15-150 Assigment 6

Jonathan Li
jlli
Section S
June 3, 2016

Task 2.1

```
foldl (fn (x,y) => (x=3) orelse y) false ["Three", "Four"]
```

The above expression is not well-typed, as the infix operator `=` must take in two arguments of the same type, and while `3 : int`, `'Three' : string`.

Task 2.2

```
foldl (fn (x,y) => x ^ y) "" ["Hello", "Hola"]
```

The above expression is well-typed. It evaluates to `'HolaHello' : string`, concatenating each of the elements in the list from right to left.

Task 2.3

```
map (fn x => case x of 42 => [41,x] |_=> [43,x]) [[42],[43]]
```

The above expression is not well-typed, as the function provided to the first argument of `map` is of the type `int -> int list`, while each of the elements in the list provided as the argument to the function are of type `int list`.

Task 2.4

```
map (fn L => foldl (fn (x,y) => x+y) 0 L)
```

The above expression is well-typed. It is of type `int list list -> int list`, taking in a list of `int lists`, and returning a list of the sum of each sub-list.