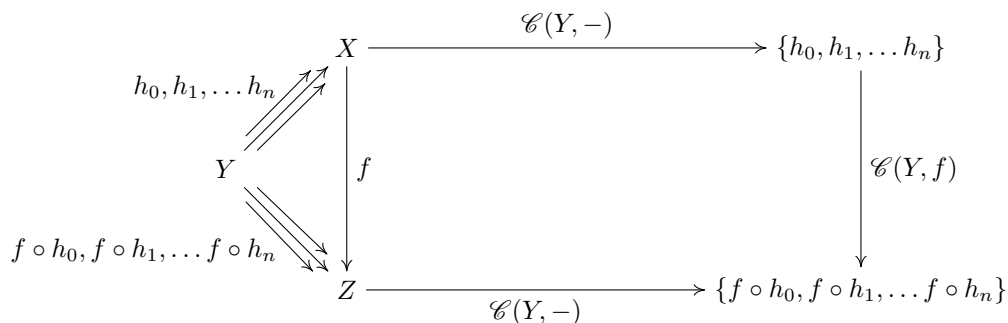


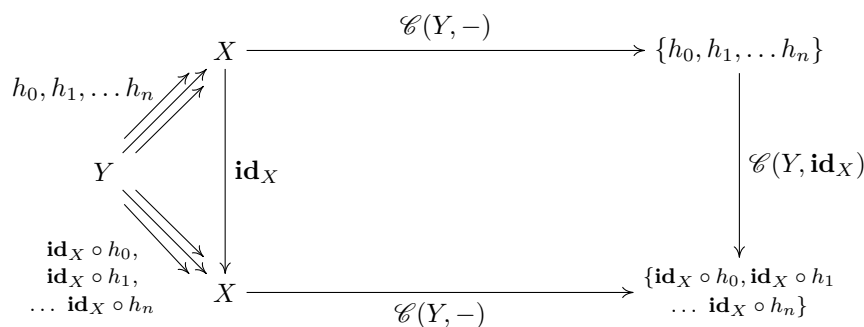
## CTfP Challenges — Chapter 14: Representable Functors

**Question 1.** Show that the hom-functors map identity morphisms in  $\mathcal{C}$  to corresponding identity functions in **Set**

Consider objects  $X, Y, Z$  along with morphisms  $X \xrightarrow{f} Z$  and  $Y \xrightarrow{h_0, h_1, \dots, h_n} X$ . By functoriality, we know that the Hom functor  $\mathcal{C}(Y, -)$  maps the morphism  $f$  to some function from the homset  $\mathcal{C}(Y, X)$  to the homset  $\mathcal{C}(Y, Z)$ . Specifically, this function will send each  $h_n \in \mathcal{C}(Y, X)$  to  $f \circ h_n \in \mathcal{C}(Y, Z)$ . We can see this more clearly with a diagram:



Now suppose we set  $Z = X$  and  $f = \text{id}_X$ :



We now see that for all  $Y \xrightarrow{h} X$  the function  $\mathcal{C}(Y, \text{id}_X)$  maps  $h$  to  $\text{id}_X \circ h$ . By the definition of the identity morphism,  $\text{id}_X \circ h \equiv h$ . Therefore,  $\mathcal{C}(Y, \text{id}_X)$  is precisely the identity function on the homset  $\mathcal{C}(Y, X)$ . A similar argument may be constructed for the contravariant case.

**Question 2.** Show that *Maybe* is not representable.

We can make an argument based on cardinality. Any isomorphism between sets must be bijective, and in order to form a bijection between two sets, they must have the same cardinality. For any type  $T$  with cardinality  $t$ , the type  $\text{Maybe}\langle T \rangle$  will have cardinality  $t + 1$ . Given any candidate representing type  $R$ , the function type  $R \rightarrow T$  will have cardinality  $t^r$ . This means that we must find some  $r$  such that  $t^r = t + 1$ . Rearranging, we find that  $r = \log_t(t + 1)$ . For  $t = 1$ , we find  $r$  is undefined, for  $t = 2$ ,  $r = 1.5849\dots$ —we will struggle to find a type with such cardinality! The function tends toward 1 as  $t$  approaches infinity, so we have no hope of finding an integer-valued result.

**Question 3.** *Is the `Reader` functor representable?*

Yes, since `Reader` maps two types `A` and `B` onto their function type `A → B`, it is trivially representable.

**Question 4.** *Using `Stream` representation, memoize a function that squares its argument.*

(See accompanying `F#` script.)

**Question 5.** *Show that `tabulate` and `index` for `Stream` are indeed the inverse of each other. (Hint: use induction.)*

I don't think there's any way to do this other than the solution written here:  
<http://danshiebler.com/2018-11-10-category-solutions/>

**Question 6.** *The functor: `Pair a = Pair a a` is representable. Can you guess the type that represents it? Implement `tabulate` and `index`.*

We can make another cardinality argument. Given a type `T` with cardinality  $t$ , The type `Pair<T>`, has cardinality  $t \times t$ . This means that the function `R → T` must also have cardinality  $t \times t$ . Which means we must find a type with cardinality  $r$  such that  $t^r = t \times t$ . This is plainly 2, so we can easily choose `Bool` for `R`; however, we may wish to choose something with more descriptive values (such as `fst` and `snd`, `car` and `cdr`, or `left` and `right`), as any type of cardinality 2 will do the trick. Implementation is provided in the accompanying script.