append_commutes_under_sublist THM

```
⊢  Discrete{T} ⇒ (∀eq:{T≡₂}, L,M:T List. (L @ M)(⊆eq)(M @ L))
| by UnivCD
|
1. T : Type
2. Discrete{T}
3. eq : {T≡₂}
4. L : T List
5. M : T List
⊢  (L @ M)(⊆eq)(M @ L)  by UnfoldTopAb 0
|
⊢  ∀x∈(L @ M).x(∈eq) (M @ L)
| by FwdThru
|    Thm* Discrete{T} ⇒ (∃f:{T=₂}. True)
|    {discrete_implies_discrete_equality}
|    [2]
|    THEN
|    Analyze -1
|    THEN
|    Thin -1
|    THEN
|    RenameVar ‘EQ’ -1
|    THEN
|    DiscreteEq -1
|    THEN
|    Equivalence 3
|
4. eq ∈ T→T→𝔹
5. ∀x:T. eq(x,x)
6. ∀x,y:T. eq(x,y) ⟺ eq(y,x)
7. ∀x,y,z:T. eq(x,y) ⇒ eq(y,z) ⇒ eq(x,z)
8. L : T List
9. M : T List
10. EQ : {T=₂}
11. EQ ∈ T→T→𝔹
12. ∀x,y:T. EQ(x,y) ⟺ x = y
⊢  ∀x∈(L @ M).x(∈eq) (M @ L)
| by Using [‘eq’,EQ]
|    (BackThru
|     Thm* ∀eq:{T=₂}, P:(T→Prop), L:T List.
|          ∀x∈L.P(x) ⟺ (∀z:T. z(∈eq) L ⇒ P(z))
|     {list_all_is_member_lemma})
|
⊢  ∀z:T. z(∈EQ) (L @ M) ⇒ z(∈eq) (M @ L)  by UnivCD
|
13. z : T
14. z(∈EQ) (L @ M)
⊢  z(∈eq) (M @ L)
| by Using [‘eq’,EQ]
|    (BackThru
|     Thm* ∀eq:{T=₂}, u:T, L:T List. u(∈eq) L ⇒ (∀f:{T≡₂}. u(∈f) L)
|     {is_member_equalities_lemma})
|
⊢  z(∈EQ) (M @ L)
```

```
| by Inst
|    Thm* ∀T:Type, EQ:{T=₂}. EQ ∈ {T≡₂}
|    {discrete_equality_is_equivalence}
|    [T;EQ]
|
15. EQ ∈ {T≡₂}
⊢  z(∈EQ) (M @ L)
| by BackThru
|    Thm* ∀eq:{T≡₂}, L,M:T List, x:T. x(∈eq) (L @ M) ⟺ x(∈eq) (M @ L)
|    {append_commutes_under_is_member}
|
⊢  z(∈EQ) (L @ M)  by Trivial
END
```