comparison operators

(Only some of the operators below are relevant to COMP9414 at University of New South Wales - see green colouring below.)

Prolog has two main classes of comparison operators - arithmetic comparison operators (and similar alphabetic comparison operators) and unification-style operators:

Comparison	Definition	Evaluates?
`X = Y`	succeeds if X and Y unify (match) in the Prolog sense	No
`X \= Y`	succeeds if X and Y do not unify; i.e. if `not (X = Y)`	No
`T1 == T2`	succeeds if terms T1 and T2 are identical; e.g. names of variables have to be the same	No
`T1 \== T2`	succeeds if terms T1 and T2 are not identical	No
`E1 =:= E2`	succeeds if values of expressions E1 and E2 are equal	Yes
`E1 =\= E2`	succeeds if values of expressions E1 and E2 are not equal	Yes
`E1 < E2`	succeeds if numeric value of expression E1 is < numeric value of E2	Yes
`E1 =< E2`	succeeds if numeric value of expression E1 is ≤ numeric value of E2	Yes
`E1 > E2`	succeeds if numeric value of expression E1 is > numeric value of E2	Yes
`E1 >= E2`	succeeds if numeric value of expression E1 is ≥ numeric value of E2	Yes
`T1 @< T2`	succeeds if T1 is alphabetically < T2	No
`T1 @=< T2`	succeeds if T1 is alphabetically ≤ T2	No
`T1 @> T2`	succeeds if T1 is alphabetically > T2	No
`T1 @>= T2`	succeeds if T1 is alphabetically ≥ T2	No

See also `is`. `is` is not a comparison operator, but is frequently confused with = by novice Prolog programmers. Briefly, you use `X is Exp` to *evaluate* an arithmetic expression, like `Y + 2`, that contains an arithmetic operator, like `+`, and bind the resulting value to the variable `X` to the left of the the operator `is`.

As an example of `@<` and its relatives,

```
?- likes(mary, pizza) @< likes(mary, plums).
true.
```

This succeeds because `likes` and `mary` are the same in both terms, and `pizza` alphabetically precedes `plums`.