Cheat Sheet - Comparators & Logical Operators

This sheet is intended to provide an overview of Fortran comparison and logical operators.

- 00 Logical constants (NOTE THE FULL STOPS!)
  - .TRUE. - Logical constant for true state
  - .FALSE. - Logical constant for false state
- 01 Comparators
  - Comparators can be applied to numeric types (integers, reals etc.) and character variables
    - NOT logical types
  - Can apply them to derived types but you have to write the operators yourself
    - No default comparators for derived types
  - a == b
    - Returns .TRUE. if **a** is the same as **b** otherwise returns .FALSE.
  - a /= b
    - Returns .TRUE. if **a** is different to **b** otherwise returns .FALSE.
  - a > b
    - Returns .TRUE. if **a** is strictly greater than **b** otherwise returns .FALSE.
    - If you apply this to a character variable then it will use Lexicographical order to determine truth (https://en.wikipedia.org/wiki/Lexicographical_order)
  - a >= b
    - Returns .TRUE. if **a** is greater than or equal to **b** otherwise returns .FALSE.
    - If you apply this to a character variable then it will use Lexicographical order to determine truth (https://en.wikipedia.org/wiki/Lexicographical_order)
  - a < b

- Returns .TRUE. if **a** is strictly less than **b** otherwise returns .FALSE.
- If you apply this to a character variable then it will use Lexicographical order to determine truth (https://en.wikipedia.org/wiki/Lexicographical_order)

- a <= b
  - Returns .TRUE. if **a** is less than or equal to **b** otherwise returns .FALSE.
  - If you apply this to a character variable then it will use Lexicographical order to determine truth (https://en.wikipedia.org/wiki/Lexicographical_order)

- 02 Logical operators (NOTE THE FULL STOPS)
  - Logical variables cannot be compared or combined with the comparator operators - they have their own
  - .NOT. a
    - Invert the truth state of **a** .TRUE. <=> .FALSE.
  - a .OR. b
    - Returns .TRUE. if either **a** or **b** is .TRUE. otherwise returns .FALSE.
  - a .AND. b
    - Returns .TRUE. if both **a** and **b** are .TRUE. otherwise returns .FALSE.
  - a .EQV. b
    - Returns .TRUE. if **a** and **b** are both in the same truth state otherwise returns .FALSE.
  - a .NEQV. b
    - Returns .TRUE. if a and b are both in different truth states otherwise returns .FALSE. This is the XOR operator

- 03 FORTRAN 77 Comparator operators (NOTE THE FULL STOPS!)
  - The above comparator operators were introduced in Fortran 90. FORTRAN 77 and earlier had different comparators that are still valid in the Fortran 90 standard and are quite common in the wild
  - They come from the days when many computers had far fewer permissible characters so you couldn't use things like > or <. C and C++ have what are called "Alternative Operator Representations" for the same reason
  - a .EQ. b
    - Returns .TRUE. if **a** is the same as **b** otherwise returns .FALSE.
  - a .NE. b
    - Returns .TRUE. if **a** is different to **b** otherwise returns .FALSE.
  - a .GT. b
    - Returns .TRUE. if **a** is strictly greater than **b** otherwise returns .FALSE.
    - If you apply this to a character variable then it will use Lexicographical order to determine truth (https://en.wikipedia.org/wiki/Lexicographical_order)
  - a .GE. b
    - Returns .TRUE. if **a** is greater than or equal to **b** otherwise returns .FALSE.
    - If you apply this to a character variable then it will use Lexicographical order to determine truth (https://en.wikipedia.org/wiki/Lexicographical_order)
  - a .LT. b
    - Returns .TRUE. if **a** is strictly less than **b** otherwise returns .FALSE.
    - If you apply this to a character variable then it will use Lexicographical order to determine truth (https://en.wikipedia.org/wiki/Lexicographical_order)
  - a .LE. b
    - Returns .TRUE. if **a** is less than or equal to **b** otherwise returns .FALSE.
    - If you apply this to a character variable then it will use Lexicographical order to determine truth (https://en.wikipedia.org/wiki/Lexicographical_order)