Programming Languages
CSCI-GA.2110.001 Fall 2019

Homework 2
Due Sunday, December 15 at 11:55pm

You should write the answers using word, latex, etc., and upload them as a PDF document. **Important**: You must turn this in by 11:55pm on Sunday, December 15. I will be posting the solutions right afterwards, to aid in your study for the final exam on December 18.

1. (a) As we discussed in class, the expression $(\lambda x.\ (x\ x))\ (\lambda x.\ (x\ x))$ has no normal form. Write another expression that has no normal form. Make sure that your expression is distinct from $(\lambda x.\ (x\ x))\ (\lambda x.\ (x\ x))$, i.e. that it wouldn't be convertible to $(\lambda x.\ (x\ x))\ (\lambda x.\ (x\ x))$. Hint: Think about how you'd write a non-terminating expression in a functional language.

   (b) Write the actual expression in the λ-calculus representing the Y combinator, and show that it satisfies the property Y(f) = f(Y(f))

   (c) Define the terms *normal order evaluation* and *applicative order evaluation*.

   (d) Write an expression containing at least two *redexes*, such that normal order evaluation and applicative order evaluation would choose a different redex to reduce. Justify your answer by showing one step of reduction for each order of evaluation.

   (e) Summarize, in your own words, what the two Church-Rosser theorems state.

2. (a) Write a function in ML that has the following type:
   `('a -> 'b) -> ('a -> 'c) -> 'a list -> ('b * 'c) list`.

   (b) What is the type of the following function?

```
fun f (a,b) x [] = []
 |  f (a,b) x (y::ys) = [a x, b y] @ f (a,b) x ys
```

   (c) Explain how the compiler would infer the type in your previous answer.

3. (a) Define the term *dynamic dispatch* and give an example in Java.

   (b) Given a class `A` and a subclass `B` of `A`, explain why, under the *subset interpretation of subtyping*, `B` denotes a set that is a subset of the set denoted by `A`.

   (c) For a language (such as Scala) that allows subtyping among function types, draw a diagram showing the subtyping relationships among the types, `A->A`, `A->B`, `B->A`, and `B->B`, assuming `B` is a subtype of `A`.

   (d) Suppose that `B` is a subtype of `A`. Write some code in Scala that, if it were allowed by the compiler, would illustrate that having `B->Int` be a subtype of `A->Int` would lead to an unsafe program.

4. In Java generics, subtyping on instances of generic classes is invariant. That is, two different instances `C<A>` and `C<B>` of a generic class `C` have no subtyping relationship, regardless of a subtyping relationship between `A` and `B` (unless, of course, A and B are the same class).

(a) Write a function (method) in Java that illustrates why, even if `B` is a subtype of `A`, `C<B>` should not be a subtype of `C<A>`. That is, write some Java code that, if the compiler allowed such covariant subtyping among instances of a generic class, would result in a run-time type error.

(b) Modify the code you wrote for the above question that illustrates how Java allows a form of polymorphism among instances of generic classes, without allowing subtyping. That is, make the function you wrote above be able to be called with many different instances of a generic class.

5. (a) What does the term *covariant subtyping* mean in the context of Scala generics? Just define the term, no code needed.

(b) What does the term *contravariant subtyping* mean in the context of Scala generics?

(c) In ML, a polymorphic list type can be defined using ML's datatype facility:

```
datatype 'a myList = nil | cons of 'a * 'a myList
```

i. Using Scala's case class facility, define a generic type `myList` that is the equivalent of the above `myList` type in ML. It should not use Scala's built-in `List` class.

ii. Write a polymorphic `map()` function in Scala, which takes a function `f` and a `myList` `L`, and returns a `myList` resulting from applying `f` to every element of `L` (that is, exactly what `map()` in ML does). Your `map()` should not be a method of the `MyList` class and it should be just as polymorphic as `map()` in ML.

6. (a) What is the advantage of a reference counting collector over a mark and sweep collector?

(b) What is the advantage of a copying garbage collector over a mark and sweep garbage collector?

(c) Write a brief description of generational copying garbage collection.

(d) Write, in the language of your choice, the procedure `delete(x)` in a reference counting GC system, where `x` is a pointer to a structure (e.g. object, struct, etc.) and `delete(x)` reclaims the structure that `x` points to. Assume that there is a free list of available blocks and `addToFreeList(x)` puts the structure that `x` points to onto the free list.