Language definition

Alphabet

- Upper (A-Z) and lowercase letters (a-z) of English alphabet
- Underline character (_)
- Decimal digits (0-9)

Lexic

- Special symbols, including:
  - Operators + - * / = < <= == >=
  - Separators [ ] { } : ; \n
  - Reserved words: include using void int float string bool return for while if else

- Identifiers

  A sequence of letters and digits, such that the first character is a letter; the rule is:

Identifier = letter | letter(letter | digit | zero) | letter(letter | digit)(letter | digit | zero) | letter(letter | digit | zero)(letter | digit | zero)(letter | digit | zero) | letter(letter | digit | zero)(letter | digit | zero)(letter | digit | zero)(letter | digit | zero) | letter(letter | digit | zero)(letter | digit | zero)(letter | digit | zero)(letter | digit | zero)(letter | digit | zero) | letter(letter | digit | zero)(letter | digit | zero)(letter | digit | zero)(letter | digit | zero)(letter | digit | zero)(letter | digit | zero) | letter(letter | digit | zero)(letter | digit | zero)(letter | digit | zero)(letter | digit | zero)(letter | digit | zero)(letter | digit | zero)(letter | digit | zero)

Letter = lowercase_letter | uppercase_letter

Lowercase_letter = “a” | “b” | … | “z”

Uppercase_letter = “A” | “B” | … | “Z”

digit = “1” | … | “9”

zero = “0”

- Constants
    - Integers

      ```
      Integer = +number | -number | number
      Number = (digit | digit zero) {number}
      ```

    - Real values

      ```
      Real_value = -number | number
      Number = integer.real_part
      Real_part = sequence_of_zeros number
      Sequence_of_zeros = {0}sequence_of_zeros
      ```

    - Character

      ```
      Character = ‘letter’ | ‘digit’ | ‘zero’
      ```

    - Bool

      ```
      Bool = false | true
      ```

    - String

      ```
      String = “string_literal”
      String_literal = char{string_literal}
      Char = letter | digit | zero
      ```

Syntax

- Syntactic rules

```
program = include_statement define_main statement finish_program

include_statement = "#include<iostream>"
define_main = "int main() {"
finish_program = "return 0; }"
```

```
single_statement = declaration_statement | assignment_statement | conditional_statement | loop_statement | read_statement | write_statement
```

```
declaration_statement = primitive_declaration | array_declaration ";"

assignment_statement = identifier "=" expression ";"    statement = single_statement | single_statement{statement}

conditional_statement = "if (" expression ") {" statement "}" "else {" statement "}"

loop_statement = "while (" expression ") {" statement "}"

read_statement = "std::cin >> " identifier ";"

write_statement = "std::cout << " expression " << std::endl;"

expression = integer_expression | float_expression | char_expression | bool_expression

integer_expression = integer_constant | identifier | "(" integer_expression ")" | integer_expression arithmetic_operator integer_expression

float_expression = float_constant | identifier
 | "(" float_expression ")" | float_expression arithmetic_operator float_expression

arithmetic_operator = "+" | "-"

term = term (“*” | “/” | “%”) expression | expression

char_expression  = char_constant | identifier

bool_expression = bool_constant | identifier  | "(" bool_expression ")" | bool_expression relational_operator bool_expression

relational_operator = "<" | ">" | "<=" | "==" | ">=" | "!="

primitive_type = "int" | "float" | "char" | "bool"

primitive_declaration = primitive_type identifier

array_declaration = primitive_type identifier "[" integer_constant "]"


structure_declaration = “struct” identifier “{“ declaration_list “}”

declaration_list = declaration_statement | declaration_statement {declaration_list}
```

Symbol Table

Token Type	code
identifier	0
constant	1
program	2
array	3
int	4
bool	5
float	6
char	7
std::cin	8
std::cout	9
if	10
else	11
while	12
;	13
=	14
+	15
-	16
==	17
!=	18

<	19
>	20
>=	21
<=	22
/	23
*	24
%	25
]	26
[	27
}	28
{	29
)	30
(	31
.	32
struct	33

Sample program

Compute the sum of `N` real numbers.

```
 1 #include <iostream>
 2
 3 int main() {
 4   int N;
 5   std::cin >> N;
 6
 7   float SUM;
 8   SUM = 0;
 9
10   int I;
```

```
11    I = 0;
12
13    while (I < N) {
14          float NUM;
15          std::cin >> NUM;
16          SUM = SUM + NUM;
17          I = I + 1;
18    }
19
20    std::cout << SUM << std::endl;
21    return 0;
22}
```