Mini-Language Specification

Alphabet

- A-Z, a-z (Uppercase and lowercase letters)
- 0-9 (Digits)
- _

Operators

- +, -, *, /, % (Addition, Subtraction, Multiplication, Division, … - Arithmetic)
- ==, !=, >, < (Equality, Inequality, Greater, Equal - Relational)
- &&, ||, ! (And, Or, Not - Logical)
- = (Assignment)

Separators (for this document's readability each separator will be in "")

- ";", ",", " ", "{", "}", "(", ")"

Keywords

- read
- write
- if
- else
- for
- while
- break
- integer
- string
- character
- array
- return

Identifiers

- identifier = letter {letter|digit} | letter
- letter = "A"| "B"| …| "Z"| "a"| "b"| …|"z"
- digit = "0"| non_zero_digit
- non_zero_digit = "1"| … "9"

Constants

- integer = “0” |[“+”|”-“] non_zero_digit {digit}
- character = ‘letter’|’digit’
- string = “{letter|digit}”

Token
[
]
{
}
;
:
,
<
>
==
!=
!
&&
||
=
+
-
*
/
%
character
integer
string
array
if
else
for
while
break
return
read
write
start

Syntax

- program = “start” compound_statement
- declaration = type “ “ identifier
- simple_type = “integer” | “string” | “character”
- array_declaration = simple_type “array” “[“ integer “]”
- type = simple_type | array_declaration
- compound_statement = “{“ statement_list “}”
- statement_list = statement| statement “;” statement_list
- statement = simple_statement | struct_statement
- simple_statement = assign_statement | io_statement | declaration
- struct_statement = compound_statement | if_statement | while_statement | for_statement
- if_statement = “if” condition statement [“else” statement]
- for_statement = “for” “(“ “number” assign_statement “;” condition “;” assign_statement “)” statement
- while_statement = “while” condition statement
- assign_statement = Identifier “=“ expression
- expression = [expression (“+”|”-”)] term
- term = term (“*” | “/”) factor | factor
- factor = “(“ expression “)” | integer | Identifier| Identifier “[“ integer “]”
- io_statement = (“read” IDENTIFIER) | (“write” (Identifier| Constant))
- condition = “(“ expression relation expression “)”
- relation = “<“ | “<=“ | “==“ | “!=“ | “>=“ | “>“

Examples

P1.
Requirement: Compute the maximum number out of 3 input numbers and display it.
Solution:

```
start {

integer a;
integer b;
integer c;
integer max;

read a;
read b;
read c;
if(a>b&&a>c){
        max=a;
}
else{
        if(b>c&&b>a){
                max=b;
        }
        else{
                max=c;
        }
}

write max;
}
```

P2.

Requirement: Check if an input is a prime number.

Solution:

```
start{

integer a;
integer i;
integer is_prime;

is_prime=0;
read a;

for(i=2;i<a;i=i+1){
        if(a%i==0){
                is_prime=1;
                break;
        }
}

if(is_prime==1){
        write "a is prime"
}else{
        write "a is not prime";
}
}
```

P3.
Requirement: Compute the sum of n input numbers which are bigger than m (another input number) and display it.
Solution:

```
start{

integer n;
integer m;
integer sum;
integer current_number;

sum=0;
read n;
read m;

for(i=0;i<n;i=i+1){
            read current_number;
            if(current_number>m){
                        sum=sum+current_number;
            }
}

write sum;
}
```

P1err.
Requirement: Compute the sum of 2 input numbers and display it.
Solution:

```
start{

integer 2a; <- lexical error
integer b;
integer sum;

sum=0;
read a;
read b;

sum+=b;             <- lexical error
sum=sum+a;

write sum;}
```