```
----------------------------------------------------------------------------------------------------------------------------------------
-----
Lexic.txt

Alphabet:
    - a-z
    - A-Z
    - 0-9

Operators: + - * / % && || ! = == < <= > >=
Separators: {} () ; space newline " , ’
Reserved Words: int, bool, char, string, if, else, while, print, readInt, readString, array, true, false, set, get

-------------------------------------------------------------------
Identifier:

letter = "A" | "B" | ... | "Z" | "a" | ... | "z".
digit = "0" | "1" | ... | "9".
identifier = letter{letter | digit}.

-------------------------------------------------------------------
Constants:

bool = "true" | "false".

char = ’letter’ | ’digit’.

non_zero_digit = "1" | "2" | ... | "9".
int = ["+" | "-"]non_zero_digit{digit} | "0".

string = """{letter | digit | " "}""".


----------------------------------------------------------------------------------------------------------------------------------------
-----
token.in

+
-
*
/
%
&&
||
!
=
==
<
<=
>
>=
{
}
(
)
```

```
;
space
newline
"
,
’
int
bool
char
string
if
else
while
print
readInt
readString
array
true
false
set
get
```

```
------------------------------------------------------------------------------------------------------------------------------------------------------
Syntax.in

program = {declaration}{statement}.

declaration = (simpledeclaration | arraydeclaration) ";".

simpledeclaration = type identifierlist.
type = "int" | "bool" | "char" | "string".
identifier_list = identifier ["=" expression] {"," identifier ["=" expression]}.
expression = intexpression | boolexpression | charexpression | stringexpression.
intexpression = int | identifier | intexpression ("+"|"-"|"*"|"/"|"%") intexpression |
        "(" intexpression ("+"|"-"|"*"|"/"|"%") intexpression ")".
boolexpression = bool | ["!"] identifier | boolexpression ("&&" | "||" | "==" | "<" | "<=" | ">" | ">=") boolexpression |
        "(" boolexpression ("&&" | "||" | "==" | "<" | "<=" | ">" | ">=") boolexpression ")".
charexpression = char | identifier.
stringexpression = string | identifier | stringexpression "+" stringexpression |
        "(" stringexpression "+" stringexpression ")".

arraydeclaration = "array" "[" type "]" "[" positivint "]" simpleidentifierlist.
positivint = non_zero_digit{digit}.
simpleidentifierlist = identifier {"," identifier}.

statement = assignstatement ";" | ifstatement | whilestatement | functionstatement ";".

assignstatement = identifier "=" (expression | functionstatement).
ifstatement = "if" "(" boolexpression ")" "{" {statement} "}" ["else" "{" {statement} "}"].
whilestatement = "while" "(" boolexpression ")" "{" {statement} "}".
functionstatement = functionname "(" [expressionlist] ")".
functionname = "readInt" | "readString" | "get" | "set".
```

expressionlist = expression {"," expression}.

Example

```
int n = 5;
int sum = 0;
int i = 0;
array[int][3] firstNumbers;
int var;
while (i < n)
{
    var = readInt();
    sum = sum + var;
    if (i < 3)
    {
        set(firstNumbers, i, var)
    }
    i = i + 1;
}

print("The first number was ");
print(get(firstNumbers, 0));
```