```
module test

abstract sig Person {spouse: lone Person, parents: set Person}
sig Man, Woman extends Person {}
one sig Eve extends Woman {}
one sig Adam extends Man {}

pred Parenthood {
    -- no person is their own ancestor
    no p: Person | p in p.^parents

    -- every person except Adam and Eve has a mother and father
    all p: Person - (Adam+Eve)| one mother: Woman | one father: Man |
        p.parents = mother + father
    }

fact enforceParentHood {Parenthood}

pred SocialNorms[sp: Person -> lone Person] {
    -- spouse is symmetric
    sp = ~sp
    -- a man's spouse is a woman and vice versa
    Man.sp in Woman && Woman.sp in Man
    -- can't marry a sibling unless person is child of Adam
    no p: Person | some p.sp.parents & p.parents and not Adam in p.parents
    -- can't marry a parent
    -- no p: Person | some p.sp & p.parents
    -- parents are married
    all p: Person | p.parents.sp = p.parents
    }

fact enforceSocialNorms {SocialNorms[spouse]}

pred Show {}

pred getMarried  [sp1: Person -> lone Person, p1, p2: Person] {
    (no p1.spouse and no p2.spouse
     and ((p1 in Woman and p2 in Man) or (p1 in Man and p2 in Woman))
     and no p1.parents & p2.parents)

     implies sp1 = spouse + p1->p2

     else sp1 = spouse
  }
```