COMS 4115

Programming Languages and Translators

ASML: Language Reference Manual

10/28/2003

Davidov, Olga
olga.davidov@riag.com

Introduction

ASML is a small language for array manipulation based on C. It uses ANTLR to specify its grammar and Java on the back end to implement the grammar.

Tokens

There are five kinds of tokens: identifiers, keywords, operators, literals, and other separators. Spaces, new lines and tabs constitute white space. The purpose of white space is to separate tokens, otherwise it is ignored. A token is a longest possible sequence of characters that may be matched to any token.

Comments

Both C and C++ style comments are supported. Multiline comments start with /* and end with */ sequence of characters. Single line comments start with // and continue to the end of the line.

Identifiers

Identifier is a sequence of letters and digits which start with a letter. The first 100 symbols are significant.

Keywords

The following keywords are reserved in ASML:

program
const
int
float
string
if
elsif
else
while
return

Literals

There are integer literals, floating point literals and string literals. Integer literal is a sequence of one or more digits, optionally signed. Floating point literal is also optionally signed sequence of digits followed by a dot, followed by another sequence of digits. A string literal is a sequence of zero or more characters enclosed in double quotes. Double quote followed by another double quote is used to represent literal double quote character inside a string.

Types

ASML has 3 basic scalar types: int, float, string. Int and float types are equivalent to those in C, and string type is equivalent to char array in C (char[])

The only user-defined type is array type. Only one-dimensional arrays are allowed. Arrays can be indexed only by integers.

When arithmetic operations are performed on operands of integer and float types, the integer operand is converted to float.

The string type is only used for built-in I/O functions.

Operator Precedence

The following table lists operators in ASML in order of decreasing precedence.

[] ()	Array element dereference, function call
!	Logical negation
- +	Unary adding operators
* /	Binary multiplicative operators
- +	Binary additive operators
== < <= > >=	Relational operators
&& ||	Logical operators
=	Assignment operator

Declarations

Variables and constants must be declared before they are used. A variable declaration has the following form:

```
<type> <list of optionally initialized identifiers>;
```

For example:

```
int i, j;
float x, y;
int A[10];
float q = 3.0;
```

A constant declaration has the following form:

```
const <type> <list of initialized identifiers>;
const float pi = 3.1415;
```

Statements

Statements can be either simple or compound. A simple statement is an expression; compound statement is a sequence of statements in curly braces:

```
{
        statement
        …
        statement
}
```

Conditional statement

The following id the syntax for the if statement:

```
if (expression)
        statement
elsif (expression)
        statement
….
else
        statement
```

Looping statement

The following id the syntax for the while loop:

```
while (expression)
{
        statement
}
```

Return statement

Return statement is used to return control from a function to the caller. In ASML every function must return a value. Return statement has the following form:
return expression;

Scope

There are two levels of scoping in ASML: global and local. Identifiers can either be declared at the top of the file, in which case they are global or in the header of a function, in that case they are local to that function. Nested functions are not allowed, therefore there's only one local scope. Functions must be declared before they are used.

Built-in Functions

Two functions are used for I/O: print and read.
print function takes a list of arguments; Each argument is converted to a string and they are concatenated together and printed out to standard output.

read function accepts only int or float argument and attempts to read a value from standard input into specified variable.

The following array-manipulation functions are predefined:
min - returns min element
max - returns max element
length - returns the length
median - returns median value of the elements
avg – return average value of the elements
swap - swaps 2 element with given indices
slice - returns an array of elements staring from given position and containing specified number of elements

Functions

Only functions are supported in ASML. Functions can accept parameters and must return a value. Parentheses after the function name are required even if the parameter list is empty. Both return value and parameters must be of simple types: int or float. Parameters can be passed by value or by reference. The default is by value, to be passed by reference, the variable name should be preceded by the backslash '\' in the actual parameter list.

Sample Program

```
program abc
        const int N = 10;
        int i;
        float x;
        int grades[N];
        function sort(int[]);
{
        i = 0;
        while (i < length(grades))
        {
                read(ids[i]);
        }

        x = avg(grades);

        print("Average grade is ", x);

        print("Here are the grades in the sorted order:");

        sort(grades);

        i = 0;
        while (i < length(grades))
        {
                print(grades[i]);
        }
}
```

```
int sort(int[] array)
    int i, j, max;
{
    i = 0;
    min = 0;
    while (i < length(array))
    {
        j = i;
        while (j < length(array))
        {
            if (array[j] < array[min])
            {
                min = j;
            }
            j = j + 1;
        }
        swap(i, min);
        i = i + 1;
    }
}
```