Scanner

***Github:*** https://github.com/balean12/FLCD/tree/master/src/first

***Requirement:***

**Statement:** Implement a scanner (lexical analyzer): Implement the scanning algorithm and use ST from [lab 2](https://moodle.cs.ubbcluj.ro/mod/assign/view.php?id=2614" \o "Lab 2) for the symbol table.

**Input:**Programs p1/p2/p3/p1err and token.in (see [Lab 1a](https://moodle.cs.ubbcluj.ro/mod/assign/view.php?id=2546" \o "Lab 1a))

**Output:**PIF.out, ST.out, message “lexically correct” or “lexical error + location”

**Details:**

* ST.out should give information about the data structure used in representation
* If there exists an error the program should give a description and the location (line and token)

***Implementation details:***

* Extracting tokens
  + Take each line at a time.
  + Each line is taken character by character
  + Spaces and tabs are excluded
  + If a separator is found, it adds it in the tokens list
  + If an operator is found, it adds it in the tokens list
  + If a reserved word is found, it adds it in the tokens list
  + If a string is found (in the “”), it adds it in the tokens list
* Scanning algo
  + Reads one line at a time from the file
  + Tokenizes the line and returns a list of tokens
  + Process each token and completes the pif and symbol table
  + Throws error if a lexical error is encountered

***Testing:***

begin main{  
 int a = readk(int);  
 int b = readk(int);  
 int c = readk(int);  
 string s = "vai";  
 int max;  
 if[ a > b && a > c] then:  
 max=a;  
 else if[ b > c ] then:  
 max=b;  
 else:  
 max=c;  
 console\_jkprint(max);  
 a = b + c;  
 c++;  
 int t = +0;  
 end main;  
}

[begin, main, {, int, a, =, readk, (, int, ), ;, int, b, =, readk, (, int, ), ;, int, c, =, readk, (, int, ), ;, string, s, =, "vai", ;, int, max, ;, if, [, a, >, b, &&, a, >, c, ], then, :, max, =, a, ;, else, if, [, b, >, c, ], then, :, max, =, b, ;, else, :, max, =, c, ;, console\_jkprint, (, max, ), ;, a, =, b, +, c, ;, c, ++, ;, int, t, =, 0, ;, end, main, ;, }]

Lexical error at line 13 token console\_jkprint

ProgramInternalForm=[begin=-1, main=-1, {=-1, int=-1, a=71, ==-1, readk=-1, (=-1, int=-1, )=-1, ;=-1, int=-1, b=81, ==-1, readk=-1, (=-1, int=-1, )=-1, ;=-1, int=-1, c=91, ==-1, readk=-1, (=-1, int=-1, )=-1, ;=-1, string=-1, s=101, ==-1, "vai"=131, ;=-1, int=-1, max=111, ;=-1, if=-1, [=-1, a=71, >=-1, b=81, &&=-1, a=71, >=-1, c=91, ]=-1, then=-1, :=-1, max=111, ==-1, a=71, ;=-1, else=-1, if=-1, [=-1, b=81, >=-1, c=91, ]=-1, then=-1, :=-1, max=111, ==-1, b=81, ;=-1, else=-1, :=-1, max=111, ==-1, c=91, ;=-1]}

<< : 20

|| : 23

<= : 15

constant : 1

main : 49

do : 36

program : 41

while : 45

else : 37

if : 38

: 7

== : 21

-- : 31

identifier : 0

! : 24

% : 13

in : 35

& : 26

( : 4

) : 5

\* : 11

+ : 9

then : 44

- : 10

/ : 12

console\_print : 47

: : 8

; : 6

< : 14

!= : 25

= : 16

begin : 48

> : 18

>= : 17

>> : 19

&& : 22

const : 34

string : 50

write\_file : 46

array : 32

of : 40

[ : 0

] : 1

^ : 29

++ : 30

readk : 42

readf : 43

int : 39

char : 33

{ : 2

| : 28

} : 3

~ : 27

Symbol Table

[]

[]

[]

[]

[]

[]

[]

[a]

[b]

[c]

[s]

[max]

[]

["vai"]

[]