1. What are the boolean values in your language? (e.g., True and False, true and false, 1, and 0, etc.)
   1. Technically, Fortran classifies Booleans as ‘Logicals’ but they generally perform the same way as Booleans in other languages. They are .TRUE. for true, and .FALSE. for false. You can also classify these with unique variable names as well, for instance: “ LOGICAL :: yes, no
2. What types of conditional statements are available in your language?  (if/else, if/then/else, if/elseif/else).  Does your language allow for statements other than “if” (for example, Perl has an “unless” statement, which does the opposite of “if”!)
   1. Fortran supports a general, wide range of conditional statements but in the form of if/then/else and if/then/elif/then/else. It does not seem that Fortran supports any other statements other than ‘if’, however it does require a formal ending to the conditional statement with an ‘endif’.
3. How does your language delimit code blocks under each condition in selection control statements?
   1. In Fortran, it delimits code blocks by indentation only when it comes to conditional statements. Also, technically if we add the endif to signalize the end of that entire code block. An example would be;
      1. IF (x == 57) THEN ! Start of the code block
         1. PRINT \*, “The value is 57.” ! Indentation helps delimits the code
      2. ELSE
         1. PRINT \*, “The value is not 57.” ! Indentation helps delimits the code
      3. ENDIF ! End of the code block

(I sort of assumed here, there wasn’t any clear answers online).

1. Does your language use short-circuit evaluation?  Include an example of the short-circuit logic working or not working (or both, if your language is like Java and supports both!)
   1. Yes, Fortran does use short-circuit evaluation but in a complex method. It’s hard to really explain by words alone and right now it’s a bit difficult to fully comprehend being so exhausted at 9PM. So here is an example! Apparently you need to include the value variable in an IF/THEN statement with a nested IF/THEN. I am currently not sure what top/z.get and .ne.0 does at the moment but it makes short circuiting happen!
      1. ! ----- Short circuiting
      2. IF(z.ne.0 ) THEN
      3. IF (top/z.gt.10.0 ) THEN
      4. WRITE(\*,\*) "This works!"
      5. ENDIF
      6. ENDIF ! ----- CODE INSPIRED BY https://fortranwiki.org/fortran/show/shortcircuiting
2. How does your programming language deal with the “dangling else” problem?
   1. Fortran wonderfully handles the ‘dangling else’ problem by something rather simple, naming loops! By naming loops, the program can apparently exit the loop easier than other languages leaving loops unnamed. There is the addition of Fortran’s control structures, which like ‘ENDIF’, help halts things before mayhem starts.
3. If your language supports switch or case statements, do you have to use “break” to get out of them?  Can you use “continue” to have all of the conditions evaluated?
   1. In Fortran, things are a breeze when it comes to switch statements, to which it does not need ‘breaks’ or ‘continue’. Again, Fortran has the super handy control structures that help halting things, which is the case also for switches. You just need to add a END SELECT (name of your switch).

SOURCES:

<https://www.tutorialspoint.com/fortran/select_case_construct.htm>

<https://fortranwiki.org/fortran/show/short-circuiting>

<https://pages.mtu.edu/~shene/COURSES/cs201/NOTES/chap03/else-if.html>

<https://www.tutorialspoint.com/fortran/fortran_decisions.htm>

<https://craftofcoding.wordpress.com/2022/02/11/what-fortran-does-better-than-c-like-languages/>

<https://pages.mtu.edu/~shene/COURSES/cs201/NOTES/chap03/log-type.html>

<https://www.tutorialspoint.com/fortran/if_elseif_else_construct.htm>