## Exercise 7.1 Interval Analysis: Overflows and Soundness

The interval analysis as we have seen it in the lecture is only defined for programs that use mathematical integers ($\mathbb{Z}$) rather than fixed-width machine integers ($\mathbb{B}^n$ for e.g. $n = 32$). The abstract transformers for this analysis are obviously unsound if the analyzed programming language specifies integers to wrap on overflow (e.g. Java's integers and C's unsigned integers).

Elaborate on whether the defined abstract transformers are sound for C's signed integer arithmetic (where overflows result in undefined behavior), i.e. argue whether compiler transformations based on information computed with these transformers would preserve the semantics of the program. What if we want to use the analysis to verify properties of a program rather than to justify compiler transformations?

## Exercise 7.2 Interval Analysis: Division

In the lecture, we defined the abstract division operation $[l_1, u_1] /^{\#} [l_2, u_2] = [a, b]$ for our interval analysis as follows:

$$\begin{aligned} a &= l_1/l_2 \sqcap l_1/u_2 \sqcap u_1/l_2 \sqcap u_1/u_2 \\ b &= l_1/l_2 \sqcup l_1/u_2 \sqcup u_1/l_2 \sqcup u_1/u_2 \end{aligned} \qquad \text{if } 0 \notin [l_2, u_2]$$

and

$$[a, b] = [-\infty, +\infty] \qquad \text{if } 0 \in [l_2, u_2]$$

This can be imprecise for cases where the divisor interval contains 0 if we decide to conclude from the occurence of a division $a/b$ that $b$ cannot have the value 0 in any valid execution.

Improve the abstract interval division such that it is also precise for this case.

## Exercise 7.3 Interval Analysis: Widening

The lecture slides give a simple widening operator for the interval analysis (slide 205 of the corresponding slide set). In this exercise, you will improve on this:

1. Define a widening operator $\nabla : \mathbb{I} \times \mathbb{I} \to \mathbb{I}$ for intervals that leads to more precise results than the one from the slides.
2. Argue that it fulfills the requirements of a widening operator:
   - It computes an upper bound: $a \nabla b \sqsupseteq a$ and $a \nabla b \sqsupseteq b$.
   - Any widening sequence $(G^i(\bot))_{i \in \mathbb{N}}$ with $G(x_1, \ldots, x_n) = (y_1, \ldots, y_n)$ where $y_i = x_i \nabla f_i(x_1, \ldots, x_n)$ can only increase a finite number of times.
3. Provide an example program and an application of the interval analysis with your widening operator for this program that shows that it computes more precise results than the widening operator from the slides.

## Exercise 7.4 Global Value Numbering

Reconsider Kildall's Algorithm and the AWZ Algorithm for global value numbering as introduced in the lecture. They describe two different approaches to detect equivalence of program expressions.

1. Give an example for a loop-free program for which Kildall's approach can detect more equivalences than the AWZ algorithm can.
2. Perform both analyses on your example program to show that Kildall's approach really results in more detected equivalences. Make the intermediate steps of your fixed point iteration explicit for both analyses.
3. Describe at which point of the AWZ Algorithm the imprecision arises and explain why this happens.