# Abstract Interpretation: Exercises for day 4

[Note: you are welcome to solve these exercises in groups of 2]

February 5, 2015

1. Develop the `=0`, `<>0`, `+1` and `-1` operations for the interval domain
2. Implement the interval domain and the above operations as a module with (roughly) the following signature:

```
type interval

val bot     : interval
val leq     : interval -> interval -> bool
val join    : interval -> interval -> interval

val iszero  : interval -> interval
val notzero : interval -> interval
val incr    : interval -> interval
val decr    : interval -> interval
...
```

3. To plug this module into your implementation from yesterday we need a widening operator to ensure termination.
   - How can we design a widening operator over $\mathit{Interval} \times \mathit{Interval} \times \mathit{Interval}$ ( $\langle I_x, I_y, I_z\rangle \triangledown \langle I'_x, I'_y, I'_z\rangle$ ) in terms of the simpler widening operator over $\mathit{Interval}$ ( $I \triangledown I'$ )?
   - How can we design a widening operator over $PC \to \mathit{Interval} \times \mathit{Interval} \times \mathit{Interval}$ ( $f \triangledown f'$ ) in terms of the widening operator over $\mathit{Interval} \times \mathit{Interval} \times \mathit{Interval}$ ( $\langle I_x, I_y, I_z\rangle \triangledown \langle I'_x, I'_y, I'_z\rangle$ )?
4. Implement the above widening operators (each in its appropriate module) over all three domains and adjust the implementation's iterative loop to incorporate widening.

## Bonus exercises

- Do you need full intervals for analysing the 3 counter machine?
- Extend your analysis further with a subsequent narrowing stage, analogous to the above widening design.