Problem Set 1

IAP 2020 18.S097: Programming with Categories

Due Thursday January 16

Good academic practice is expected. In particular, cooperation is encouraged, but assignments must be written up alone, and collaborators and resources consulted must be acknowledged.

We suggest that you attempt all problems, but we do not expect all problems to be solved. We expect some to be easier if you have more experience with mathematics, and others if you have more experience with programming. A guideline is that five complete problems is a good number to write up and submit as homework.

Question 1. *Functions in mathematics and Haskell.*

Suppose $f\colon \texttt{Int} \to \texttt{Int}$ sends an integer to its square, $f(x) \coloneqq x^2$ and that $g\colon \texttt{Int} \to \texttt{Int}$ sends an integer to its successor, $g(x) \coloneqq x + 1$.

(a) Write f and g in Haskell, including their type signature and their implementation.
(b) Let $h \coloneqq f \circ g$. What is $h(2)$?
(c) Let $i \coloneqq f \fatsemi g$. What is $i(2)$?

Question 2. *Two small categories.*

Recall that a *category* consists of the data:

1. a set $\mathrm{Ob}(\mathcal{C})$ of objects;
2. for every pair of objects $c, d \in \mathrm{Ob}(\mathcal{C})$ a set $\mathcal{C}(c, d)$ of morphisms;
3. for every three objects b, c, d and morphisms $f\colon b \to c$ and $g\colon c \to d$, a specified morphism $(f \fatsemi g)\colon b \to d$ called the *composite of f and g*;
4. for every object c, an identity morphism $\mathrm{id}_c \in \mathcal{C}(c, c)$; and

subject to two laws:

Unit: for any $f\colon c \to d$, the equations $\mathrm{id}_c \fatsemi f = f$ and $f \fatsemi \mathrm{id}_c = f$ hold.
Associative: for any $f_1\colon c_1 \to c_2$, $f_2\colon c_2 \to c_3$, and $f_3\colon c_3 \to c_4$, the equation $(f_1 \fatsemi f_2) \fatsemi f_3 = f_1 \fatsemi (f_2 \fatsemi f_3)$ holds.

This tiny category is sometimes called the *walking arrow category* **2**.

$$\mathbf{2} \coloneqq \boxed{\mathrm{id}_1 \circlearrowright \overset{1}{\bullet} \xrightarrow{f} \overset{2}{\bullet} \circlearrowright \mathrm{id}_2}$$

(a) Write down the set of objects, the four sets of morphisms, the composition rule, and the identity morphisms.
(b) Prove that this category obeys the unit and associative laws.

Question 3. *Is it an isomorphism?.*

Suppose that someone tells you that their category $\mathcal{C}$ has two objects c, d and two non-identity morphisms, $f\colon c \to d$ and $g\colon d \to c$, but no other morphisms. Does f have to be the inverse of g, i.e. is it forced by the category axioms that $g \circ f = \mathrm{id}_c$ and $f \circ g = \mathrm{id}_d$?

Question 4. *Almost categories.*

(a) Give an example of some data – objects, morphisms, composition, and identities – that satisfies the associative laws but not the unit law.
(b) Give an example of some data – objects, morphisms, composition, and identities – that satisfies the unit laws but not the associative law.

Question 5. *Monoids.*

A *monoid* $(M, *, e)$ is
1. a set M;
2. a function $*\colon M \times M \to M$; and
3. an element $e \in M$ called the *identity*;

subject to two laws:
Unit: for any $m \in M$, the $e * m = m$ and $m * e = m$ hold.
Associative: for any m_1, m_2, and m_3, the equation $(m_1 \fatsemi m_2) \fatsemi m_3 = m_1 \fatsemi (m_2 \fatsemi m_3)$ holds.

(a) Show that $(\mathbb{N}, +, 0)$ forms a monoid.
(b) A string in 0 and 1 is a (possibly) empty sequence of 0s and 1s; examples include 0, 11, 0110, 0101110 and so on. We write the empty string []. Let $\mathrm{List}_{\{0,1\}}$ be the set of strings in 0 and 1. Given two strings a and b, we may concatenate them to form a new string ab. Show that $\mathrm{List}_{\{0,1\}}$, together with concatenation and the empty string [], form a monoid.
(c) Explain why (prove that) every monoid can be viewed as a category with a single object.

Question 6. *Preorders.*

A *preorder* is a category such that, for every two objects a, b, there is at most one morphism $a \to b$. That is, there either is or is not a morphism from a to b, but there are never two morphisms a to b. If there is a morphism $a \to b$, we write $a \leq b$; if there is not a morphism $a \to b$, we don't.

For example, there is a preorder $\mathcal{P}$ whose objects are the positive integers $\mathrm{Ob}(\mathcal{P}) = \mathbb{N}_{\geq 1}$ and where

$$\mathcal{P}(a, b) \coloneqq \{x \in \mathbb{N} \mid x * a = b\}$$

This is a preorder because either $\mathcal{P}(a, b)$ is empty (if b is not dividible by a) or contains exactly one element.

(a) What is the identity on 12?
(b) Show that if $x\colon a \to b$ and $y\colon b \to c$ are morphisms, then there is a morphism $y \circ x$ to serve as their composite.
(c) Would it have worked just as well to take $\mathcal{P}$ to have all of $\mathbb{N}$ as objects, rather than just the positive integers?

Question 7. *Church Booleans.*

Boolean logic may be encoded in the lambda calculus using the following definitions:

$$\begin{aligned} \text{True} &= \lambda x.(\lambda y.x) \\ \text{False} &= \lambda x.(\lambda y.y) \\ \text{AND} &= \lambda p.(\lambda q.(pq)p) \\ \text{OR} &= \lambda p.(\lambda q.(pp)q) \end{aligned}$$

Evaluate the lambda terms True AND False and False OR True.

Question 8. *The Y combinator.*

The Y combinator is an iconic lambda term whose reduction does not terminate. It is defined as follows:

$$Y = \lambda f.\big((\lambda x.f(xx))(\lambda x.f(xx))\big)$$

Compute Yg.

Question 9. *Defining a toy category in Haskell.*

In Haskell, we may define a typeclass whose instances are the data of a category in the following way:

```
class Category obj mor | mor -> obj where
  dom :: mor -> obj
  cod :: mor -> obj
  idy :: obj -> mor
  cmp :: mor -> mor -> Maybe mor
```

The "Maybe" part is saying that two morphisms may not compose (e.g. $f\colon a \to b$ and $g\colon b' \to c$ only compose if $b = b'$). Note also that there are no laws—associative or unital—so the user of this class has to just certify that these laws really do hold in their specific case. Finally, the weird `| mor -> obj` thing at the top is called a "functional dependency" and is just there to soothe the compiler.

(a) Implement the category $\mathbf{2} = \boxed{\bullet \to \bullet}$ from Question 2 as an instance of this `Category` class.

Question 10. *Grade the pset.*

Give a grade to this problem set, taking into account how much you learned, how interesting or fun it was, and how much time you spent on it. Explain your grade.