Homework 5
Due Thursday, 2/28/2019

No programming this week, so just turn in a pdf to https://submit.cs.pomona.edu/2019sp/cs131.

1. (20 points) Lazy Evaluation and Parallelism
   Please do problem 4.11 from Mitchell, page 87.
   The function g should be defined as follows (there may be a typo in the book, depending on the printing):

   \[ g(x, y) = \begin{cases} 
   1 & \text{if } x == 0 \\
   2 & \text{if } x + y == 0 \\
   3 & \text{else} 
   \end{cases} \]

2. (10 points) Parse Graph
   Use the parse graph below to calculate the Haskell type for the function. (This is problem 6.5 on page 148 of the revised Chapter 6.)

   \[ \text{fun } f(g, h) = g(h) + 2; \]
Be sure to show your work!

3. (10 points) **Type Inference and Bugs**

What is the type of the following Haskell function:

```
append([], l) = l
append (x: l, m) = append(l, m)
```

Write one or two sentences to explain succinctly and informally why `append` has the type you give. This function is intended to append one list onto another. However, it has a bug. How might knowing the type of this function help the programmer to find the bug?

4. (10 points) **Type Inference and Debugging**

Please do problem 6.8 from Mitchell’s revised Chapter 6, page 149.

NOTE: There is an important typo in the problem. The (incorrect) definition of `reduce` should be:
reduce(f, [x]) = [x]
reduce(f, (x:y)) = f(x, reduce(f, y))

Notice the extra square brackets in the first clause!
In your answer to this problem, explain how to fix the definition.

5. (15 points) **Dynamic Typing in Haskell**
   Please do problem 6.11 from Mitchell’s revised Chapter 6 page 151.
   For part c, assume that `car` returns `nil` when applied to anything other than a `Cons` cell.