

Self classify = _ 0 0 Equal

`=y` classifies the items of the nub of `y` (that is, `~.y`) according to equality with the items of `y`, producing a boolean table of shape `#~.y` by `#y`. For example:

```
y=: 3 3 $ 'abcdef'
y ; (~.y) ; (=y)
+-----+
|abc|abc|1 0 1|
|def|def|0 1 0|
|abc|   |   |
+-----+
```

`x=y` is 1 if `x` is equal to `y`, and is otherwise 0.

The comparison is made with a tolerance `t`, normally `2` to the power `_44` but also controlled by the fit conjunction `!.`, as in `x=! .0 y`. Formally, `x=y` is 1 if the magnitude of `x-y` does not exceed `t` times the larger of the magnitudes of `x` and `y`.

Tolerance applies similarly to other verbs as indicated for each, notably to `Match (-:)`, to `Floor (<.)`, and to `Signum (*)`, but not to `Grade (/:)`.

Both the monadic and dyadic cases of the verb `=` apply to nouns of any rank, and to boxed as well as simple nouns. For example:

```
]a=: ;: 'Try and try and try again.'
+-----+
|Try|and|try|and|try|again.|
+-----+

~. A
+-----+
|Try|and|try|again.|
+-----+

=a
1 0 0 0 0 0
0 1 0 1 0 0
0 0 1 0 1 0
0 0 0 0 0 1

a= <'and'
0 1 0 1 0 0
```

Because of the limited precision of the computer, results which should agree (such as `144*(13%144)` and `13`) may not; the tolerant comparison allows such a comparison to show agreement (a result 1). More or less stringent comparisons may be made by using the conjunction `!` to specify a tolerance `t`, as in the function `eq=: =! .t`.