

Common Expressions

The following code draws a traffic light. Propose **one** helper function that you could add to clean up the main overlay expression (there may be more than one candidate -- pick just one). Draw boxes around the expressions that would be replaced by a call to your helper function. Circle the pieces of the boxed expressions that would be inputs to the helper. Provide a meaningful name for your helper function.

```
red = circle(15, "solid", "red")      # don't clean up here -- work on
yellow = circle(15, "solid", "yellow") # the overlay expression instead
green = circle(15, "solid", "green")
```

```
overlay(
  overlay-xy(
    overlay-xy(
      red,
      0, (15 * 2) + (8 * 1),
      yellow),
      0, (15 * 4) + (8 * 2),
      green),
  rectangle(40, 15 * 8, "solid", "gray"))
```

Helper name: compute-y-offset

Conditionals/if-else Expressions

The following program tries to compute the total charges for a phone plan that includes 120 talk minutes, and 10 texts for free, but charges 5 cents per additional minute and 10 cents per additional text message.

Check the three tests in the where: block. For each, indicate whether it will pass or fail. If a test will fail, briefly explain why.

BASE-RATE = 20

```
fun phone-charges(num-minutes :: Number, num-texts :: Number) -> Number:
  doc: "compute phone plan charges based on minutes used and texts sent"
  if num-minutes > 120:
    BASE-RATE + ((num-minutes - 120) * 0.05)
  else if num-texts > 10:
    BASE-RATE + ((num-texts - 10) * 0.10)
  else:
    BASE-RATE
  end
where:
```

```
pass phone-charges(130, 0) is BASE-RATE + 0.50      # 10 extra mins ($0.05 each)
pass phone-charges(0, 20) is BASE-RATE + 1           # 10 extra texts ($0.10 each)
fail  phone-charges(130, 20) is BASE-RATE + 0.50 + 1 # extra mins and texts
end
```

if expressions take answer
to first question that returns
true, so num-texts aren't checked
on these inputs

Errors in Functions/Programs

The following program shows an attempt to write a function to compute the cost of an airline ticket. Each ticket has a base price, plus \$75 to sit in a row with extra legroom, plus fees to check bags. Flyers with gold status don't pay for baggage; flyers with silver status get one bag free. Other bags, including those for passengers without special status, are \$25 each.

The **where:** block shows four tests; the math in the expected answers for each one is correct. For each test, indicate whether it passes or fails. If it fails, explain the problem that Pyret detects (explain the problem, don't fix the code). You may find it helpful to use the line numbers in writing your explanations.

```
1 fun flight-cost(base :: Number, num-bags :: Number, leg-room :: Boolean,
2                 status-level :: String) -> Number:
3   seat-charge =
4     if leg-room: 75
5     else: "no charge"
6     end
7   bag-charge =
8     if status-level == "gold":
9       0
10    else if status-level == "silver":
11      25 * (num-bags - 1)
12    else: 25 * num-bags
13    end
14   base + seat-charge + bag-charge
15 where:
16
17   flight-cost(250, 2, "true", "none") is 375
18     error that "true" isn't a boolean
19
20
21   flight-cost(250, 0, "silver", false) is 250
22     error that "silver" isn't a boolean
23
24
25   flight-cost(250, 2, true, "gold") is 325
26
27     passes
28
29   flight-cost(250, 0, false, "gold") is 250
30
31     + fails on line 14 because seat-charge
32       gets defined as a string on line 5
33 end
```

Table Programming

A classmate has written several functions for processing a table of data about medals won during the olympics (shown below). Each of the questions in this section shows the code that your classmate has written for the problem stated in the comment. For each question, (1) circle or highlight anything in the code that looks like it will throw an error and (2) write a brief summary of the problem. There may be multiple errors or no errors in a question. (Assume the usual **include** statements are in the file.)

```
olympics = table: country, rank, hosting, gold, silver, bronze
  row: "Canada",      3,   false,   11,    8,    10
  row: "Germany",     2,   false,   14,   10,    7
  row: "Netherlands", 5,   false,    8,    6,    6
  row: "Norway",      1,   false,   14,   14,   11
  row: "Republic of Korea", 6,  true,    5,    8,    4
  row: "United States", 4,  false,    8,    8,    6
end
```

Question: sort the table by the total medal count in descending order

```
fun total-medals(r :: Row) -> Number:
```

```
  r["gold + silver + bronze"]
```

```
end
```

not a column name in the table - this is a literal string "gold + silver + bronze"

```
build-column(olympics, "total", total-medals)
```

```
order-by(olympics, "total", false) # false means "sort descending order"
```

Question: produce a table that only includes countries that are hosting

```
fun is-host(r :: Row) -> Boolean:
```

```
  r[hosting]
```

```
end
```

not bound - column names aren't in the directory - they must be referenced as strings like "hosting"

```
host = filter-with(olympics, get-host)
```

function is named is-host

Question: produce a table of countries with more silver than gold medals

```
fun silver-gt-gold(r :: Row) -> Boolean:
```

```
  r["gold" < "silver"]
```

```
end
```

doesn't evaluate to a string as needed for column lookup - can use < to check alphabetical order, so this expression returns a boolean

```
filter-with(olympics, silver-gt-gold)
```

Question: produce a table of the top-three ranked teams

```
fun is-top-three(r :: Row) -> Boolean:
```

```
  rank <= 3
```

```
end
```

needs to be extracted from row r - not a bound name on its own

```
filter-with(olympics, is-top-three)
```


Nothing to add here - answers on page 3

Here are the same programs again, this time showing the error messages that Pyret produces. Now that you see the errors, do you want to change any of your explanations? **Don't edit your original answers.** Just state any changes you want to make on this page (grading will look at both pages)

```
# Question: sort the table by the total medal count in descending order
fun total-medals(r :: Row) -> Number:
  r["gold + silver + bronze"]
end
```

```
build-column(olympics, "total", total-medals)
order-by(olympics, "total", false)
```

ERROR: No such column: gold + silver + bronze

```
# Question: produce a table that only includes countries that are hosting
fun is-host(r :: Row) -> Boolean:
  r[hosting]
end
```

```
host = filter-with(olympics, get-host)
```

ERROR: The identifier hosting is unbound

ERROR: The identifier get-host is unbound

```
# Question: produce a table of countries with more silver than gold medals
fun silver-gt-gold(r :: Row) -> Boolean:
  r["gold" < "silver"]
end
```

```
filter-with(olympics, silver-gt-gold)
```

ERROR: The bracket expression r["gold < "silver"] failed because the second argument evaluated to an unexpected value. An annotation String in <builtin tables> was not satisfied by the value true.

```
# Question: produce a table of the top-three ranked teams
fun is-top-three(r :: Row) -> Boolean:
  rank <= 3
end
```

```
filter-with(olympics, is-top-three)
```

ERROR: The identifier rank is unbound