```
theory mutilated
imports Main

begin

definition chessboard :: (nat × nat) set where
  chessboard = {..7} × {..7}

definition mchessboard :: (nat × nat) set where
  mchessboard = chessboard − {(0, 0), (7, 7)}

datatype domino =
  horizontal nat × nat
| vertical   nat × nat

fun covers :: domino ⇒ (nat × nat) set where
  covers (horizontal (i, j)) = {(i, j), (i+1, j)}
| covers (vertical (i, j)) = {(i, j), (i, j+1)}

definition white :: (nat × nat) set where
  white = {p. even (fst p + snd p)}

definition flip :: (nat × nat) ⇒ (nat × nat) where
  flip p = (7 − fst p, snd p)

theorem mutilated-chessboard:
  fixes s :: domino set
  assumes finite s and ∀ x ∈ s. ∀ y ∈ s. x ≠ y ⟶ covers x ∩ covers y = {}
  shows (⋃ x ∈ s. covers x) ≠ mchessboard
proof
  have h1: finite chessboard by (auto simp add: chessboard-def)
  have bij-betw flip (chessboard ∩ white) (chessboard ∩ −white)
  proof −
    have aux1: ∀ x. x ∈ chessboard ⟶ flip x ∈ chessboard
      by (auto simp add: chessboard-def flip-def)
    have aux2: ∀ x ∈ chessboard. flip x ∈ white ⟷ x ∈ −white
      by (auto simp add: chessboard-def flip-def white-def)
    have aux3: inj-on flip (chessboard ∩ white)
      by (auto simp add: inj-on-def flip-def chessboard-def white-def)
    have aux4: ∀ x ∈ chessboard. flip (flip x) = x
      by (auto simp add: flip-def chessboard-def)
    have aux5: ∀ x ∈ chessboard ∩ −white. x ∈ flip ‘ (chessboard ∩ white)
      by (metis IntI Int-iff aux1 aux2 aux4 imageI)
    show ?thesis
      by (auto simp add: bij-betw-def aux1 aux2 aux3 aux4 aux5)
  qed
  hence h2: card (chessboard ∩ white) = card (chessboard ∩ −white)
    using bij-betw-same-card by blast
  have card (mchessboard ∩ white) < card (mchessboard ∩ −white)
```

```
  proof -
    have (7, 7) ∈ chessboard ∩ white
      by (simp add: chessboard-def white-def)
    moreover have mchessboard ∩ white = chessboard ∩ white - {(0, 0), (7, 7)}
      by (auto simp add: mchessboard-def white-def)
    moreover have mchessboard ∩ -white = chessboard ∩ -white
      by (auto simp add: mchessboard-def white-def)
    ultimately show ?thesis
       by (metis Diff-insert Diff-insert0 card-Diff1-less card-Diff2-less finite-Int h1
h2)
  qed
  moreover have card ((⋃ x ∈ s. covers x) ∩ white) = card ((⋃ x ∈ s. covers
x) ∩ -white)
  proof -
    have aux6: ∀ x. finite (covers x)
    proof fix x show finite (covers x) by (cases x, auto) qed
    have ∀ x. card (covers x ∩ white) = card (covers x ∩ -white)
    proof
      fix x
      show card (covers x ∩ white) = card (covers x ∩ -white)
        by (induction x, auto simp add: Int-insert-left white-def)
    qed
    with assms aux6 show ?thesis
      by (subst UN-simps(4) [symmetric], subst card-UN-disjoint, auto)+
  qed
  moreover assume (⋃ x ∈ s. covers x) = mchessboard
  ultimately show False by auto
qed

end
```