CS558 Programming Languages – Fall 2023 – Suggested Study Question Solutions for Lecture 6a

1.

```
def h(a:Int,b:Int,c:Int) = a+b+c
def g(a:Int,b:Int) = h(a,b,1) + h(a,b,2)
def f(a:Int) = g(a,a+10)
```

2.

```
def sortup (xs:List[Int])  = insSort((x:Int,y:Int) => x <= y) (xs)
def sortdown (xs:List[Int])  = insSort((x:Int,y:Int) => x >= y) (xs)
```

or, more simply:

```
val sortup =  insSort((x:Int,y:Int) => x <= y) _
val sortdown = insSort((x:Int,y:Int) => x >= y) _
```

3. (a) `def above(n:Int) = map ((x:Int) => (x,x>n))`

(b) `def sumeach(xs:List[List[Int]]) = map (sum) (xs)`

or just

`val sumeach = map (sum)`

4. (a) `def concat(xs:List[String]) : String =  (xs :\ "") ((a,b) => a + b)`

Or, more compactly:

`def concat(xs:List[String]) : String =  (xs :\ "") (_+_)`

(b) `def max(xs:List[Int]) : Int = (xs :\ xs.head) ((x,a) => if (x > a) x else a)`

Or, using a handy library function:

`def max(xs:List[Int]) : Int = (xs :\ xs.head) (Math.max _)`

(c)

```
def unzip[A,B](xys:List[(A,B)]) : (List[A],List[B]) =
       (xys :\ ((Nil:List[A],Nil:List[B]))) ((xy,xys) => (xy._1::xys._1,xy._2::xys._2))
```

Or, more compactly:

```
def unzip[A,B](xys:List[(A,B)]) : (List[A],List[B]) =
        (xys :\ ((Nil:List[A],Nil:List[B]))) {case ((x,y),(xs,ys)) => (x::xs,y::ys) }
```