

Chaining Methods 2: Order Matters!

You have the following functions defined below (read them *carefully!*):

```
fun is-female(r): r["sex"] == "female" end
fun kilograms(r): r["pounds"] / 2.2 end
fun is-heavy(r): r["kilos"] > 25 end
```

The table `t` below represents four animals from the shelter:

name	sex	age	fixed	pounds
"Toggle"	"female"	3	true	48
"Fritz"	"male"	4	true	92
"Nori"	"female"	6	true	35.3
"Maple"	"female"	3	true	51.6

Match each Pyret expression (left) to the description of what it does (right). Note: one description might match multiple expressions!

`t.order-by("kilos", true)`

1

A Produces a table containing Toggle, Nori and Maple, with an extra column showing their weight in kilograms

`t.filter(is-female)
.build-column("kilos", kilograms)`

2

B Produces a table containing Maple, Nori and Toggle (in that order)

`t.build-column("kilos", kilograms)
.filter(is-heavy)`

3

C Produces a table containing only Fritz, with a single extra column called `kilos`

`t.filter(is-heavy)
.build-column("kilos", kilograms)`

4

D Won't run: will produce an error

`t.build-column("kilos", kilograms)
.filter(is-heavy)
.order-by("sex", true)`

5

E Produces a table containing only Fritz, with two extra columns

`t.build-column("female", is-female)
.build-column("kilos", kilograms)
.filter(is-heavy)`

6

F Produces a table containing Maple and Fritz