CS320 Programming Languages
Homework #2: MUWAE

Due: 20 March 2019

The goal of Homework #2 is to implement the interpreter of `MUWAE`, which is the extended version of `WAE` that deals with multiple values.

1 Supporting Multi-Value

The first thing we should fix is that we're still define a `Num` with a single numeric value and use it as the result of `run`.

One way to resolve this would be to add a new `case class` called `Nums` to our AST definition. But this would require reworking new cases for it in a few places. So instead, we will choose an easier solution: just change the existing `Num` so instead of holding a single integer it will hold a `List[Int]`.

We need to fix the arithmetic operators. This is a bit tricky, since each of them receives two inputs that are both lists, and they should apply the operator on each pair from these two inputs, and collect a list of all of the results. So to make it easy, here is a skeleton of a utility function that will do this work. It is near-complete, and you have a tiny hole to fill:

```
// applies a binary numeric function on all combinations of integers from
// the two input lists, and return the list of all of the results
def binOp(
  op: (Int, Int) => Int,
  ls: List[Int],
  rs: List[Int]
): List[Int] = ls match {
  case Nil => Nil
  case l :: rest =>
    def f(r: Int): Int = ??? // TODO: complete this function
    rs.map(f) ++ binOp(op, rest, rs)
}
```

Don't forget to add tests that demonstrate that this works: that using with to bind a name to a multi-valued expression works as expected. Here are some tests that should work once you're done with this part:

```
test(run("{+ 3 7}"), List(10))
test(run("{- 10 {3 5}}"), List(7, 5))
test(run("{with {x {+ 5 5}} {+ x x}}"), List(20))
```

2 Adding More Arithmetic Operators

Next, add two arithmetic operators, `min` and `max` that take three expressions and return the minimum and the maximum list of integers, respectively, to `MUWAE`:

```
test(run("{min 3 4 5}"), List(3))
test(run("{max {+ 1 2} 4 5}"), List(5))
test(run("{min {1 4} {2 9} 3}"), List(1, 1, 2, 3))
```

```
test(run("{max {1 6} {2 5} {3 4}}"), List(3, 4, 5, 5, 6, 6, 6, 6))
```