CSCC24 2025 Summer Assignment 4

Due: August 7 11:59PM
This assignment is worth 10% of the course grade.

Q1: Type Inference [10 marks]

The starter file and the file to hand in is a4-infer.txt

Show type inference steps for the following. It is intentionally incomplete; it means that you can stop after finding the generalized polymorphic type of `mmap`.

```
let mmap = \f m -> case m of { Nothing -> Nothing ; Just x -> Just (f x) }
in ...
```

You may omit detailed unification steps, but do show how `unify` calls `unify-intern` for clarity. (Similar to examples in the lecture.)

Q2: Parametricity

The starter file and the file to hand in is a4-parametricity.txt.

2(a) [10 marks]

Prove that $e :: \forall b.(\mathit{Int} \to b \to b) \to b \to b$ satisfies

$$\mathit{foldr}\ \mathit{op}_R\ z_R\ (e\ (:)\ [\,])= e\ \mathit{op}_R\ z_R$$

for all B_R, $z_R :: B_R$, $\mathit{op}_R :: \mathit{Int} \to B_R \to B_R$.

It is simpler to use a function for the relation $\langle b \rangle$.

2(b): Comparing with Java [3 marks]

If `f :: a -> [a]` and a test case shows `f () = [()]`, then we know `f x = [x]` in general. Another point to note is that `f x` cannot use `show x` because the type of `f` is not `Show a => a -> [a]`.

In contrast, some programmers really love the fact that in Java *every type* has a `toString()` method. We will see why it sacrifices parametricity.

Implement in Java

```
<T> LinkedList<T> bad(T x)
```

such that `bad` returns a list of length 1 for some inputs, and the empty list for some other inputs, by exploiting `x.toString()`.