CS4400 *Alex Clemmer, u0458675*

Lecture 4 Problem Set

Problem 2.86

1. Give a correct implementation using only left and right shifts along with 1 subtraction.

```
1  2  3
4  5  6
7  8  9
```

Problem 2.88

1. `(float) x == (float) dx`

   True. Rounding may cause problems in the general case, but `float` is well-defined for the range this deals with. (I verified this experimentally.)

2. `dx - dy == (double) (x - y)`

   True. `double` is well-defined over the range of `int`, so there is no chance of a rounding error. We're dealing in integers, so decimal rounding errors are out. There is also no way to cause ambiguous 0 comparisons.

3. `(dx + dy) + dz == dx + (dy + dz)`

   True. Not true in the general case, as floating-point arithmetic is not associative or distributive, but `double` encapsulates all numbers possible in `int`, and therefore is well defined for these conditions.

4. `(dx * dy) * dz == dx * (dy * dz)`

   False. Distributed

5. `dx / dx == dz / dz`

   False. If `dx` or `dz` is `0` while the other isn't, then they will have different results.