

## Lab 1

Zadanie polega na stworzeniu analizatora leksykalnego (skanera) dla prostego języka umożliwiającego obliczenia na macierzach. Analizator leksykalny powinien rozpoznawać następujące leksemy:

- operatory binare: +, -, \*, /
- macierzowe operatory binarne (dla operacji element po elemencie): .+, .-, .\*,
- ./ operatory przypisania: =, +=, -=, \*=, /= operatory relacyjne: <, >, <=, >=,
- !=, ==
- nawiasy: (,), [,], {,} operator zakresu: :
- transpozycja macierzy: ' przecinek i średnik: , ;
- słowa kluczowe: if, else, for, while słowa
- kluczowe: break, continue oraz return
- słowa kluczowe: eye, zeros oraz ones słowa
- kluczowe: print
- identyfikatory (pierwszy znak identyfikatora to litera lub znak \_, w kolejnych znakach mogą dodatkowo wystąpić
- cyfry) liczby całkowite liczby zmiennoprzecinkowe stringi
- 
- 
- Dla rozpoznanych leksemów stworzony skaner powinien zwracać:
- 
- 
- odpowiadający token
- rozpoznany leksem
- numer linii

Następujące znaki powinny być pomijane:

- białe znaki: spacje, tabulatory, znaki nowej linii komentarze: komentarze
- rozpoczynające się znakiem # do znaku końca linii

Przykład.

Dla następującego kodu:

```
A = zeros(5); # create 5x5 matrix filled with zeros
B = ones(7); # create 7x7 matrix filled with ones
I = eye(10); # create 10x10 matrix filled with ones on diagonal and zeros elsewhere
D1 = A.+B' ; # add element-wise A with transpose of B
D2 -= A.-B' ; # subtract element-wise A with transpose of B
D3 *= A.*B' ; # multiply element-wise A with transpose of B
D4 /= A./B' ; # divide element-wise A with transpose of B
```

analizator leksykalny powinien zwracać następującą sekwencję i wypisywać

ją na standardowym wyjściu:

```
(1): ID(A)
(1): =(=)
(1): ZEROS(zeros)
(1): (()
(1): INTNUM(5)
(1): )()
(1): ;(;
(2): ID(B)
(2): =(=)
(2): ONES(ones)
(2): (()
(2): INTNUM(7)
(2): )()
(2): ;(;
(3): ID(I)
(3): =(=)
(3): EYE(eye)
(3): (()
(3): INTNUM(10)
(3): )()
(3): ;(;
(4): ID(D1)
(4): =(=)
(4): ID(A)
(4): DOTADD(.+)
(4): ID(B)
(4): '('(')
(4): ;(;
(5): ID(D2)
(5): SUBASSIGN(-=)
(5): ID(A)
(5): DOTSUB(.-)
(5): ID(B)
(5): '('(')
(5): ;(;
(6): ID(D3)
(6): MULASSIGN(*=)
(6): ID(A)
(6): DOTMUL(.*)
(6): ID(B)
(6): '('(')
(6): ;(;
(7): ID(D4)
(7): DIVASSIGN(/=)
(7): ID(A)
(7): DOTDIV(/.)
(7): ID(B)
(7): '('(')
(7): ;(;
```