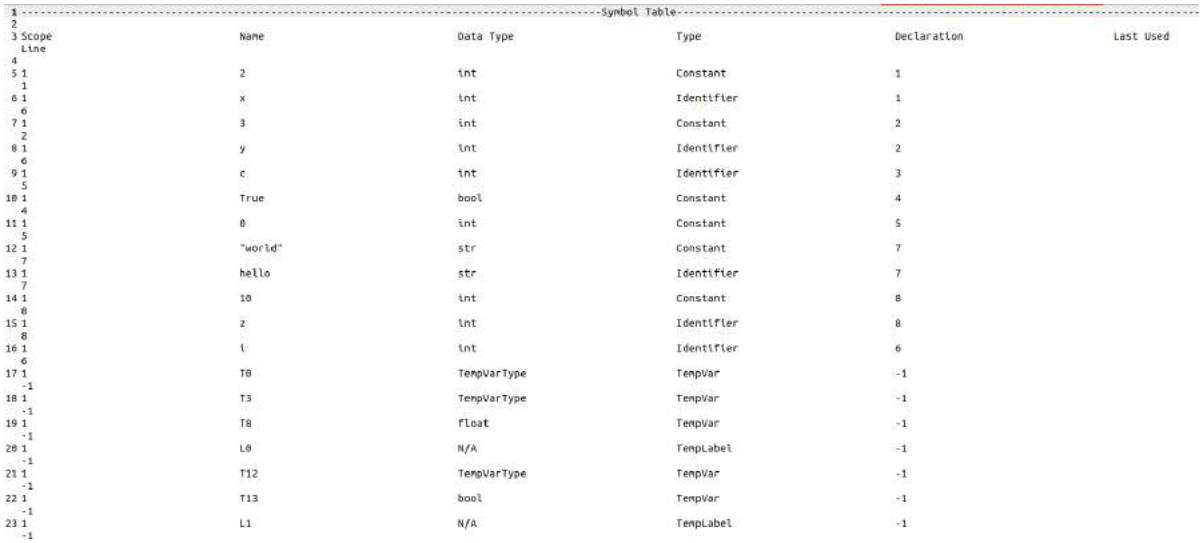
TAC.txt

```
1 ----------------------------------------------------------------Three Address Code---------------------
2 T0 = 2
3 x = T0
4 T3 = 3
5 y = T3
6 T0 = x
7 T3 = y
8 T8 = T0 * T3
9 c = T8
10
11 L0: T0 = x
12 T12 = True
13 T13 = T0 == T12
14 If False T13 goto L1
15 T14 = 0
16 c = T14
17 T0 = x
18 i = T17
19
20 L0: T17 = i
21 T0 = x
22 T29 = T17 >= T17
23 If False T29 goto L1
24 T17 = i
25 T3 = y
26 T30 = T17 < T17
27 If False T30 goto L1
28 T19 = "world"
29 hello = T19
30 T22 = 10
31 z = T22
32 goto L0
33 L1: goto L0
34 L1:
```

Quads.txt

```
1 ------------------------------------------------------------------------------------------------------------------------------------------------------
2
3 ------------------------------------------------------------------Quadruples ----------------------------------------------------------------------
4
5 Lno.                          Oper.                       Arg1                        Arg2                        Res
6
7 0                            =                           2                           -                           T0
8 1                            =                           T0                          -                           x
9 2                            =                           3                           -                           T3
10 3                           =                           T3                          -                           y
11 4                           =                           x                           -                           T0
12 5                           =                           y                           -                           T3
13 6                           *                           T0                          T3                          T8
14 7                           =                           T8                          -                           c
15 8                           Label                       -                           -                           L0
16 9                           =                           x                           -                           T0
17 10                          =                           True                        -                           T12
18 11                          ==                          T0                          T12                         T13
19 12                          If False                    T13                         -                           L1
20 13                          =                           0                           -                           T14
21 14                          =                           T14                         -                           c
22 15                          =                           x                           -                           T0
23 16                          =                           T17                         -                           i
24 17                          Label                       -                           -                           L0
25 18                          =                           i                           -                           T17
26 19                          =                           x                           -                           T0
27 20                          >=                          T17                         T17                         T29
28 21                          If False                    T29                         -                           L1
29 22                          =                           i                           -                           T17
30 23                          =                           y                           -                           T3
31 24                          <                           T17                         T17                         T30
32 25                          if false                    T30                         -                           L1
33 26                          =                           "world"                     -                           T19
34 27                          =                           T19                         -                           hello
35 28                          =                           10                          -                           T22
36 29                          =                           T22                         -                           z
37 30                          goto                        -                           -                           L0
38 31                          Label                       -                           -                           L1
39 32                          goto                        -                           -                           L0
40 33                          Label                       -                           -                           L1
41 ------------------------------------------------------------------------------------------------------------------------------------------------------
42
```

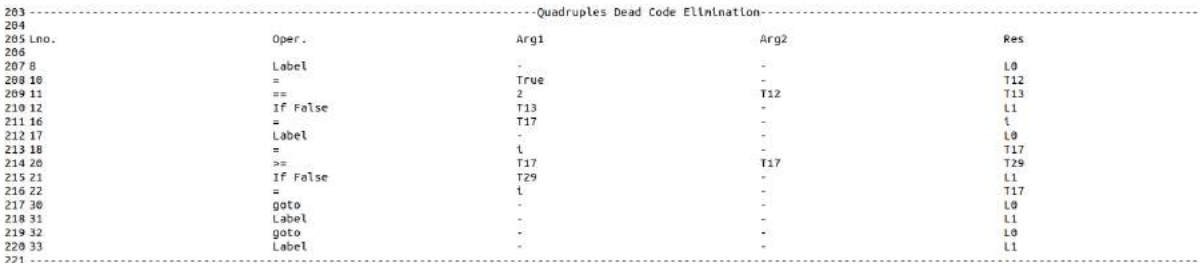
SymTab.txt (partial sample)



Symbol Table

Scope Line	Name	Data Type	Type	Declaration	Last Used
1 1	2	int	Constant	1	
1 1	x	int	Identifier	1	
1 1	3	int	Constant	2	
2 1	y	int	Identifier	2	
6 1	c	int	Identifier	3	
5 1	True	bool	Constant	4	
4 1	0	int	Constant	5	
5 1	"world"	str	Constant	7	
7 1	hello	str	Identifier	7	
7 1	10	int	Constant	8	
8 1	z	int	Identifier	8	
8 1	i	int	Identifier	6	
6 1	T0	TempVarType	TempVar	-1	
-1 1	T3	TempVarType	TempVar	-1	
-1 1	T8	float	TempVar	-1	
-1 1	L0	N/A	TempLabel	-1	
-1 1	T12	TempVarType	TempVar	-1	
-1 1	T13	bool	TempVar	-1	
-1 1	L1	N/A	TempLabel	-1	
-1					

Quads.txt (Final quadruples post full optimization)



Quadruples Dead Code Elimination

Lno.	Oper.	Arg1	Arg2	Res
8	Label	-	-	L0
10	=	True	-	T12
11	==	2	T12	T13
12	If False	T13	-	L1
16	=	T17	-	i
17	Label	-	-	L0
18	=	i	-	T17
20	>=	T17	T17	T29
21	If False	T29	-	L1
22	=	i	-	T17
30	goto	-	-	L0
31	Label	-	-	L1
32	goto	-	-	L0
33	Label	-	-	L1

2 : ARCHITECTURE OF LANGUAGE

SYNTAX

With the fluidity of python syntax, we have taken a subset of the language, as per the constructs given to us (for and while), to implement a mini compiler for python, with the following features:

- 'for' and 'while' constructs (including nested)
- Functions (including nested)
- Indentation
- Single line comments
- Multi line comments
- Import, break, void return statements
- Boolean and Arithmetic expressions
- Strings

SEMANTICS

- Identified semantic flaws and point out incorrect tokens
- Identify whether or not a variable used in an evaluation has been defined in the existing scope or not

3 : LITERATURE SURVEY

1. PLY : https://www.dabeaz.com/ply/ply.html#ply_nn24
2. PLY : https://ply.readthedocs.io/en/latest/ply.html#
3. Lex and Yacc: A Brisk Tutorial, Saumya K. Debray, Department of CSE, The University of Arizona,Tucson, AZ 85721.
4. LEX & YACC TUTORIAL by Tom Niemann.
5. https://docs.python.org/3/reference/grammar.html
6. https://docs.python.org/3/reference/lexical_analysis.html#indentation
7. https://silcnitc.github.io/yacc.html#yylex
8. https://docs.python.org/3/reference/expressions.html
9. https://github.com/jengelsma/lex-tutorial/blob/master/myscanner.c
10. https://www.cs.ccu.edu.tw/~naiwei/cs5605/YaccBison.html
11. https://codedost.com/flex/flex-programs/flex-program-check-use-yyless-function/

4 : CONTEXT FREE GRAMMAR

$accept: RunCompiler $end ; **$@1:** %empty

RunCompiler: $@1 BeginParse T_EndOfFile

constant: T_Number | T_String | T_True | T_False

term: T_ID | constant | list_index

list_index: T_ID T_OB constant T_CB

BeginParse: T_NL BeginParse ; **$@2:** %empty

BeginParse: finalStatements T_NL $@2 BeginParse | finalStatements T_NL

simple_stmt: pass_stmt | break_stmt | import_stmt | assign_stmt | arith_exp | bool_exp | return_stmt

arith_exp: term | arith_exp T_PL arith_exp | arith_exp T_MN arith_exp | arith_exp T_ML arith_exp
| arith_exp T_DV arith_exp | T_MN arith_exp | T_OP arith_exp T_CP

bool_exp: bool_term T_Or bool_term | arith_exp T_LT arith_exp | bool_term T_And bool_term
| arith_exp T_GT arith_exp | arith_exp T_ELT arith_exp | arith_exp T_EGT arith_exp
| arith_exp T_In T_ID | bool_term

bool_term: bool_factor | arith_exp T_EQ arith_exp | T_True | T_False

bool_factor: T_Not bool_factor | T_OP bool_exp T_CP

import_stmt: T_Import T_ID

pass_stmt: T_Pass

break_stmt: T_Break

return_stmt: T_Return

assign_stmt: T_ID T_EQL arith_exp | T_ID T_EQL bool_exp | T_ID T_EQL func_call
| T_ID T_EQL T_OB T_CB

finalStatements: simple_stmt | cmpd_stmt | func_def | func_call | error T_NL

cmpd_stmt: while_stmt | for_stmt

while_stmt: T_While bool_exp T_Cln begin_block

for_stmt: T_For T_ID T_In T_Range T_OP term T_Comma term T_CP T_Cln begin_block

begin_block: simple_stmt ; **$@3:** %empty

begin_block: T_NL T_Indent $@3 finalStatements block

block: T_NL T_Nodent finalStatements block | T_NL end_block ; **$@4:** %empty

end_block: T_Dedent $@4 finalStatements

$@5: %empty

end_block: T_Dedent $@5 | %empty

$@6: %empty

args: T_ID $@6 args_list | %empty

$@7: %empty

args_list: T_Comma T_ID $@7 args_list | %empty

$@8: %empty

call_list: T_Comma term $@8 call_list | %empty

$@9: %empty

call_args: T_ID $@9 call_list

$@10: %empty

call_args: T_Number $@10 call_list

$@11: %empty

call_args: T_String $@11 call_list | %empty

$@12: %empty

func_def: T_Def T_ID $@12 T_OP args T_CP T_Cln begin_block

func_call: T_ID T_OP call_args T_CP

5 : DESIGN STRATEGY

SYMBOL TABLE CREATION

The symbol table is stored using two structures: struct SymTable and struct Record.

The Symbol Table maintains Records(struct), the number of elements in the current symbol table, current scope, and the details of the elements stored.

The Records structure stores the name of each variable as well as the type. It also maintains the line number that the variable was declared on and the last line it was used.

Scope has been recorded for functions, per python scope rules.

For scope handling, in cases where code is at the same indentation level, but different scope, we have used a hashing technique to ensure that the two blocks of code have different scopes.

The information stored in the symbol table is needed and accessed by the other functions that implement intermediate code generation etc.

INTERMEDIATE CODE GENERATION

The Abstract Syntax Tree is created and utilised for the generation of intermediate code. Once a grammar rule is recognised in the input, the corresponding elements of that rule are pushed into the AST, which is consequently read into three address code.

An index variable is maintained for tracking the temporary variables and labels used in the intermediate code.

The generated AST is read recursively, on a node-by-node basis, in order to generate intermediate code. The intermediate three address code is generated and stored in the form of quadruples, which are then optimized in the following phase.

CODE OPTIMIZATION

The optimization of generated TAC was performed using 5 techniques,

1. Strength Reduction - To replace costly operations like * & / with cheaper ones like shifts at bit level.

2. Common Subexpression Elimination - If E is previously computed and the values in E have not changed since the previous computation, the duplicate instances are removed.

3. Constant Propagation - Constant assigned to a variable is substituted when the variable is encountered during compile time.

4. Constant Folding - Recognizing and evaluating constant expressions at compile time rather than computing them at runtime.

5. Dead Code Elimination - To remove useless or unreachable code.

6 : IMPLEMENTATION DETAILS

SYMBOL TABLE CREATION

In order to implement the symbol table, we used two different structures:

```
typedef struct Record
{
    char *type;
    char *name;
    char *datatype;
    int decLine;
    int lastLine;
} Record;

typedef struct SymTable
{
    int ele_count;
    int symTableScope;
    Record *Elements;
} SymTable;
```

The Record structure represents each record in a symbol table that holds the name and type of the data as well as the line it was declared on and the last line it was used.
The SymTable structure keeps track of the number of elements in the current symbol table, its scope, and parent scope. It also contains a pointer to Elements of type Record that stores the details of the data. A new symbol table is created for each scope.

INTERMEDIATE CODE GENERATION

As elaborated in the above section, the Abstract Syntax Tree is utilised for the generation of intermediate code. The AST is maintained by a structure, ASTNode:

```
typedef struct ASTNode
{
   int nodeNo;
   //if the Node is an operator
   char *NType;
   int opCount;
   struct ASTNode** NextLevel;
   //if the Node is an identifier or a constant
   Record *id;
} Node;
```

The intermediate code, generated from the AST, is stored in the form of quadruples:

```
typedef struct Quad
{
   char *R;
   char *A1;
   char *A2;
   char *Op;
   //for optimisation
   int I;
} Quad;
```

CODE OPTIMIZATION

The function optimization() calls all the code optimization technique functions.

First strength reduction is performed to replace costlier expressions with cheaper ones. The function strengthRedn() iterates through the Quad table and finds records with non null arg2, non = operator. It finds the value stored in arg2 and performs left or right shift operations depending on * or / operator, iff the arg is a power of 2 which is checked using pow2 function, else it remains the same. So, arg1 * arg2 becomes arg1 << arg2 and arg1 / arg2 becomes arg1 >> arg2.

The next type of optimization performed is common subexpression elimination using the commonSubexprElim() function, which searches the strength reduced quad table to find duplicate records with the same operator and args.

constantProp() is the next optimization technique which replaces a variable with it's constant value, if available in all the following instances. The subexpression eliminated quad table is iterated through and if any record with numeral arg 1 and empty arg 2 is found, all it's upcoming instances are replaced with its numeric value.

Following this is constantFolding() which computes any numerical expressions and stores the value in the result. While iterating through the constant propagated quad table, if any record with numeric arg1 and arg 2 are found, the expected computation based on the operator value is performed.

Finally the dead code elimination is performed to remove useless using deadCodeElimination() function which continuously iterates through the optimized quad table until no more dead code is found. A record is marked as dead code in case the result variable does not appear anywhere else beyond that point and eliminated by marking it's index as -1 generating the final optimized code.

ERROR HANDLING

We have identified semantic flaws and offered expected tokens whenever possible, in the places where an error has been detected.

Using the symbol table, we identify whether or not a variable used in an evaluation has been defined in the existing scope or not, and proceed accordingly.

INSTRUCTIONS TO BUILD AND RUN PROGRAM

1. The python_compiler.l file is the lex code and python_compiler.y file is the yacc code, ip.py is the python input file and the 'makefile' contains all the instructions to run the files.

2. Upon running 'make' command, the following files are generated : lex.yy.c y.tab.c y.tab.h Test.out y.output.

3. ‘Test.out’ is the file to be executed, the input to this file is redictered from ‘ip.py’ as the source code to the compiler. The command is as follows ‘ ./Test.out < ip.py ‘.

4. The output is redirected to separate text files. The terminal displays the syntax validity, the tokenization output can be found in ‘Tokens.txt’, symbol table in ‘SymTab.txt’, TAC in ‘TAC.txt’ and the quadruples before and after optimization in ‘Quads.txt’.

5. Finally, the commands to be executed are,
   $make
   $./Test.out < ip.py

7 : RESULTS AND POSSIBLE SHORTCOMINGS

RESULTS

A mini-compiler was built for python, implementing the specified constraints. In the process, we acquired a more in-depth understanding of the following points:

- Familiarity with the Lex Yacc tool
- Context-Free Grammar
- Applications and variations of the symbol table
- Reading and writing intermediate code
- Code optimisation techniques

SHORTCOMINGS

With the python syntax being as diverse as it is, we have not completed alternative syntax for certain constructs:

- for with step-counter in range
  for i in range (10,5,-1):
- return statement with a value

8 : SNAPSHOTS

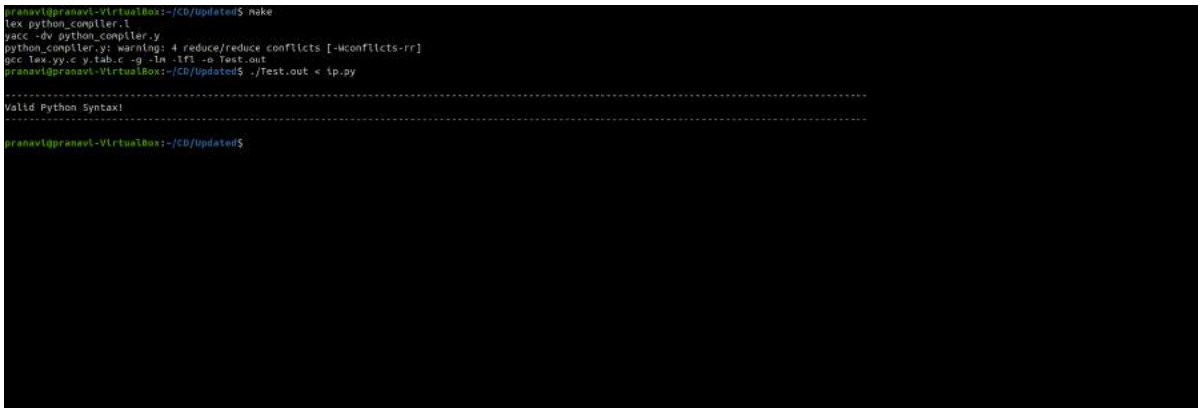
INPUT

Source code input
File name : ip.py

```
import hWorld
x=80
y=10

#Comment1
#x+y
#listX = []

c=x*4
m=x/8

n=x/10
'''
#multi line cmt
'''
l=m+n

k=m*n

h=m+n

def F1(A, B, C):
        while(x==True):                                 #p==True
                c=0
                for i in range(x,y):
                        hello="world"
                z=10
                b=z
                w=21
        return

                                        #sdbfkdsjbgdjhbgdkf

def F2():
        def F3():
                c=111
                def F4():
                        d=233


F2()
#Comment2
F1(10, 10, 10)
```

OUTPUT

Terminal output



Tokenization output

File name : Tokens.txt

```
1
2 ----------------------------------------------------------Token Sequence----------------------------------------------------------
3 1 T_IMPT T_hWorld T_NL
4 2 T_x T_EQL T_80 T_NL
5 3 T_y T_EQL T_10 T_NL
6 4 T_NL
7 5 T_NL
8 6 T_NL
9 7 T_NL
10 8 T_NL
11 9 T_c T_EQL T_x T_ML T_4 T_NL
12 10 T_m T_EQL T_x T_DV T_8 T_NL
13 11 T_NL
14 12 T_n T_EQL T_x T_DV T_10 T_NL
15 13 T_NL
16 16 T_l T_EQL T_m T_PL T_n T_NL
17 17 T_NL
18 18 T_k T_EQL T_m T_ML T_n T_NL
19 19 T_NL
20 20 T_h T_EQL T_m T_PL T_n T_NL
21 21 T_NL
22 22 T_Def T_F1 T_OP T_A T_Comma T_B T_Comma T_C T_CP T_Cln T_NL
23 23 T_ID T_While T_OP T_x T_EQ T_True T_CP T_Cln T_NL
24 24 T_ID T_c T_EQL T_0 T_NL
25 25 T_ND T_For T_i T_In T_Range T_OP T_x T_Comma T_y T_CP T_Cln T_NL
26 26 T_ID T_hello T_EQL T_"world" T_NL
27 27 T_DD T_z T_EQL T_10 T_NL
28 28 T_ND T_b T_EQL T_z T_NL
29 29 T_ND T_w T_EQL T_21 T_NL
30 30 T_DD T_Return T_NL
31 31 T_NL
32 32 T_NL
33 33 T_NL
34 34 T_Def T_F2 T_OP T_CP T_Cln T_NL
35 35 T_ID T_Def T_F3 T_OP T_CP T_Cln T_NL
36 36 T_ID T_c T_EQL T_111 T_NL
37 37 T_ND T_Def T_F4 T_OP T_CP T_Cln T_NL
38 38 T_ID T_d T_EQL T_233 T_NL
39 39 T_NL
40 40 T_NL
41 41 T_NL
42 42 T_NL
43 43 T_F2 T_OP T_CP T_NL
44 44 T_NL
45 45 T_F1 T_OP T_10 T_Comma T_10 T_Comma T_10 T_CP T_NL
```

ICG - Three Address Code output

File name : TAC.txt

```
1 ------------------------------------------------------------Three Address Code------------------------------------------------------------
 2 import hWorld
 3 T2 = 80
 4 x = T2
 5 T5 = 10
 6 y = T5
 7 T2 = x
 8 T9 = 4
 9 T10 = T2 * T9
10 c = T10
11 T2 = x
12 T14 = 8
13 T15 = T2 / T14
14 m = T15
15 T2 = x
16 T19 = 10
17 T20 = T2 / T19
18 n = T20
19 T15 = m
20 T20 = n
21 T25 = T15 + T20
22 l = T25
23 T15 = m
24 T20 = n
25 T30 = T15 * T20
26 k = T30
27 T15 = m
28 T20 = n
29 T35 = T15 + T20
30 h = T35
31 Begin Function F1
32
33 L0: T2 = x
34 T40 = True
35 T41 = T2 == T40
36 If False T41 goto L1
37 T42 = 0
38 c = T42
39 T2 = x
40 i = T45

41
42 L0: T45 = i
43 T2 = x
44 T57 = T45 >= T45
45 If False T57 goto L1
46 T45 = i
47 T5 = y
48 T58 = T45 < T45
49 If False T58 goto L1
50 T47 = "world"
51 hello = T47
52 T50 = 10
53 z = T50
54 goto L0
55 L1: T50 = z
56 b = T60
57 T63 = 21
58 w = T63
59 goto L0
60 L1: End Function F1
61 Begin Function F2
62 Begin Function F3
63 T79 = 111
64 c = T79
65 Begin Function F4
66 T83 = 233
67 d = T83
68 End Function F4
69 End Function F3
70 End Function F2
71 (T101)Call Function F2
72 Push Param 10
73 Push Param  10
74 Push Param  10
75 (T106)Call Function F1, 3
76 Pop Params for Function F1, 3
```

ICG - Quadruple output

File name : Quads.txt

```
1  ------------------------------------------------------------------------------------------------------------------------------------------------------
2
3  ------------------------------------------------------------------Quadruples ------------------------------------------------------------------------
4
5  Lno.                    Oper.                   Arg1                    Arg2                    Res
6
7  0                       import                  hWorld                  -                       -
8  1                       =                       80                      -                       T2
9  2                       =                       T2                      -                       x
10 3                       =                       10                      -                       T5
11 4                       =                       T5                      -                       y
12 5                       =                       x                       -                       T2
13 6                       =                       4                       -                       T9
14 7                       *                       T2                      T9                      T10
15 8                       =                       T10                     -                       c
16 9                       =                       x                       -                       T2
17 10                      =                       8                       -                       T14
18 11                      /                       T2                      T14                     T15
19 12                      =                       T15                     -                       n
20 13                      =                       x                       -                       T2
21 14                      =                       10                      -                       T19
22 15                      /                       T2                      T19                     T20
23 16                      =                       T20                     -                       n
24 17                      =                       n                       -                       T15
25 18                      =                       n                       -                       T20
26 19                      +                       T15                     T20                     T25
27 20                      =                       T25                     -                       l
28 21                      =                       n                       -                       T15
29 22                      =                       n                       -                       T20
30 23                      *                       T15                     T20                     T30
31 24                      =                       T30                     -                       k
32 25                      =                       n                       -                       T15
33 26                      =                       n                       -                       T20
34 27                      +                       T15                     T20                     T35
35 28                      =                       T35                     -                       h
36 29                      BeginF                  F1                      -                       -
37 30                      Label                   -                       -                       L0
38 31                      =                       x                       -                       T2
39 32                      =                       True                    -                       T40
40 33                      ==                      T2                      T40                     T41
41 34                      If False                T41                     -                       L1
42 35                      =                       0                       -                       T42
43 36                      =                       T42                     -                       c
44 37                      =                       x                       -                       T2

45 38                      =                       T45                     -                       i
46 39                      Label                   -                       -                       L0
47 40                      =                       i                       -                       T45
48 41                      =                       x                       -                       T2
49 42                      >=                      T45                     T45                     T57
50 43                      If False                T57                     -                       L1
51 44                      =                       i                       -                       T45
52 45                      =                       y                       -                       T5
53 46                      <                       T45                     T45                     T58
54 47                      if false                T58                     -                       L1
55 48                      =                       "world"                 -                       T47
56 49                      =                       T47                     -                       hello
57 50                      =                       10                      -                       T50
58 51                      =                       T50                     -                       z
59 52                      goto                    -                       -                       L0
60 53                      Label                   -                       -                       L1
61 54                      =                       z                       -                       T50
62 55                      =                       T60                     -                       b
63 56                      =                       21                      -                       T63
64 57                      =                       T63                     -                       w
65 58                      goto                    -                       -                       L0
66 59                      Label                   -                       -                       L1
67 60                      EndF                    F1                      -                       -
68 61                      BeginF                  F2                      -                       -
69 62                      BeginF                  F3                      -                       -
70 63                      =                       111                     -                       T79
71 64                      =                       T79                     -                       c
72 65                      BeginF                  F4                      -                       -
73 66                      =                       233                     -                       T83
74 67                      =                       T83                     -                       d
75 68                      EndF                    F4                      -                       -
76 69                      EndF                    F3                      -                       -
77 70                      EndF                    F2                      -                       -
78 71                      Call                    F2                      -                       T101
79 72                      Param                   10                      -                       -
80 73                      Param                    10                     -                       -
81 74                      Param                    10                     -                       -
82 75                      Call                    F1                      3                       T106
83 ------------------------------------------------------------------------------------------------------------------------------------------------------
```

Symbol Table output

File name : SymTab.txt

Symbol Table

Scope	Name	Data Type	Type	Declaration	Last Used Line
1	hWorld	N/A	PackageName	1	1
1	80	int	Constant	2	2
1	x	int	Identifier	2	25
1	10	int	Constant	3	12
1	y	int	Identifier	3	25
1	4	int	Constant	9	9
1	c	int	Identifier	9	9
1	8	int	Constant	10	10
1	m	int	Identifier	10	20
1	n	int	Identifier	12	20
1	l	int	Identifier	16	16
1	k	int	Identifier	18	18
1	h	int	Identifier	20	20
1	F1	fun	Func_Name	22	22
1	T2	TempVarType	TempVar	-1	-1
1	T5	TempVarType	TempVar	-1	-1
1	T9	TempVarType	TempVar	-1	-1
1	T10	float	TempVar	-1	-1
1	T14	TempVarType	TempVar	-1	-1
1	T19	TempVarType	TempVar	-1	-1
1	T20	float	TempVar	-1	-1
1	T25	float	TempVar	-1	-1
1	T30	float	TempVar	-1	-1
1	T35	float	TempVar	-1	-1
1	L0	N/A	TempLabel	-1	-1
1	T40	TempVarType	TempVar	-1	-1
1	T41	bool	TempVar	-1	-1
1	L1	N/A	TempLabel	-1	-1
1	T42	TempVarType	TempVar	-1	-1
1	T45	TempVarType	TempVar	-1	-1
1	T57	bool	TempVar	-1	-1
1	T58	bool	TempVar	-1	-1
1	T47	TempVarType	TempVar	-1	-1
1	T50	TempVarType	TempVar	-1	-1
1	T60	TempVarType	TempVar	-1	-1
1	T63	TempVarType	TempVar	-1	-1
1	T79	TempVarType	TempVar	-1	-1
1	T83	TempVarType	TempVar	-1	-1
1	T101	fun	TempVar	-1	-1
1	T106	fun	TempVar	-1	-1
4	True	bool	Constant	23	23
9	0	int	Constant	24	24
9	c	int	Identifier	24	24
9	10	int	Constant	27	45
9	z	int	Identifier	27	28
9	i	int	Identifier	25	25
9	b	int	Identifier	28	28
9	21	int	Constant	29	29
9	w	int	Identifier	29	29
9	F2	fun	Func_Name	34	34
16	"world"	str	Constant	26	26
16	hello	str	Identifier	26	26
64	F3	fun	Func_Name	35	35
25	111	int	Constant	36	36
25	c	int	Identifier	36	36
25	F4	fun	Func_Name	37	37
36	233	int	Constant	38	38
36	d	int	Identifier	38	38

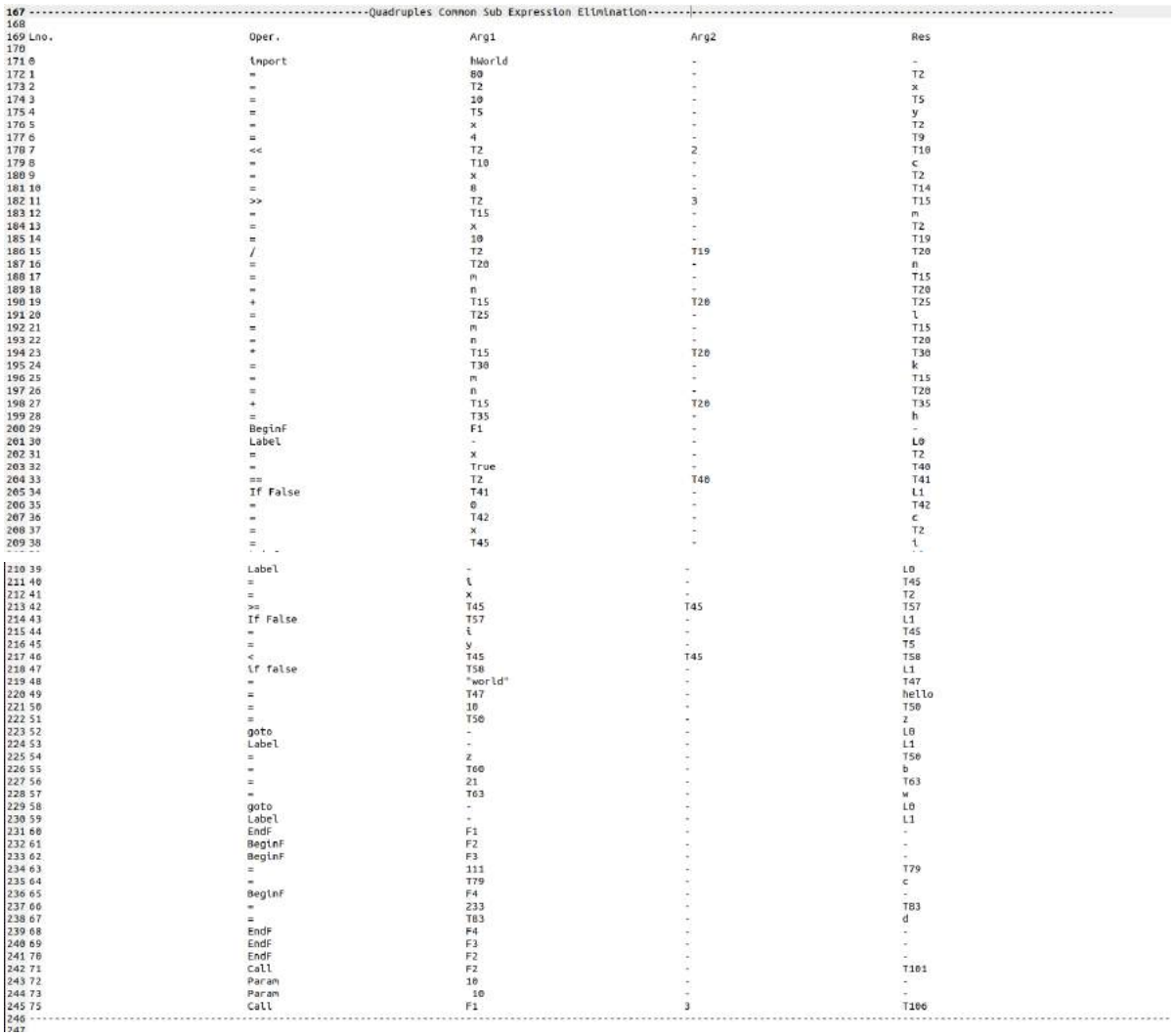
Optimized code output - After Strength Reduction

File name : Quads.txt

```
                                                    Quadruples Strength Reduction

Lno.    Oper.       Arg1        Arg2        Res

0       import      hWorld      -           -
1       =           80          -           T2
2       =           T2          -           x
3       =           10          -           T5
4       =           T5          -           y
5       =           x           -           T2
6       =           4           -           T9
7       <<          T2          2           T10
8       =           T10         -           c
9       =           x           -           T2
10      =           8           -           T14
11      >>          T2          3           T15
12      =           T15         -           m
13      =           x           -           T2
14      =           10          -           T19
15      /           T2          T19         T20
16      =           T20         -           n
17      =           m           -           T15
18      =           n           -           T20
19      +           T15         T20         T25
20      =           T25         -           l
21      =           m           -           T15
22      =           n           -           T20
23      *           T15         T20         T30
24      =           T30         -           k
25      =           m           -           T15
26      =           n           -           T20
27      +           T15         T20         T35
28      =           T35         -           h
29      BeginF      F1          -           -
30      Label       -           -           L0
31      =           x           -           T2
32      =           True        -           T40
33      ==          T2          T40         T41
34      If False    T41         -           L1
35      =           0           -           T42
36      =           T42         -           c
37      =           x           -           T2
38      =           T45         -           i

39      Label       -           -           L0
40      =           i           -           T45
41      =           x           -           T2
42      >=          T45         T45         T57
43      If False    T57         -           L1
44      =           l           -           T45
45      =           y           -           T5
46      <           T45         T45         T58
47      If false    T58         -           L1
48      =           "world"     -           T47
49      =           T47         -           hello
50      =           10          -           T50
51      =           T50         -           z
52      goto        -           -           L0
53      Label       -           -           L1
54      =           z           -           T50
55      =           T60         -           b
56      =           21          -           T63
57      =           T63         -           w
58      goto        -           -           L0
59      Label       -           -           L1
60      EndF        F1          -           -
61      BeginF      F2          -           -
62      BeginF      F3          -           -
63      =           111         -           T79
64      =           T79         -           c
65      BeginF      F4          -           -
66      =           233         -           T83
67      =           T83         -           d
68      EndF        F4          -           -
69      EndF        F3          -           -
70      EndF        F2          -           -
71      Call        F2          -           T101
72      Param       10          -           -
73      Param        10         -           -
74      Param        10         -           -
75      Call        F1          3           T106
```

Optimized code output - After Common Subexpression Elimination

File name : Quads.txt



Lno.	Oper.	Arg1	Arg2	Res
0	Import	hWorld	-	-
1	=	80	-	T2
2	=	T2	-	x
3	=	10	-	T5
4	=	T5	-	y
5	=	x	-	T2
6	=	4	-	T9
7	<<	T2	2	T10
8	=	T10	-	c
9	=	x	-	T2
10	=	8	-	T14
11	>>	T2	3	T15
12	=	T15	-	m
13	=	x	-	T2
14	=	10	-	T19
15	/	T2	T19	T20
16	=	T20	-	n
17	=	m	-	T15
18	=	n	-	T20
19	+	T15	T20	T25
20	=	T25	-	l
21	=	m	-	T15
22	=	n	-	T20
23	*	T15	T20	T30
24	=	T30	-	k
25	=	m	-	T15
26	=	n	-	T20
27	+	T15	T20	T35
28	=	T35	-	h
29	BeginF	F1	-	-
30	Label	-	-	L0
31	=	x	-	T2
32	=	True	-	T40
33	==	T2	T40	T41
34	If False	T41	-	L1
35	=	0	-	T42
36	=	T42	-	c
37	=	x	-	T2
38	=	T45	-	i
39	Label	-	-	L0
40	=	i	-	T45
41	=	x	-	T2
42	>=	T45	T45	T57
43	If False	T57	-	L1
44	=	i	-	T45
45	=	y	-	T5
46	<	T45	T45	T58
47	If false	T58	-	L1
48	=	"world"	-	T47
49	=	T47	-	hello
50	=	10	-	T50
51	=	T50	-	z
52	goto	-	-	L0
53	Label	-	-	L1
54	=	z	-	T50
55	=	T60	-	b
56	=	21	-	T63
57	=	T63	-	w
58	goto	-	-	L0
59	Label	-	-	L1
60	EndF	F1	-	-
61	BeginF	F2	-	-
62	BeginF	F3	-	-
63	=	111	-	T79
64	=	T79	-	c
65	BeginF	F4	-	-
66	=	233	-	T83
67	=	T83	-	d
68	EndF	F4	-	-
69	EndF	F3	-	-
70	EndF	F2	-	-
71	Call	F2	-	T101
72	Param	10	-	-
73	Param	10	-	-
75	Call	F1	3	T106

Optimized code output - After Constant Propagation

File name : Quads.txt

```
----------------------------------------------------------------------Quadruples Constant Propagation----------------------------------------------------------------------

Lno.    Oper.       Arg1        Arg2        Res
0       import      hWorld      -           -
1       =           80          -           T2
2       =           80          -           x
3       =           10          -           T5
4       =           10          -           y
5       =           80          -           TZ
6       =           4           -           T9
7       <<          80          2           T10
8       =           T10         -           c
9       =           80          -           T2
10      =           8           -           T14
11      >>          80          3           T15
12      =           T15         -           m
13      =           80          -           T2
14      =           10          -           T19
15      /           80          10          T20
16      =           T20         -           n
17      =           m           -           T15
18      =           n           -           T20
19      +           T15         T20         T25
20      =           T25         -           l
21      =           m           -           T15
22      =           n           -           T20
23      *           T15         T20         T30
24      =           T30         -           k
25      =           m           -           T15
26      =           n           -           T20
27      +           T15         T20         T35
28      =           T35         -           h
29      BeginF      F1          -           -
30      Label       -           -           L0
31      =           80          -           T2
32      =           True        -           T40
33      ==          80          T40         T41
34      If False    T41         -           L1
35      =           0           -           T42
36      =           T42         -           c
37      =           80          -           T2

38      =           T45         -           i
39      Label       -           -           L0
40      =           i           -           T45
41      =           80          -           T2
42      >=          T45         T45         T57
43      If False    T57         -           L1
44      =           i           -           T45
45      =           10          -           T5
46      <           T45         T45         T58
47      if false    T58         -           L1
48      =           "world"     -           T47
49      =           T47         -           hello
50      =           10          -           T50
51      =           10          -           z
52      goto        -           -           L0
53      Label       -           -           L1
54      =           10          -           T50
55      =           T60         -           b
56      =           21          -           T63
57      =           21          -           w
58      goto        -           -           L0
59      Label       -           -           L1
60      EndF        F1          -           -
61      BeginF      F2          -           -
62      BeginF      F3          -           -
63      =           111         -           T79
64      =           111         -           c
65      BeginF      F4          -           -
66      =           233         -           T83
67      =           233         -           d
68      EndF        F4          -           -
69      EndF        F3          -           -
70      EndF        F2          -           -
71      Call        F2          -           T101
72      Param       10          -           -
73      Param       10          10          -
75      Call        F1          3           T106
-----------------------------------------------------------------------------------------------------------------------------------------------------------------------
```

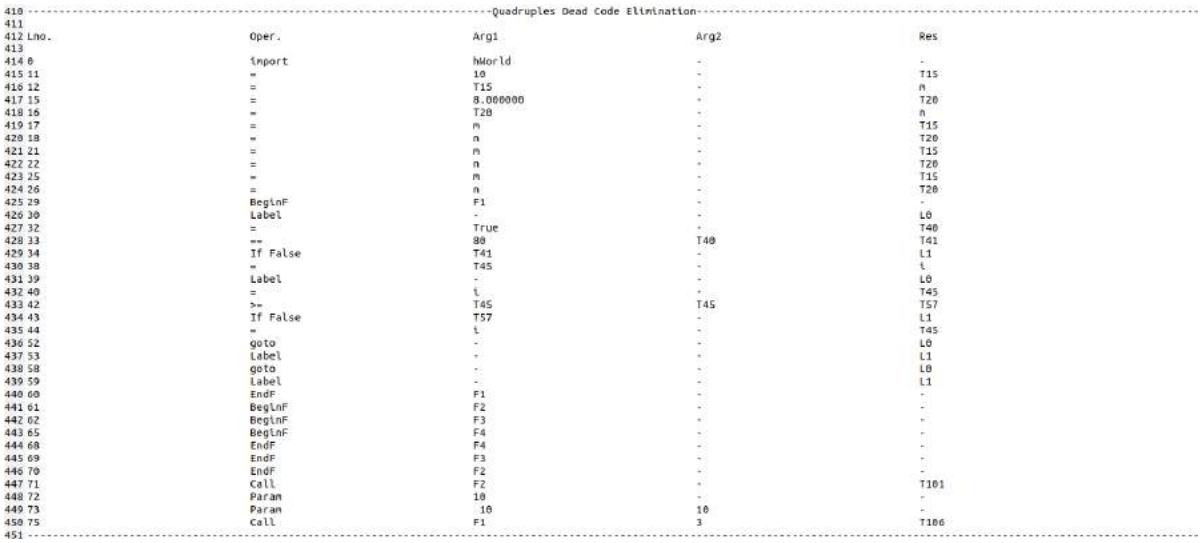
Optimized code output - After Constant Folding

File name : Quads.txt

```
329 ----------------------------------------------------------------------Quadruples Constant Folding----------------------------------------------------------------------
330
331 Lno.                        Oper.                       Arg1                        Arg2                        Res
332
333 0                           import                      hWorld                      -                           -
334 1                           =                           80                          -                           T2
335 2                           =                           80                          -                           x
336 3                           =                           10                          -                           T5
337 4                           =                           10                          -                           y
338 5                           =                           80                          -                           T2
339 6                           =                           4                           -                           T9
340 7                           =                           320                         -                           T10
341 8                           =                           T10                         -                           c
342 9                           =                           80                          -                           T2
343 10                          =                           8                           -                           T14
344 11                          =                           10                          -                           T15
345 12                          =                           T15                         -                           m
346 13                          =                           80                          -                           T2
347 14                          =                           10                          -                           T19
348 15                          =                           8.000000                    -                           T20
349 16                          =                           T20                         -                           n
350 17                          =                           m                           -                           T15
351 18                          =                           n                           -                           T20
352 19                          +                           T15                         T20                         T25
353 20                          =                           T25                         -                           l
354 21                          =                           m                           -                           T15
355 22                          =                           n                           -                           T20
356 23                          *                           T15                         T20                         T30
357 24                          =                           T30                         -                           k
358 25                          =                           m                           -                           T15
359 26                          =                           n                           -                           T20
360 27                          +                           T15                         T20                         T35
361 28                          =                           T35                         -                           h
362 29                          BeginF                      F1                          -                           -
363 30                          Label                       -                           -                           L0
364 31                          =                           80                          -                           T2
365 32                          =                           True                        -                           T40
366 33                          ==                          80                          T40                         T41
367 34                          If False                    T41                         -                           L1
368 35                          =                           0                           -                           T42
369 36                          =                           T42                         -                           c
370 37                          =                           80                          -                           T2
371 38                          =                           T45                         -                           i
372 39                          Label                       -                           -                           L0

373 40                          =                           i                           -                           T45
374 41                          =                           80                          -                           T2
375 42                          >=                          T45                         T45                         T57
376 43                          If False                    T57                         -                           L1
377 44                          =                           i                           -                           T45
378 45                          =                           10                          -                           T5
379 46                          <                           T45                         T45                         T58
380 47                          If false                    T58                         -                           L1
381 48                          =                           "world"                     -                           T47
382 49                          =                           T47                         -                           hello
383 50                          =                           10                          -                           T50
384 51                          =                           10                          -                           z
385 52                          goto                        -                           -                           L0
386 53                          Label                       -                           -                           L1
387 54                          =                           10                          -                           T50
388 55                          =                           T60                         -                           b
389 56                          =                           21                          -                           T63
390 57                          =                           21                          -                           w
391 58                          goto                        -                           -                           L0
392 59                          Label                       -                           -                           L1
393 60                          EndF                        F1                          -                           -
394 61                          BeginF                      F2                          -                           -
395 62                          BeginF                      F3                          -                           -
396 63                          =                           111                         -                           T79
397 64                          =                           111                         -                           c
398 65                          BeginF                      F4                          -                           -
399 66                          =                           233                         -                           T83
400 67                          =                           233                         -                           d
401 68                          EndF                        F4                          -                           -
402 69                          EndF                        F3                          -                           -
403 70                          EndF                        F2                          -                           -
404 71                          Call                        F2                          -                           T101
405 72                          Param                       10                          -                           -
406 73                          Param                       10                          10                          -
407 75                          Call                        F1                          3                           T106
408 ----------------------------------------------------------------------------------------------------------------------------------------------------------------
409
```

Optimized code output - After Dead Code Elimination

File name : Quads.txt



```
----------------------------------------Quadruples Dead Code Elimination----------------------------------------

Lno.    Oper.       Arg1        Arg2        Res
0       %mport      hWorld      -           -
11      =           10          -           T15
12      =           T15         -           n
15      =           8.000000    -           T20
16      =           T20         -           n
17      =           n           -           T15
18      =           n           -           T20
21      =           n           -           T15
22      =           n           -           T20
25      =           n           -           T15
26      =           n           -           T20
29      BeginF      F1          -           -
30      Label       -           -           L0
32      =           True        -           T40
33      ==          80          T40         T41
34      If False    T41         -           L1
38      =           T45         -           t
39      Label       -           -           L0
40      =           t           -           T45
42      >=          T45         T45         T57
43      If False    T57         -           L1
44      =           t           -           T45
52      goto        -           -           L0
53      Label       -           -           L1
58      goto        -           -           L0
59      Label       -           -           L1
60      EndF        F1          -           -
61      BeginF      F2          -           -
62      BeginF      F3          -           -
65      BeginF      F4          -           -
68      EndF        F4          -           -
69      EndF        F3          -           -
70      EndF        F2          -           -
71      Call        F2          -           T101
72      Param       10          -           -
73      Param       10          10          -
75      Call        F1          3           T106

----------------------------------------------------------------------------------------------------
```

9 : CONCLUSION

In Conclusion, a mini compiler for Python is implemented for ‘for’ and ‘while’ constructs. In addition to the constructs specified, the basic python constructs are implemented and function definitions and calls are supported.

The compiler also reports the basic errors and gives the line number and column number. The Intermediate code is represented by quads which is later optimized to remove dead code.

10 : FURTHER ENHANCEMENT

The compiler could be further enhanced by adding/improving

- Lists
- if-else constructs
- return statements with value
- Functions with a non-void return type
- More optimization techniques wrt loops.