Category Theory (ITI9200) – Exercise Sheet 1

4 March 2024

Outline. Complete as many of the exercises below as you are able. Each exercise has a number of tasks. Each task has an assigned number of points in square brackets, e.g. **[1]**. Points may be awarded for answers that demonstrate effort, even if the answer is not entirely correct. There are **20** total points. The exercise sheet is expected to take around 2 – 4 hours.

Submission. Email your work to Fosco Loregian at

`fosco.loregian@taltech.ee`

or hand in your work to one of the lecturers or teaching assistants at the start of end of a lecture. Deadline:

23:59 on 18 March 2024.

Exercise 1

The category $\mathbf{Pred}(\mathbf{Set}_t)$ of **predicates and total functions** is defined as follows.

- An *object* is a pair (X, A) where X is a set, and A is a subset of X;
- A *morphism* from (X, A) to (Y, B) is a total function $f\colon X \to Y$ such that $f(a) \in B$ for every $a \in A$.
  We shall use the same symbol to denote both a morphism $(X, A) \to (Y, B)$ in $\mathbf{Pred}(\mathbf{Set}_t)$ and the total function that defines it.
- The *identity* of an object (X, A) is the identity function $\mathrm{id}_X\colon X \to X$.
- The *composite* of $f\colon (X, A) \to (Y, B)$ and $g\colon (Y, B) \to (Z, C)$ is the composite function $g \circ f\colon X \to Z$.
- Two morphisms $f, f'\colon (X, A) \to (Y, B)$ are *equal* when they are equal as functions.

The category $\mathbf{Pred}(\mathbf{Set}_p)$ of **predicates and partial functions** is defined in the same way as $\mathbf{Pred}(\mathbf{Set}_t)$, except that a morphism from (X, A) to (Y, B) is now a *partial* function $f\colon X \to Y$ such that $f(a) \in B$ for every $a \in A$.

Tasks.

1. Why are $\mathbf{Pred}(\mathbf{Set}_t)$ and $\mathbf{Pred}(\mathbf{Set}_p)$ both valid definitions of categories, i.e. why is composition in each category unital and associative? **[1]**
2. Does $\mathbf{Pred}(\mathbf{Set}_t)$ have an initial object (i.e. at least one)? If so, say what it is. If not, explain why not. **[1]**
3. Does $\mathbf{Pred}(\mathbf{Set}_t)$ have a terminal object (i.e. at least one)? If so, say what it is. If not, explain why not. **[1]**
4. Does $\mathbf{Pred}(\mathbf{Set}_p)$ have an initial object (i.e. at least one)? If so, say what it is. If not, explain why not. **[1]**
5. Does $\mathbf{Pred}(\mathbf{Set}_p)$ have a terminal object (i.e. at least one)? If so, say what it is. If not, explain why not. **[1]**

Exercise 2

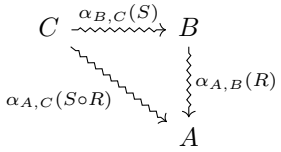
The category of **Rel** of **sets and relations** is defined as follows.

- An *object* is a set.
- A *morphism* from a set A to a set B is a **relation** from A to B, which is a subset of $A \times B$.[1]

  To distinguish relations between sets from functions between sets, we shall write $R\colon A \rightsquigarrow B$ for a morphism $R \in \mathbf{Rel}(A, B)$.
- The *identity* of an object A is the set of all pairs (a, a) where $a \in A$ (this is a subset of $A \times A$).
- The *composite* of morphisms $R\colon A \rightsquigarrow B$ and $S\colon B \rightsquigarrow C$ is the set of all pairs of the form (a, c) with $a \in A$ and $c \in C$ for which there exists an element $b \in B$ such that $(a, b) \in R$ and $(b, c) \in S$ (this is a subset of $A \times C$).
- Two morphisms $R, R'\colon A \rightsquigarrow B$ are *equal* when R and R' are equal as sets, i.e. when they contain exactly the same pairs of elements (a, b).

Tasks.

1. Why is **Rel** a valid definition of a category, i.e. why is composition unital and associative? **[1]**
2. Does **Rel** have an initial object (i.e. at least one)? If so, say what it is. If not, explain why not. **[1]**
3. Does **Rel** have a terminal object (i.e. at least one)? If so, say what it is. If not, explain why not. **[1]**
4. Find a bijective function
   $$\alpha_{A,B}\colon \mathbf{Rel}(A, B) \to \mathbf{Rel}(B, A)$$
   for each set A and set B, such that the following conditions hold.
   (a) For each set A, we have $\alpha_{A,A}(\mathrm{id}_A) = \mathrm{id}_A$ in the category **Rel**.
   (b) For all sets A, B, C and relations $R\colon A \rightsquigarrow B$ and $S\colon B \rightsquigarrow C$, we have $\alpha_{A,C}(S \circ R) = \alpha_{A,B}(R) \circ \alpha_{B,C}(S)$, i.e. the following diagram commutes in the category **Rel**.

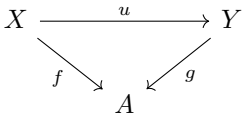


[1]

Exercise 3

Let $\mathcal{C}$ be a category and A an object of $\mathcal{C}$. The **category over** A (denoted $\mathcal{C}/A$) is defined as follows.

- An *object* is a pairs (X, f) where X is an object of $\mathcal{C}$, and $f\colon X \to A$ is a morphism in $\mathcal{C}$.
- A *morphism* from (X, f) to (Y, g) is a morphism $u\colon X \to Y$ in $\mathcal{C}$ such that $g \circ u = f$, i.e. the following triangle commutes.



  We shall use the same symbol to denote both a morphism $(X, f) \to (Y, g)$ in $\mathcal{C}/A$ and the morphism in $\mathcal{C}$ that defines it.
- The *identity* of an object (X, f) is the identity morphism $\mathrm{id}_X\colon X \to X$ in $\mathcal{C}$.
- The *composite* of $u\colon (X, f) \to (Y, g)$ and $v\colon (Y, g) \to (Z, h)$ is the composite morphism $v \circ u\colon X \to Z$ in $\mathcal{C}$.
- Two morphisms $u, u'\colon (X, f) \to (Y, g)$ are *equal* when they are equal as morphisms in $\mathcal{C}$.

[1] Recall that $A \times B$ is the Cartesian product of A and B, i.e. the set of pairs (a, b) with $a \in A$ and $b \in B$.

Tasks.

1. Why is $\mathcal{C}/A$ a valid definition of a category, i.e. why is composition unital and associative? **[1]**
2. The identity morphism $\mathrm{id}_A \colon A \to A$ in $\mathcal{C}$ is a terminal object in $\mathcal{C}/A$. Explain why. **[1]**
3. An object (X, f) in $\mathcal{C}/A$ is terminal if and only if $f \colon X \to A$ is an isomorphism in $\mathcal{C}$. Explain why (i.e. show that such an isomorphism is terminal in $\mathcal{C}/A$, and show that a terminal object in $\mathcal{C}/A$ must be an isomorphism in $\mathcal{C}$). **[1]**
4. Does $\mathcal{C}/A$ have an initial object? (Hint: the answer might depend on $\mathcal{C}$.) **[1]**

Exercise 4

Tasks.

1. Come up with your own example of a category, i.e. define the collection of objects, morphisms, identity morphisms, composites of morphisms, and equality of morphisms, and show that it satisfies unitality and associativity of composition. It can be as simple or as complicated as you like (but should not be an example we have already seen in the lectures/exercises). **[1]**
2. Come up with another example of a category. Again, it can be as simple or as complicated as you like, but should be different from your first example. **[1]**
3. Give an example of a graph that is *not* a category (i.e. there is no possible choice of identities and composition that forms a category), and explain why it is not a category. It can be as simple or as complicated as you like. **[1]**

Exercise 5

In a similar fashion to the category **ParProg** of partial programs, we may define a category $\mathbf{CtxProg}_{\mathtt{G}}$ of *programs with context* `G`. The idea is that programs now can use a read-only context (i.e. value) of type `G` during their computation, which is passed as one of the arguments. For this exercise, we will only consider programs containing `Int`, `String`, and `Bool` as types. Let's pick a **fixed** type $\mathtt{G} \in \{\mathtt{Int}, \mathtt{String}, \mathtt{Bool}\}$ and consider the following graph $\mathbf{CtxProg}_{\mathtt{G}}$.

- A *node* is a type in $\{\mathtt{Int}, \mathtt{String}, \mathtt{Bool}\}$.
- Given two types `A`,`B`, the set $\mathbf{CtxProg}_{\mathtt{G}}(\mathtt{A}, \mathtt{B})$ of *edges* is the set of Crust functions which take **two** arguments, one of type `G` and one of type `A`, and which have `B` as return type.

  (Note: the type `G` is fixed for every edge.)

For instance, for the fixed type $\mathtt{G} := \mathtt{Bool}$, the following function is an edge from `Int` to `Int` in $\mathbf{CtxProg}_{\mathtt{Bool}}$.

```
fn negate_if(flip: Bool, x: Int): Int => if flip then -x else x
```

Tasks.

1. Show that the graph $\mathbf{CtxProg}_{\mathtt{G}}$ may be extended to a category in which equality of morphisms is given by program equivalence (i.e. equality of programs in the category **Prog**), by defining identity morphisms and composition of morphisms, and showing that composition is associative and unital. You may find it helpful to refer back to the definitions of **Prog** and **ParProg** to see how those were shown to be categories.

   (Note: the type `G` can be left as an abstract type. You do not need to prove that $\mathbf{CtxProg}_{\mathtt{Int}}$, $\mathbf{CtxProg}_{\mathtt{String}}$, and $\mathbf{CtxProg}_{\mathtt{Bool}}$ are all categories separately.) **[4]**