This report shows how to solve the famous riddle from Singapore with epistemic model checking. We use DEMO_S5 from [http://homepages.cwi.nl/~jve/software/demo_s5](http://homepages.cwi.nl/~jve/software/demo_s5) and a modified version of KRIPKEVIS from [https://w4eg.de/malvin/illc/kripkevis](https://w4eg.de/malvin/illc/kripkevis).

We first define the set of all possibilities:

\[
\text{allpos} :: [(\text{Int, String})]
\]

\[
\text{allpos} = [(15, "May"), (16, "May"), (19, "May"), (17, "June"), (18, "June"),
\]

\[
(14, "July"), (16, "July"), (14, "August"), (15, "August"), (17, "August")]
\]

This forms the set of worlds in our initial model. Moreover, also the set of actual worlds is the full set, hence `allpos` occurs twice in the definition below. The two elements of `rels` define the epistemic relations of Albert and Bernard. Instead of listing explicitly which possibilities they can distinguish we use haskell functions to say that they confuse the same day and the same month, respectively.

This is the initial model with all possibilities:

![initCheryl](image)

Figure 1: `initCheryl`

The formula saying that `i` knows Cheryl’s birthday is defined as the disjunction over all statements of the form “Agent `i` knows that the birthday is `s`”:

\[
\text{knWhich} :: \text{Agent} \to \text{Form (Int, [Char])}
\]

\[
\text{knWhich} \ i = \text{Disj} [ \text{Kn} \ i \ (\text{Info} \ s) \mid s \leftarrow \text{allpos} ]
\]
Now we update the model three times, using the function `upd_pa` for public announcements. First with Albert: I don’t know when Cheryl’s birthday is and I know that Bernard does not know.

\[
\text{model2} = \text{upd}_{pa}\text{ InitCheryl}\left(\text{Conj}\left[\text{Ng}\;\text{knWhich}\;a,\;\text{Kn}\;a\;\text{Ng}\;\text{knWhich}\;b\right]\right)
\]

The second announcement by Bernard: “Now I know when Cheryl’s birthday is.”

\[
\text{model3} = \text{upd}_{pa}\;\text{model2}\;\text{ knWhich}\;b
\]

Finally, Albert says: “Now I also know when Cheryl’s birthday is.”

\[
\text{model4} = \text{upd}_{pa}\;\text{model3}\;\text{ knWhich}\;a
\]

Lastly, this helper function uses `texModel` from `KRIPKEVIS` to generate the drawings:

```haskell
myTexModel :: EpistM (Int, String) -> String -> IO String
myTexModel (Mo states _ _ rels pointed) fn = texModel showState showAg showVal "" (VisModel states rels [(s,0) |s<- states] pointed) fn
where
  showState (n,string) = (show n) ++ string
  showVal _ = ""
  showAg i = if i==a then "Albert" else "Bernard"
```