Nested Functions Review

Kathi Fisler, CS-111

Why is has-discount nested inside filter-by-discount? – Look at code structure

```
fun filter-by-discount(t :: Table, d :: String) -> Table:
  doc: "filter table to rows with given discount"
  fun has-discount(r :: Row) -> Boolean:
    r["discount"] == d
  end
  filter-by(t, has-discount)
end

student-tickets =
  sum(filter-by-discount(event-data, "student"),
      "tickcount")
```

To filter-by-discount, we need to use filter-by

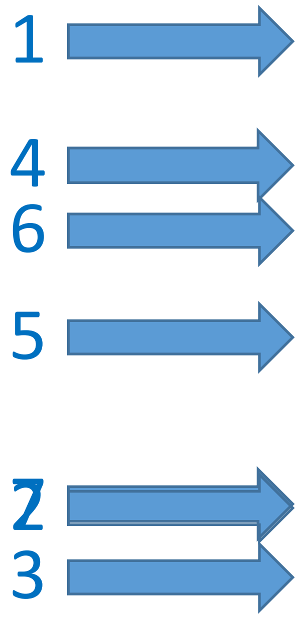
filter-by requires a function that takes a Row and returns a Boolean
(this is just how Pyret works)

has-discount needs to compare the value in the discount column to the value for d originally given to filter-by-discount

d is only visible (only gets substituted) within the body of its enclosing function

thus, `has-discount` is nested inside `filter-by-discount`

How do the pieces tie together? Evaluate this file by hand – what order do steps happen in?



```
fun filter-by-discount(t :: Table, d :: String) -> Table:
  doc: "filter table to rows with given discount"
  fun has-discount(r :: Row) -> Boolean:
    r["discount"] == ~~d~~ “student”
  end
  filter-by(~~t~~, has-discount)
end       event-data

student-tickets =
   sum(filter-by-discount(event-data, "student"),
       "tickcount")
```

1. Pyret remembers that you defined a filter-by-discount function, but doesn’t look inside the body
2. Pyret notices that you want to define student-tickets, but must evaluate the sum(…) expression first
3. Pyret calls filter-by-discount. It substitutes event-data for t and “student” for d in the function body
4. Pyret remembers that you defined has-discount, but doesn’t look inside
5. Pyret evalulates the filter-by call.
6. Internally, Pyret calls has-discount once on each row.
7. Pyret calls sum on the table that resulted from filter-by-discount, then remembers the value of student-tickets