List Properties

```
module Lists where

-- If we do not have two equal secrets in the same list, then this is a proposition
data _∈_ {A : 𝓤 ̇} (x : A) : (ls : List A) → 𝓤  ̇ where
 here : ∀{y ls} → x ＝ y → x ∈ (y ∷ ls)
 there : ∀{y ls} → (ind : x ∈ ls) → x ∈ (y ∷ ls)

_⊃_ : {A : 𝓤 ̇} (xs ys : List A) → 𝓤 ̇
xs ⊃ [] = 𝟙
xs ⊃ (y ∷ ys) = y ∈ xs × (xs ⊃ ys)

⊃-is-prop : {A : 𝓤 ̇} → ∀ xs ys → ((x : A) → is-prop (x ∈ xs)) → is-prop (xs ⊃ ys)
⊃-is-prop xs [] _ = 𝟙-is-prop
⊃-is-prop xs (y ∷ ys) xs-is-prop = Σ-is-prop (xs-is-prop y) λ _ → ⊃-is-prop xs ys xs-
is-prop

s⟨_⟩ : {A : 𝓤 ̇} → (bs-secr scrs : List A) → 𝓤 ̇
s⟨ bs-secr ⟩ scrs = scrs ⊃ bs-secr × bs-secr ⊃ scrs


s⟨⟩-is-prop : {A : 𝓤 ̇} → ∀ ascrs scrs → ((x : A) → is-prop (x ∈ ascrs))
              → ((x : A) → is-prop (x ∈ scrs)) → is-prop (scrs ⊃ ascrs × ascrs ⊃
scrs)
s⟨⟩-is-prop ascrs scrs  d e = Σ-is-prop (⊃-is-prop _ _ e) (λ _ → ⊃-is-prop _ _ d)

∈→∈ : {A : 𝓤 ̇} → ∀ x → (as bs : List A) → (c : bs ⊃ as)
    → x ∈ as → x ∈ bs
∈→∈ x as bs (c , cs) (here refl) = c
∈→∈ x (_ ∷ as) bs (c , cs) (there ins) = ∈→∈ x as bs cs ins


module list-decidable {A : 𝓤 } (dec : (a b : A) → is-decidable (a ＝ b)) where

 remove : A → List A → List A
 remove x [] = []
 remove x (y ∷ ls) = case (dec x y) of λ { (inl _) → ls
                                          ; (inr _) → y ∷ remove x ls}
 _∈?_ : (x : A) → (ls : List A) → is-decidable (x ∈ ls)
 x ∈? [] = inr λ ()
 x ∈? (x₁ ∷ ls) = case (dec x x₁) of
   λ { (inl eq) → inl (here eq)
     ; (inr ¬eq) → case (x ∈? ls) of
         λ { (inl x) → inl (there x)
```

```
; (inr ¬eq2) → inr λ { (here x) → ¬eq x
                     ; (there w) → ¬eq2 w}}
```