New Assignment 1: Sketch of solution

Consider the following three language constructs of Pascal, C, and Fortran 77, respectively.

Pascal:

\[
\text{function max (i, j: integer) : integer;}
\]
\[
\text{\{return maximum of integers i and j\}}
\]
\[
\begin{align*}
\text{begin} & \\
\text{if i > j then return max := i} & \\
\text{else max := j} & \\
\text{end;}
\end{align*}
\]

C:

\[
\text{int max (i, j)}
\]
\[
\text{int i, j;}
\]
\[
\text{/* return maximum of integers i and j */}
\]
\[
\begin{align*}
\text{\{ & \\
\text{return i>j?i:j;} & \\
\text{}}
\end{align*}
\]

Fortran 77:

\[
\text{FUNCTION MAX (I, J)}
\]
\[
\text{RETURN MAXIMUM OF INTEGERS I AND J}
\]
\[
\text{IF (I .GT. J) THEN}
\]
\[
\text{MAX = I}
\]
\[
\text{ELSE}
\]
\[
\text{MAX = J}
\]
\[
\text{END IF}
\]
\[
\text{RETURN}
\]

1. (50 points) Identify the lexemes that make up the tokens in the above programs. For each lexeme thus identified provide a specification in the form \(LHS = RHS\) where LHS is the token you have identified and RHS is a regular expression over the language alphabet whose language contain the lexeme.
Answer:

1. Question 1:

- **Pascal:**

  id: max, i, j
  keyword: function, integer, begin, if, then, return, else, end
  parantheses: (, )
  separators: , , `,`
  relations: >
  operators: :=
  comment: {return maximum of integers i and j}

- **C:**

  id: max, i, j
  keyword: int, return
  parantheses: (, ), {, }
  separators: :, ;, `'`
  relations: >, ?
  comment: /* return maximum of integers i and j */

- **Fortran 77:**

  id: MAX, I, J
  keyword: FUNCTION, IF, THEN, ELSE, END IF, RETURN
  parantheses: (, )
  separators: , , `
  relations: .GT.
  operators: =
  comment: C RETURN MAXIMUM OF INTEGERS I AND J

2. Question 2:

- **Pascal:**

  ">" = self
  ":=" = self
  "function" = self
  "integer" = self
  "begin" = self
  "if" = self
  "then" = self
  "return" = self
  "else" = self
  "end" = self
  "parenthesis" = body: Token=o and Lex = "(" | Token = O and Lex = ")"
  "separator = body: Token = O and Lex = ":" | Token = O and Lex = ";"
"operator" = body: Token = 0 and Lex = ":" Token = 0 and Lex = ":=
"comment" = begin: Token=0 and Lex="{"
    body: any_until_end
end: Token=0 and Lex="}"
"id" = body: Token = I

• C:

"int" = self
"return" = self
"parenthesis" = body: Token= 0 and Lex = "(" | Token = 0 and Lex = ")"
    | Token = 0 and Lex = ":" | Token = 0 and Lex = ":=
"separator = body: Token = 0 and Lex = ";" | Token = 0 and Lex = ","
"relations"= body: Token = 0 and Lex = "<" | Token = 0 and Lex = "="
"comment" = begin: Token=0 and Lex="/" Token = 0 and Lex = ":*"
    body: any_until_end
end: Token = 0 and Lex = ":*" Token = 0 and Lex = "/"
"id" = body: Token = I

• Fortran 77

"id" = body: Token = I
"FUNCTION" = self
"IF " = self
"THEN" = self
"ELSE" = self
"END" = self
"RETURN" = self
"parenthesis" = body: Token= 0 and Lex = "(" | Token = 0 and Lex = ")"
"separators" = body: Token = 0 and Lex = ","
"relations" = body: Token = 0 and Lex = ":=" Token = I and Lex = "GT"
    Token = 0 and Lex = ":="
"operators" = body: Token = 0 and Lex = "="
"comment" = begin: Token = I and Lex = "C" and Col = 1
    body: any_until_end
end: Token = U and Lex = "\n"

Duedate: Thursday 9 February 2006