You're working for a public health organization that has been studying malaria prevention in a rural area. Your team has collected data on which people have been hit with malaria cases, and discovered that people who have mosquito nets and either live 1000 feet from the lake or have had malaria at least 3 times are at low risk for getting sick.

You want to write a function that takes three inputs about a person: the distance (in feet) from the lake to their house, whether they have mosquito nets, and how many times they have had malaria before. The function will return a Boolean indicating whether they are low risk.

```
fun low-risk1(feet-to-lake :: Number, have-net :: Boolean, prev-cases :: Number)
  -> Boolean:
  doc: "determine risk based on distance to water, having nets, and prior cases"
  if have-net:
    if prev-cases >= 3:
      true
    else if feet-to-lake > 1000:
      true
    else:
      false
    end
  else:
    false
  end
where:
  low-risk1(20, true, 0) is false
  low-risk1(20, false, 0) is false
  low-risk1(20, true, 5) is true
  low-risk1(2000, true, 0) is true
  low-risk1(2000, false, 7) is false
end
```

```
fun low-risk2(feet-to-lake :: Number, have-net :: Boolean, prev-cases :: Number)
  -> Boolean:
  doc: "determine risk based on distance to water, having nets, and prior cases"
  if have-net:
    if (prev-cases >= 3) or (feet-to-lake > 1000):
      true
    else:
      false
    end
  else:
    false
  end
end
```

```
fun low-risk3(feet-to-lake :: Number, have-net :: Boolean, prev-cases :: Number)
  -> Boolean:
  doc: "determine risk based on distance to water, having nets, and prior cases"
  have-net and
  ((prev-cases >= 3) or (feet-to-lake > 1000))
end
```

```
# CS-111: The pen-cost functions

fun pen-cost(num-pens :: Number, slogan :: String) -> Number:
  doc: "total cost for pens, each 25 cents plus 2 cents per message character"
  CHAR-COST = 0.02
  ONE-PEN = 0.25

  num-pens * (0.25 + (string-length(slogan) * 0.02))
where:
  pen-cost(3, "wow") is 0.93
  pen-cost(10, "smile") is 10 * (0.25 + (string-length("smile") * 0.02))
end



fun add-shipping(order-amt :: Number) -> Number:
  doc: "increase order price by costs for shipping"
  if order-amt <= 10:
    order-amt + 4
  else if (order-amt > 10) and (order-amt < 30):
    order-amt + 8
  else:
    order-amt + 12
  end
where:
  add-shipping(10) is 10 + 4
  add-shipping(10.01) is 10.01 + 8
  add-shipping(30) is 30 + 12
end


fun total-cost(num-pens :: Number, slogan :: String) -> Number:
  doc: "produce total cost of pen order including shipping"
  add-shipping(pen-cost(num-pens, slogan))
where:
  # these examples use simple pen costs to make sure the combination of functions
  # works as expected (you already tested pen-cost, so you can trust it)
  total-cost(4, "") is 1 + 4
  total-cost(12, "") is 3 + 4
  total-cost(400, "") is 100 + 12
  total-cost(4000, "") is 1000 + 12
end



fun total-cost-discount(num-pens :: Number, slogan :: String) -> Number:
  doc: "produce total cost of pen order including shipping, with bulk discount"
  if num-pens >= 1000:
    add-shipping(pen-cost(num-pens, slogan)) * 0.80
  else:
    add-shipping(pen-cost(num-pens, slogan))
  end
where:
  total-cost-discount(400, "") is (100 + 12)
  total-cost-discount(4000, "") is (1000 + 12) * 0.80
end
```

Problem 2 – designing functions for binge alerts

```
A phone manufacturer needs to compute whether people are binging
too much on apps in a 3-hour period.  They will rate people's usage
based on the number of minutes spent on Facebook or Snapchat, as well
as the number of times someone checked the news. People get one usage
point per minute on social media and 3 points per news check.

Based on the number of usage points, the app will give a severity
rating of serious, moderate, mild, or normal.  The conditions for each
rating are as follows:

* serious when usage is above 120 points
* moderate when usage is above 60 points while at work or school
* mild when usage is below 10 points
* normal in all other cases

Develop a function binge-rating that takes the numbers of Facebook minutes,
Snapchat minutes, and news checks, as well as the person's location (work,
school, vacation, etc) and returns a severity rating.

-------------------------------------------
```

Question: What would be a good set of tests to check the correctness of a solution to this problem?

Question: Based on the problem and the tasks it seems to involve, what (helper) functions might it make sense to write? What would their input and outputs be?