METU—CENG 444, LANGUAGE PROCESSORS, FALL 2005

\textit{F}^3 \text{ Programming Language}

1 Syntax:

\begin{align*}
\text{program} & \rightarrow \text{declaration_list} \ \text{function_list} \\
\text{function_list} & \rightarrow \text{function} \mid \text{function_list} \ \text{function} \\
\text{function} & \rightarrow \text{basic_type} \ \text{func} \ \text{id} \ ((\text{parameter_list})) \ \text{function_body} \ \text{endfunc} \\
\text{function_body} & \rightarrow \text{declaration_list} \ \text{statement_list} \\
\text{declaration_list} & \rightarrow \text{declaration} \ \text{statement_list} \ | \ \epsilon \\
\text{declaration} & \rightarrow \text{var} \ \text{variable_list} \\
\text{parameter_list} & \rightarrow \text{variable_list} \ | \ \epsilon \\
\text{variable_list} & \rightarrow \text{id} : \text{type} \mid \text{variable_list}, \ \text{id} : \text{type} \\
\text{type} & \rightarrow \text{basic_type} \ \text{array_extension} \\
\text{basic_type} & \rightarrow \text{int} \ | \ \text{real} \\
\text{array_extension} & \rightarrow \ [ \ \text{num} \ ] \ | \ \epsilon \ | \ [ \ ] \\
\text{statement_list} & \rightarrow \text{statement} \ | \ \text{statement_list} \ \text{statement} \ | \ \epsilon \\
\text{statement} & \rightarrow \text{assignment_statement} \mid \text{return_statement} \mid \text{print_statement} \\
& \mid \text{read_statement} \ | \ \text{null_statement} \\
& \mid \text{if_statement} \mid \text{while_statement} \\
\text{assignment_statement} & \rightarrow \text{variable} := \ \text{expression} \\
\text{variable} & \rightarrow \text{id} \mid \text{id}[\ \text{expression}] \\
\text{expression} & \rightarrow \text{simple_expression} \mid \text{simple_expression} \ \text{relop} \ \text{simple_expression} \\
\text{simple_expression} & \rightarrow \text{term} \ | \ \text{sign} \ \text{term} \mid \text{simple_expression} \ \text{addop} \ \text{term} \\
\text{term} & \rightarrow \text{factor} \mid \text{term} \ \text{mulop} \ \text{factor} \\
\text{factor} & \rightarrow \text{variable} \mid \text{id} \ (\ \text{argument_list}) \ | \ \text{num} \mid (\ \text{expression}) \mid \text{unary_operator} \ \text{Expression} \\
\text{sign} & \rightarrow + \mid - \\
\text{unary_operator} & \rightarrow - \\
\text{argument_list} & \rightarrow \text{expression_list} \ | \ \epsilon \\
\text{expression_list} & \rightarrow \text{expression} \mid \text{expression_list}, \ \text{expression} \\
\text{print_statement} & \rightarrow \text{print} \ \text{print_list}
\end{align*}
print_list → print_item | print_list , print_item
print_item → expression | ” text ”
text → text_character | text_character text
text_character → any printable ASCII character | special_character
special_character → ‘n
read_statement → read read_list
read_list → read_item | read_list , read_item
read_item → ” text ” | variable
return_statement → return expression
null_statement → null

if_statement → if expression then statement_list endif
                | if expression then statement_list else statement_list endif

while_statement → while expression do statement_list endwhile

2 Lexical Conventions:

1. Comments are surrounded by /* and */. They can occupy more than one line but they can not be nested.

2. mulop stands for the operators: *, /, mod and div.

3. addop stands for the operators: + and −.

4. relop stands for the operators: =, <>, <, <=, >= and >.

5. The language is case sensitive and the keywords are reserved. They appear boldface in the grammar. They are:

func endfunc int real return null
mod div if then else endif
do print read while endwhile

6. Token id stands for identifiers. An identifier is a sequence of letters or digits that should start with a letter. There is no language specific restriction on the length of them, but you can assume identifier length to be at most 32. The remaining characters, if any, are ignored.

7. Token num stands for unsigned numbers. A number is composed of three parts. First part is a sequence of digits, the integral part of the number. It may be optionally followed by a decimal point and a sequence of digits i.e.
the fraction part. Lastly the exponent part comes, starting with an e or E
followed by optional \( \pm \) and a sequence of digits, i.e. the value of the
exponent. This follows the C style decimal number notation. On the other
hand, a preceding 0 does not mean an octal number as it does in C, there is
no support for different number bases.

8. Literal strings (i.e. text) may contain double quotes as data. In this case, the
quotes are escaped, e.g. "this is \" a quote". Backslash can be
written as \\\. 

9. The text item in the read function is used to prompt the user.

3 Semantics:

1. A program written in \( F^3 \) commences by executing the first function defined.
   Global variables are declared at the top, before any function definition takes
   place. A variable has the scope over the block over which it is defined. The
   statements of the function are executed in turn until a return statement or
   the end of the function body is encountered. Upon reaching the end of the
   function body with no return statement, the return value is undefined.

2. Boolean type is not supported. When an expression is used as a condition
   for if or while statements, a value of zero means false and non-zero means
   true.

3. The null statement does nothing.

4. Array declarations follow C semantics. The number indicates the number
   of elements, they are indexed as 0 through number \(-\) 1.

5. The expression in [ expression] must be integer-valued (of int type).

6. Array extension [ ] can only be used in a parameter list.

7. An array parameter with no bound specification ([ ]) can accept array argu-
   ments of any (fixed) size.

8. The special character \n causes a newline to be printed.

X No function can be used before its declaration takes place.

10. Parameters to functions are always passed by value.
11. $F^3$ allows mixed mode arithmetic where \texttt{int} values are promoted to \texttt{real}. Forcing a real value to integer, however, is not allowed.

12. \texttt{read} and \texttt{div} apply to \texttt{int}-valued operands only.

4 History

The name $F^3$ stands for “Functions, functions, functions”, or “four-fourty-four” depending on mood and taste. Other uses of f-words are strictly unofficial.

The best project wins the prestigious Annual \textbf{Gazozuna Compiler Projesi}® Award, to be delivered during the end of year ceremony in May (seriously). This is the third year of the award, and nobody has won it more than once. The award has been granted by a select committee of impartial judges, namely, me.

Happy compiling,

-Cem Bozsahin