PROGRAM FLOW CONTROL IN WOLFRAM MATHEMATICA

LACKOVIC, A[ndrej]

Abstract: Program flow control in Mathematica is usually done using two mechanisms. The first mechanism is a conditional statement based on the If command, while the second mechanism is a loop mechanism. There are two basic loop structures in Mathematica: the DO structure and the WHILE structure. Conditional expressions within these structures are examined using relational and logical operators.

Key words: flow control, operators, conditional expression, loop

1. INTRODUCTION

When solving certain problems, it is necessary to repeat a series of commands or switch to a specific location in the program depending on the given parameters, in other words, to have the necessary control over the program execution flow. Basic mechanisms controlling the program execution flow, which can be used with almost all programming languages, are loops, branches and jumps. Apart from these, Mathematica includes other management structures to facilitate writing of diverse simple and effective programs.

2. RELATIONAL AND LOGICAL OPERATORS

Relational operators (comparison operators) compare two numbers and determine whether the comparing statement is "true" ("correct") or "false" ("incorrect"). Logical operators examine expressions whose values may be true or false. Relational and logical operators can be used in mathematical expressions, and are often used in combination with other operators when making management related decisions during the execution of the program. (Wolfram, 2008.)

<table>
<thead>
<tr>
<th>Relational operator</th>
<th>description</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;</td>
<td>Less than</td>
</tr>
<tr>
<td>&gt;</td>
<td>More than</td>
</tr>
<tr>
<td>&lt;=</td>
<td>Equal to or less than</td>
</tr>
<tr>
<td>=&gt;</td>
<td>Equal to or more than</td>
</tr>
<tr>
<td>==</td>
<td>Equal to</td>
</tr>
<tr>
<td>!=</td>
<td>Different from</td>
</tr>
</tbody>
</table>

Tab. 1. Relational operators

<table>
<thead>
<tr>
<th>Logical operator</th>
<th>description</th>
</tr>
</thead>
<tbody>
<tr>
<td>&amp;&amp;</td>
<td>If both have the value true, the result is true; otherwise the result is false</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td>!</td>
<td>Gives a value opposite of the operand value; the result is true if the operand is false, and if the operand is true, the result is false</td>
</tr>
</tbody>
</table>

Tab. 2. Logical operators

3. FLOW CONTROL COMMANDS

3.1 Conditional statements

Conditional statement is a command that allows the program to decide whether to perform a group of commands following the statement for conditional execution, or to skip these commands. (Wolfram, 2008.)

IF – END structure:

If[conditional expression, command group]

If the conditional expression has the value true, the program executes commands following the comma. If the conditional expression is false, the program skips the group of commands after the comma, and continues to perform the commands after .

IF – ELSE – END structure:

If[conditional expression, command group (1), command group (2)]

If the value of the conditional expression is true, the program executes the first group of commands after the comma, and if the value of the conditional expression is false, the program executes the second group of commands (after the second comma).

ELSEIF structure:

If[condition(1), commands (1),
   If[condition(2), commands (2), ...]
]

If the conditional expression has the value true, the program executes the first group of commands after the comma, and if the conditional expression of the statement has the value false, the program moves to a new statement If, etc.

Fig. 1. Source code of Example 1
Example 1: Along with default parameters a, b and c of the quadratic equation
\[ ax^2 + bx + c = 0 \]  
the program displays two real solutions, one double solution or no real solutions.

3.2. Loops
Mathematica supports two loop structures: DO and WHILE (Maeder, 1997.)

DO structure:
\[
\text{Do[loop body, \{s, s_{min}, s_{max}, \text{step}\}]}
\]
The body of the loop is executed until s takes integer values from \( s_{min} \) to \( s_{max} \) with the default step.

Example 2: The program takes vector (list) as argument and returns the sum of its elements.

FOR structure:
\[
\text{For[start, conditional expression, step, loop body]}
\]
The body of the loop is executed until the value of the conditional expression is true.

Example 3: The program takes one real number as argument and returns the sum of first n natural numbers, sum of squares of n natural numbers and sum of cube n natural numbers.

Example 4: The program loads the numbers one by one as arguments using the Input command, and specifies the maximum number loaded. The loading process is stopped when number 0 is loaded.

4. CONCLUSION
Only the programming in Mathematica software package is called rule-based programming. As regarding the program execution flow control, there is a standard algorithm that we could program in any procedural language like Fortran or C. However, Mathematica has a very simplified syntax for using commands. In addition, the advantage is that it enables us to use software package functions (completed functions within Mathematica) within a standard algorithm, which in most other programming languages would need to be written separately, especially in the case of mathematical functions. In future research plan is to study additional options of commands for flow control of the program, and make detailed comparisons with other programming languages (like C, C++, C#).

5. REFERENCES