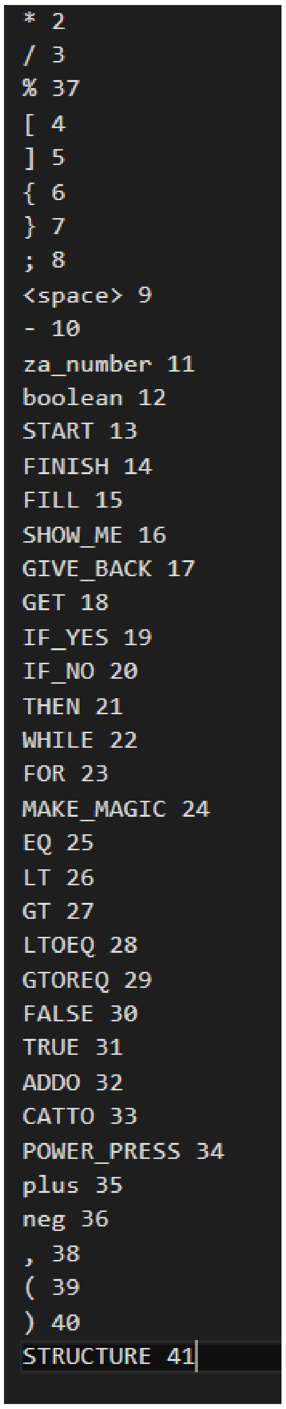
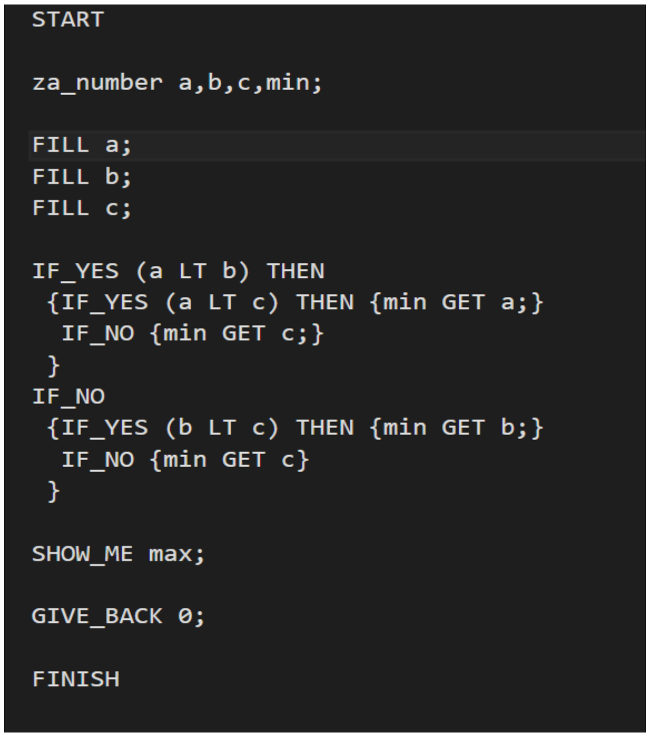
Link to repository: https://github.com/DaniiiiB/Lab2-LFTC

token.in



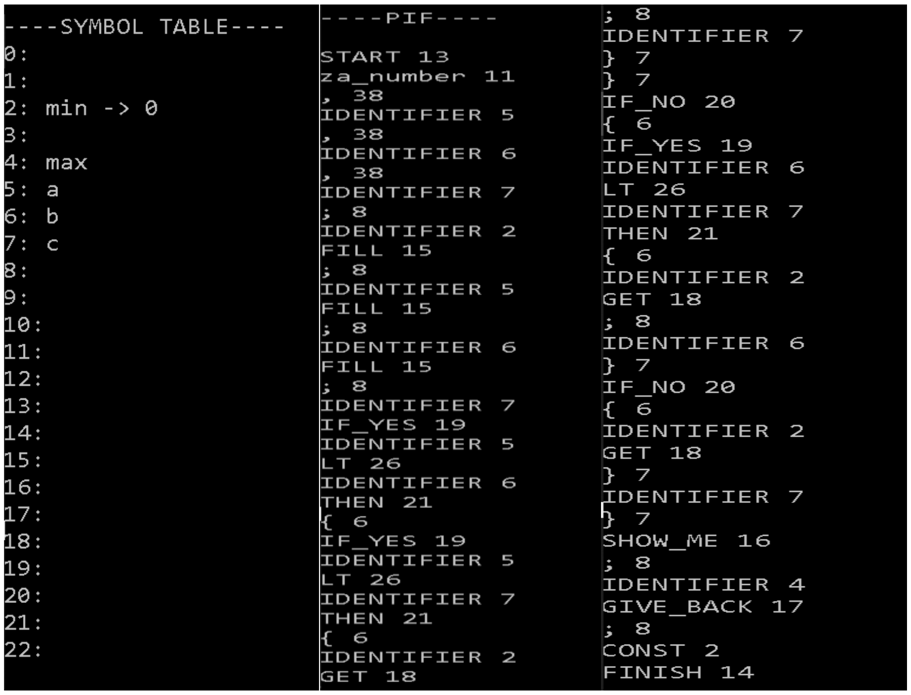
```
* 2
/ 3
% 37
[ 4
] 5
{ 6
} 7
; 8
<space> 9
- 10
za_number 11
boolean 12
START 13
FINISH 14
FILL 15
SHOW_ME 16
GIVE_BACK 17
GET 18
IF_YES 19
IF_NO 20
THEN 21
WHILE 22
FOR 23
MAKE_MAGIC 24
EQ 25
LT 26
GT 27
LTOEQ 28
GTOREQ 29
FALSE 30
TRUE 31
ADDO 32
CATTO 33
POWER_PRESS 34
plus 35
neg 36
, 38
( 39
) 40
STRUCTURE 41
```

p1.lmao - input



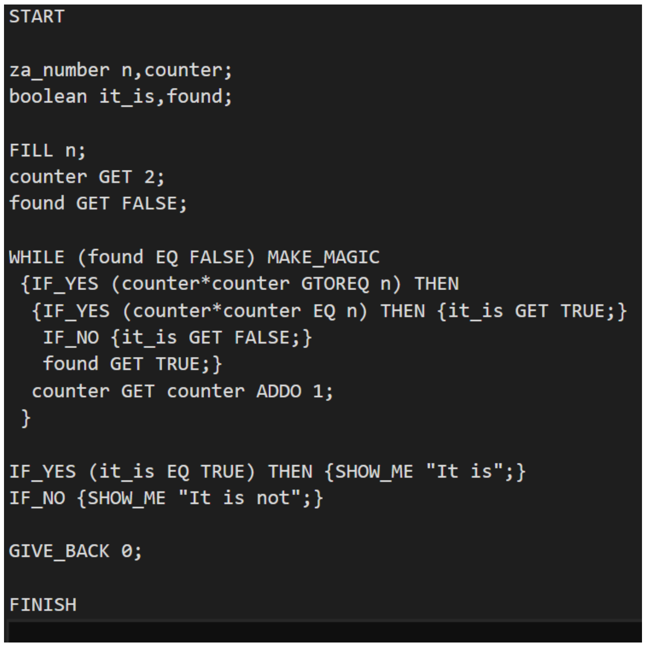
```
START

za_number a,b,c,min;

FILL a;
FILL b;
FILL c;

IF_YES (a LT b) THEN
 {IF_YES (a LT c) THEN {min GET a;}
  IF_NO {min GET c;}
 }
IF_NO
 {IF_YES (b LT c) THEN {min GET b;}
  IF_NO {min GET c}
 }

SHOW_ME max;

GIVE_BACK 0;

FINISH
```

p1.lmao – output



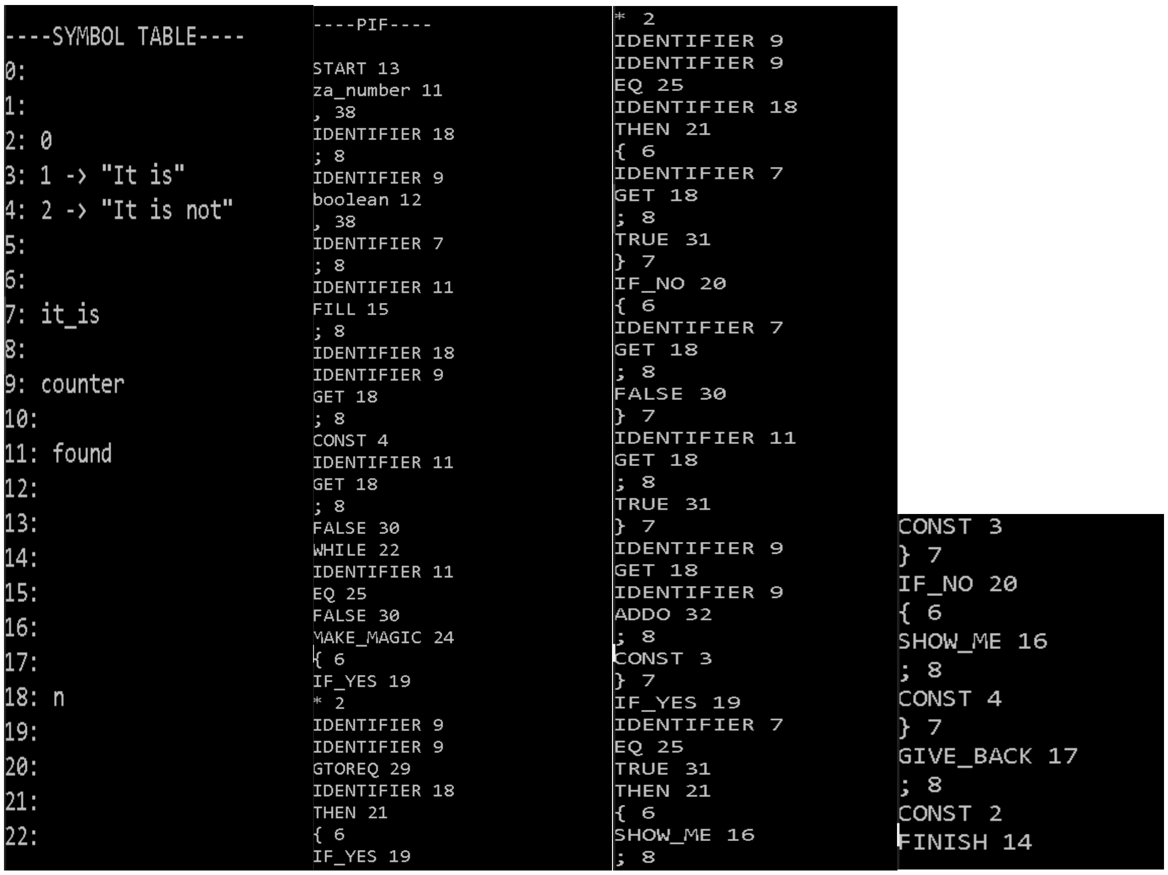
```
----SYMBOL TABLE----   ----PIF----          ; 8
0:                                          IDENTIFIER 7
1:                     START 13             } 7
2: min -> 0            za_number 11         } 7
3:                     , 38                 IF_NO 20
4: max                 IDENTIFIER 5         { 6
5: a                   , 38                 IF_YES 19
6: b                   IDENTIFIER 6         IDENTIFIER 6
7: c                   , 38                 LT 26
8:                     IDENTIFIER 7         IDENTIFIER 7
9:                     ; 8                  THEN 21
10:                    IDENTIFIER 2         { 6
11:                    FILL 15              IDENTIFIER 2
12:                    ; 8                  GET 18
13:                    IDENTIFIER 5         ; 8
14:                    FILL 15              IDENTIFIER 6
15:                    ; 8                  } 7
16:                    IDENTIFIER 6         IF_NO 20
17:                    FILL 15              { 6
18:                    ; 8                  IDENTIFIER 2
19:                    IDENTIFIER 7         GET 18
20:                    IF_YES 19            } 7
21:                    IDENTIFIER 5         IDENTIFIER 7
22:                    LT 26                } 7
                       IDENTIFIER 6         SHOW_ME 16
                       THEN 21              ; 8
                       { 6                  IDENTIFIER 4
                       IF_YES 19            GIVE_BACK 17
                       IDENTIFIER 5         ; 8
                       LT 26                CONST 2
                       IDENTIFIER 7         FINISH 14
                       THEN 21
                       { 6
                       IDENTIFIER 2
                       GET 18
```

p2.lmao – input



```
START

za_number n,counter;
boolean it_is,found;

FILL n;
counter GET 2;
found GET FALSE;

WHILE (found EQ FALSE) MAKE_MAGIC
 {IF_YES (counter*counter GTOREQ n) THEN
  {IF_YES (counter*counter EQ n) THEN {it_is GET TRUE;}
   IF_NO {it_is GET FALSE;}
   found GET TRUE;}
  counter GET counter ADDO 1;
 }

IF_YES (it_is EQ TRUE) THEN {SHOW_ME "It is";}
IF_NO {SHOW_ME "It is not";}

GIVE_BACK 0;

FINISH
```

p2.lmao – output



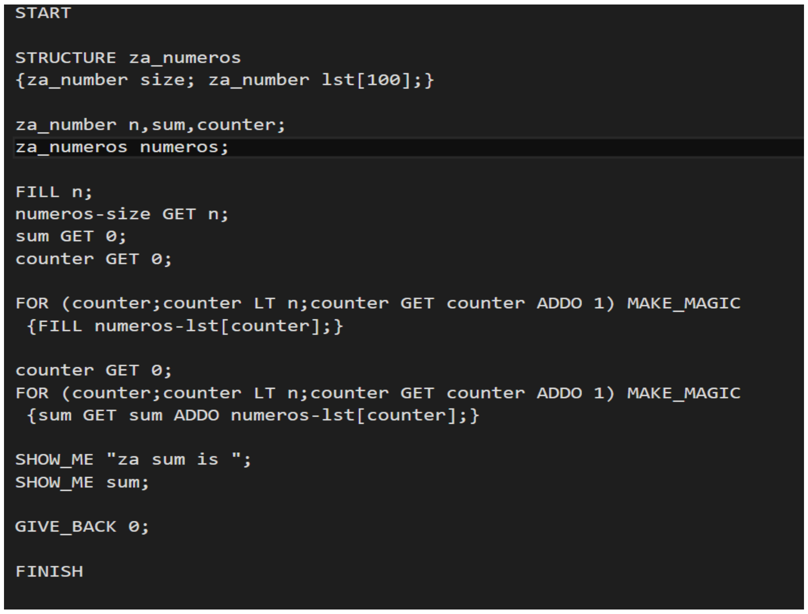
```
----SYMBOL TABLE----
0:
1:
2: 0
3: 1 -> "It is"
4: 2 -> "It is not"
5:
6:
7: it_is
8:
9: counter
10:
11: found
12:
13:
14:
15:
16:
17:
18: n
19:
20:
21:
22:
```

```
----PIF----

START 13
za_number 11
, 38
IDENTIFIER 18
; 8
IDENTIFIER 9
boolean 12
, 38
IDENTIFIER 7
; 8
IDENTIFIER 11
FILL 15
; 8
IDENTIFIER 18
IDENTIFIER 9
GET 18
; 8
CONST 4
IDENTIFIER 11
GET 18
; 8
FALSE 30
WHILE 22
IDENTIFIER 11
EQ 25
FALSE 30
MAKE_MAGIC 24
{ 6
IF_YES 19
* 2
IDENTIFIER 9
IDENTIFIER 9
GTOREQ 29
IDENTIFIER 18
THEN 21
{ 6
IF_YES 19
```

```
* 2
IDENTIFIER 9
IDENTIFIER 9
EQ 25
IDENTIFIER 18
THEN 21
{ 6
IDENTIFIER 7
GET 18
; 8
TRUE 31
} 7
IF_NO 20
{ 6
IDENTIFIER 7
GET 18
; 8
FALSE 30
} 7
IDENTIFIER 11
GET 18
; 8
TRUE 31
} 7
IDENTIFIER 9
GET 18
IDENTIFIER 9
ADDO 32
; 8
CONST 3
} 7
IF_YES 19
IDENTIFIER 7
EQ 25
TRUE 31
THEN 21
{ 6
SHOW_ME 16
; 8
```

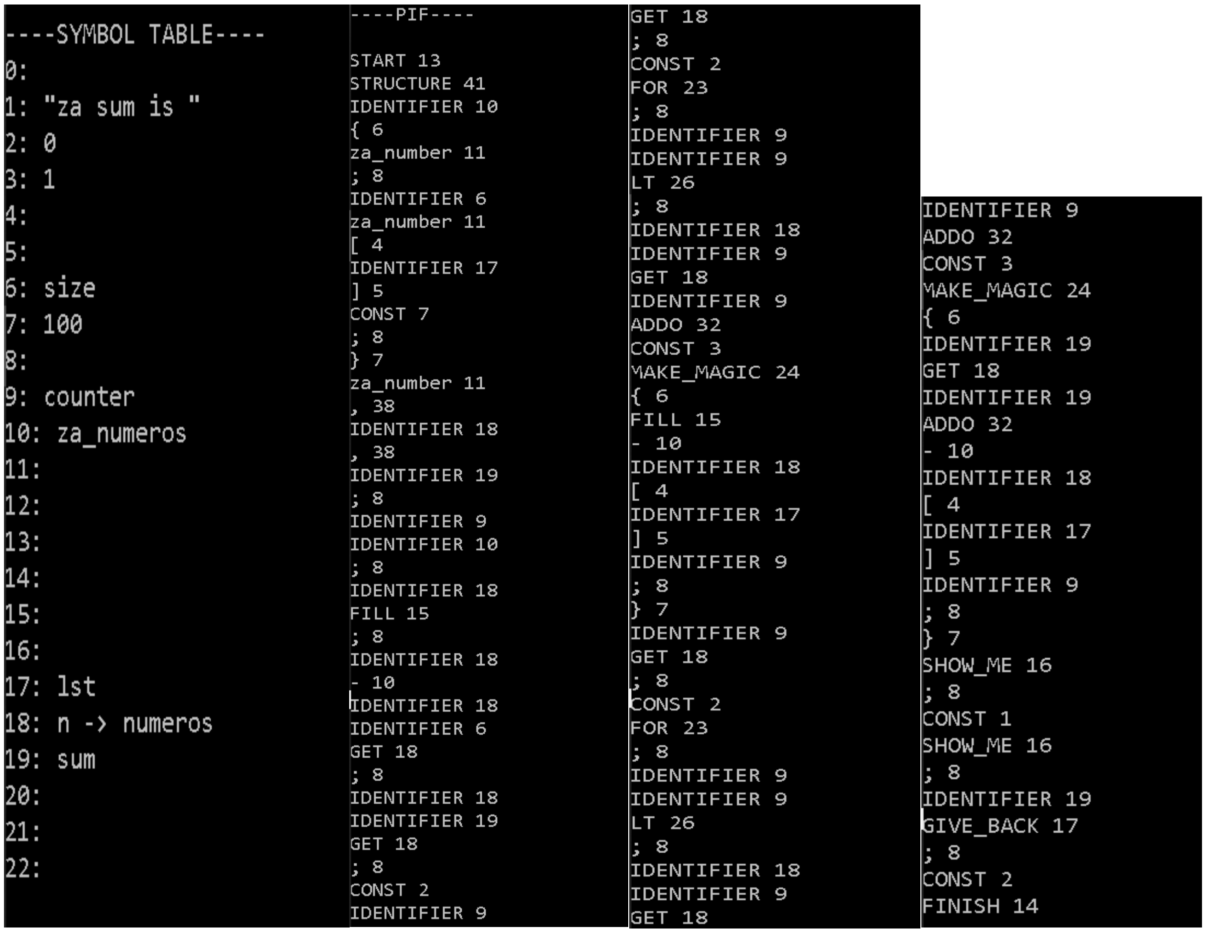
```
CONST 3
} 7
IF_NO 20
{ 6
SHOW_ME 16
; 8
CONST 4
} 7
GIVE_BACK 17
; 8
CONST 2
FINISH 14
```

p3.lmao – input



```
START

STRUCTURE za_numeros
{za_number size; za_number lst[100];}

za_number n,sum,counter;
za_numeros numeros;

FILL n;
numeros-size GET n;
sum GET 0;
counter GET 0;

FOR (counter;counter LT n;counter GET counter ADDO 1) MAKE_MAGIC
 {FILL numeros-lst[counter];}

counter GET 0;
FOR (counter;counter LT n;counter GET counter ADDO 1) MAKE_MAGIC
 {sum GET sum ADDO numeros-lst[counter];}

SHOW_ME "za sum is ";
SHOW_ME sum;

GIVE_BACK 0;

FINISH
```

p3.lmao – output



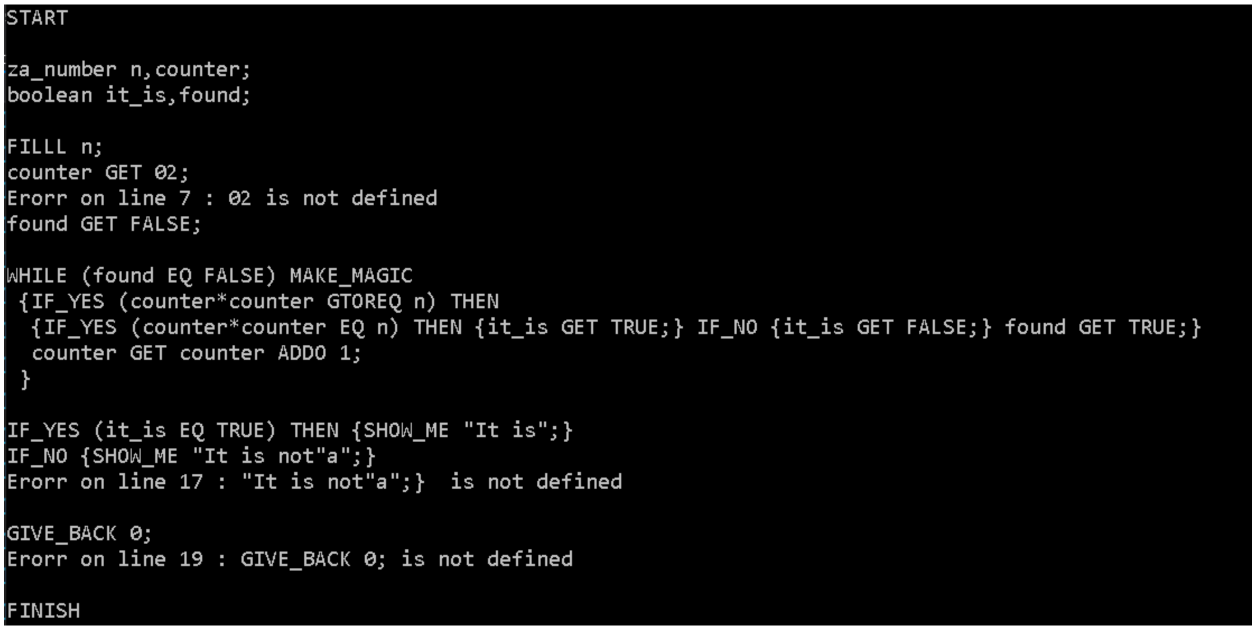
```
----SYMBOL TABLE----
0:
1: "za sum is "
2: 0
3: 1
4:
5:
6: size
7: 100
8:
9: counter
10: za_numeros
11:
12:
13:
14:
15:
16:
17: lst
18: n -> numeros
19: sum
20:
21:
22:
```

```
----PIF----

START 13
STRUCTURE 41
IDENTIFIER 10
{ 6
za_number 11
; 8
IDENTIFIER 6
za_number 11
[ 4
IDENTIFIER 17
] 5
CONST 7
; 8
} 7
za_number 11
, 38
IDENTIFIER 18
, 38
IDENTIFIER 19
; 8
IDENTIFIER 9
IDENTIFIER 10
; 8
IDENTIFIER 18
FILL 15
; 8
IDENTIFIER 18
- 10
IDENTIFIER 18
IDENTIFIER 6
GET 18
; 8
IDENTIFIER 18
IDENTIFIER 19
GET 18
; 8
CONST 2
IDENTIFIER 9
```

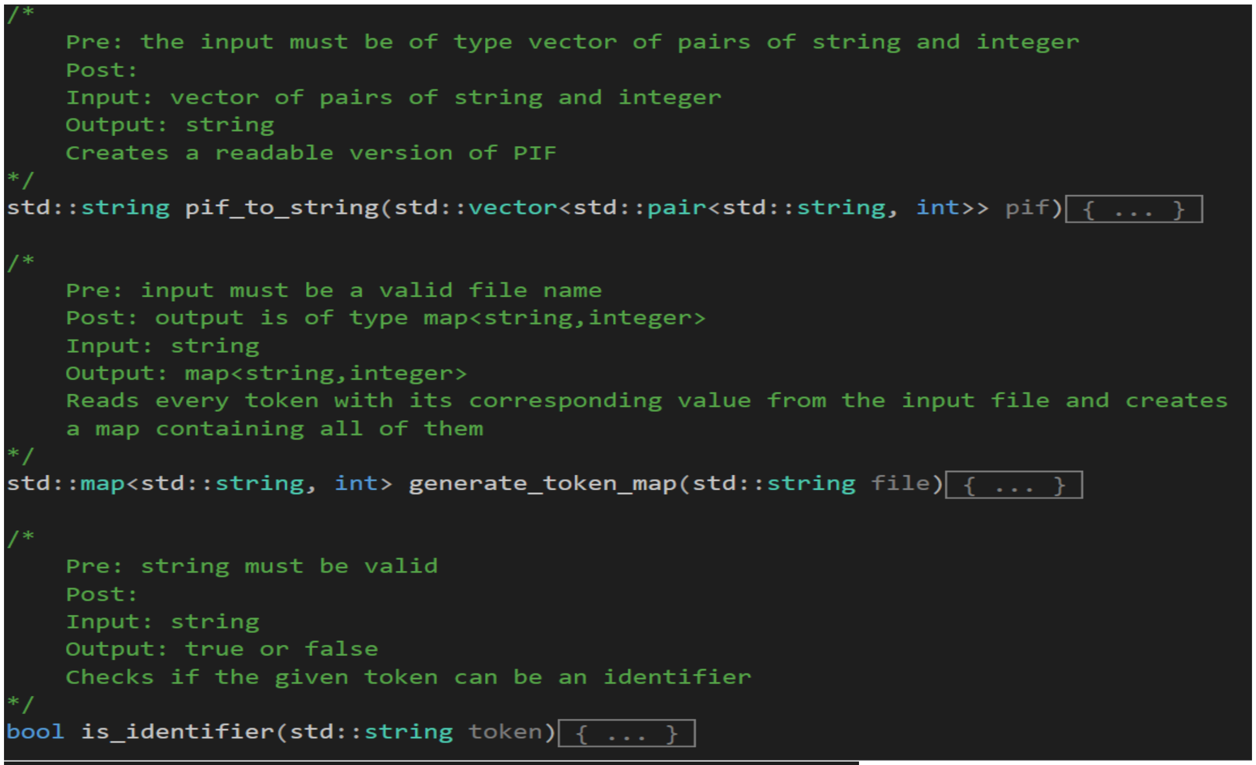
```
GET 18
; 8
CONST 2
FOR 23
; 8
IDENTIFIER 9
IDENTIFIER 9
LT 26
; 8
IDENTIFIER 18
IDENTIFIER 9
GET 18
IDENTIFIER 9
ADDO 32
CONST 3
MAKE_MAGIC 24
{ 6
FILL 15
- 10
IDENTIFIER 18
[ 4
IDENTIFIER 17
] 5
IDENTIFIER 9
; 8
} 7
IDENTIFIER 9
GET 18
; 8
CONST 2
FOR 23
; 8
IDENTIFIER 9
IDENTIFIER 9
LT 26
; 8
IDENTIFIER 18
IDENTIFIER 9
GET 18
```

```
IDENTIFIER 9
ADDO 32
CONST 3
MAKE_MAGIC 24
{ 6
IDENTIFIER 19
GET 18
IDENTIFIER 19
ADDO 32
- 10
IDENTIFIER 18
[ 4
IDENTIFIER 17
] 5
IDENTIFIER 9
; 8
} 7
SHOW_ME 16
; 8
CONST 1
SHOW_ME 16
; 8
IDENTIFIER 19
GIVE_BACK 17
; 8
CONST 2
FINISH 14
```

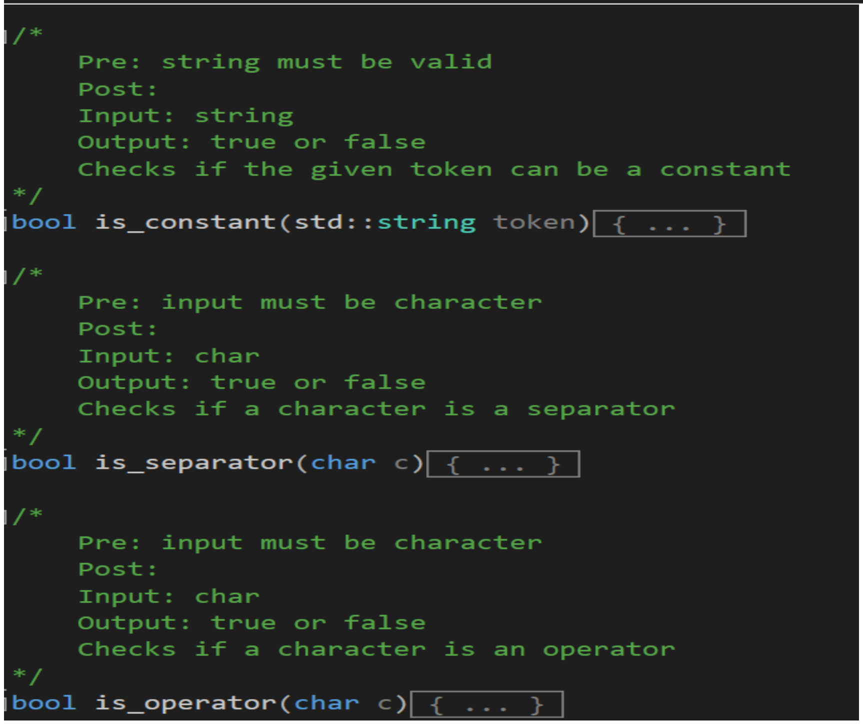
perr.lmao



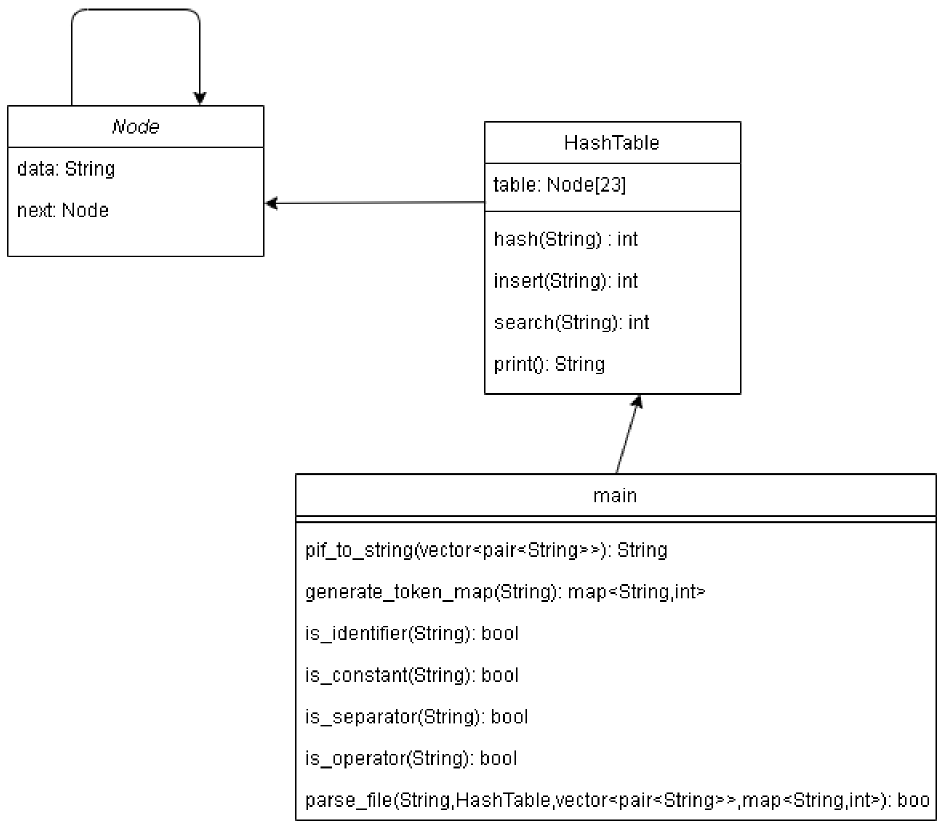
```
START

za_number n,counter;
boolean it_is,found;

FILLL n;
counter GET 02;
Erorr on line 7 : 02 is not defined
found GET FALSE;

WHILE (found EQ FALSE) MAKE_MAGIC
 {IF_YES (counter*counter GTOREQ n) THEN
  {IF_YES (counter*counter EQ n) THEN {it_is GET TRUE;} IF_NO {it_is GET FALSE;} found GET TRUE;}
  counter GET counter ADDO 1;
 }

IF_YES (it_is EQ TRUE) THEN {SHOW_ME "It is";}
IF_NO {SHOW_ME "It is not"a";}
Erorr on line 17 : "It is not"a";}  is not defined

GIVE_BACK 0;
Erorr on line 19 : GIVE_BACK 0; is not defined

FINISH
```



```
/*
    Pre: the input must be of type vector of pairs of string and integer
    Post:
    Input: vector of pairs of string and integer
    Output: string
    Creates a readable version of PIF
*/
std::string pif_to_string(std::vector<std::pair<std::string, int>> pif){ ... }

/*
    Pre: input must be a valid file name
    Post: output is of type map<string,integer>
    Input: string
    Output: map<string,integer>
    Reads every token with its corresponding value from the input file and creates
    a map containing all of them
*/
std::map<std::string, int> generate_token_map(std::string file){ ... }

/*
    Pre: string must be valid
    Post:
    Input: string
    Output: true or false
    Checks if the given token can be an identifier
*/
bool is_identifier(std::string token){ ... }
```



```
/*
    Pre: string must be valid
    Post:
    Input: string
    Output: true or false
    Checks if the given token can be a constant
*/
bool is_constant(std::string token){ ... }

/*
    Pre: input must be character
    Post:
    Input: char
    Output: true or false
    Checks if a character is a separator
*/
bool is_separator(char c){ ... }

/*
    Pre: input must be character
    Post:
    Input: char
    Output: true or false
    Checks if a character is an operator
*/
bool is_operator(char c){ ... }
```


Node
data: String
next: Node
HashTable
table: Node[23]
hash(String) : int
insert(String): int
search(String): int
print(): String
main
pif_to_string(vector<pair<String>>): String
generate_token_map(String): map<String,int>
is_identifier(String): bool
is_constant(String): bool
is_separator(String): bool
is_operator(String): bool
parse_file(String,HashTable,vector<pair<String>>,map<String,int>): boo