Problem 1. Consider the following Datalog program $P$ (given in Logicblox syntax):

\begin{verbatim}
oneway(x, y) -> int(x), int(y).
path(x, y) -> int(x), int(y).
edge(x, y) -> int(x), int(y).

oneway(x, y) <- path(x, y), !path(y, x).
path(x, y) <- edge(x, y).
path(x, y) <- edge(x, z), path(z, y).
\end{verbatim}

In the program, the predicate `edge` is input and the predicates `oneway` and `path` are derived. The set of possible constants that may appear in the predicates is fixed to $C = \{1, 2, 3\}$.

In this exercise, you will use the Z3 SMT solver to synthesize an input for the Datalog program above such that following queries hold:

\begin{verbatim}
oneway(1, 2)
!edge(1, 2)
!path(2, 1)
\end{verbatim}

**Task 1: Encode the Datalog program into SMT** Use the SMT encoding procedure given on slide 37 to generate SMT constraints for the Datalog program given above. For unrolling the rules, use a bound of $n = 2$.

The result would be a constraint $\varphi_P$. 
**Task 2: Find a model**  Use the Z3 SMT solver to find a model of the constraint

\[ \varphi_P \land \text{oneway}(1, 2) \land \neg\text{edge}(1, 2) \land \neg\text{path}(2, 1) \]

List all oneway, path, and edge predicates that evaluate to true in your model.

**Task 3: Derive and check correctness of the input**  What is the input that you derive from your model?

Check with [https://repl.logicblox.com/](https://repl.logicblox.com/) whether the queries hold for the given program and your synthesized input.