Control Structures - Return

In all the previous examples, the result of an sequence of expressions has been the value of the last expression. We have been taking advantage of a shortcut offered in workspaces: if there is no return statement, they return the result of the the last expression. Elsewhere in the system this isn't so: you need to have a return statement at the end if you want to return the last value. Furthermore, sometimes you want to return a value from the middle of a sequence of expressions.

A return statement causes the result of an expression to become the result of a sequence of expressions, and also stops all further execution of the whole sequence.

```
$ income, payment, monthlies.

income := 28500 / 12.

payment := 872.50.

monthlies := 200 + payment.

if monthlies / income > 0.50


then [ return "Monthly payments are too high." ].

if payment / income > 0.30


then [ return "Payment is too high." ].

return "This mortgage is OK".
```

If either of the blocks gets executed (because one of the conditions is true) then a string will be returned and the rest of the sequence not executed.

The word return can appear before any expression. Since it is not a message, return can appear before a message without the message having to be in parentheses:

```
if (choose 1 to 100) > 50

then [ return "aeiou" choose ]

else [ return "bcdfghjklmnpqrstvwxyz" choose ].
```

Return Without a Value

If you just want to stop execution without any result, you can use return just by itself. The expression will not have a result. In a workspace, the result will show as “No result”.

```
a := choose 1 to 6.

b := choose 1 to 6.

if (a == 1 and b == 1)

then [ return ].

snake eyes, we’re done.

a := choose 1 to 6.

b := choose 1 to 6.

return.
```