```
is_member_append_lemma1  THM

⊢  ∀eq:{T≡₂}, L:T List, x:T.
|  x(∈eq) L ⇒ (∃M,N:T List, y:T. eq(x,y) & L = (M @ (y.N)))
| by RepeatFor 2 (Analyze 0) THEN Equivalence -1 THEN UnivCD
|
1. T : Type
2. eq : {T≡₂}
3. eq ∈ T→T→𝔹
4. ∀x:T. eq(x,x)
5. ∀x,y:T. eq(x,y) ⟺ eq(y,x)
6. ∀x,y,z:T. eq(x,y) ⇒ eq(y,z) ⇒ eq(x,z)
7. L : T List
8. x : T
9. x(∈eq) L
⊢  ∃M,N:T List, y:T. eq(x,y) & L = (M @ (y.N))
| by ListInd -3 THEN Reduce 0
|
|\
| 8. x : T
| ⊢  False ⇒ (∃M,N:T List, y:T. eq(x,y) & nil = (M @ (y.N)))
|   by Analyze 0
|
.\
. 9. u : T
. 10. v : T List
. 11. x(∈eq) v ⇒ (∃M,N:T List, y:T. eq(x,y) & v = (M @ (y.N)))
. ⊢  if eq(x,u)→ true₂ else x(∈eq) v fi
. |  ⇒
. |  (∃M,N:T List, y:T. eq(x,y) & u.v = (M @ (y.N)))
. | by IfThenElseCases 0
. |
. |\
. | .....truecase .....
. | 12. eq(x,u) = true₂
. | ⊢  true₂ ⇒ (∃M,N:T List, y:T. eq(x,y) & u.v = (M @ (y.N)))
. | | by Analyze 0 THEN Thin -1
. | |
. | ⊢  ∃M,N:T List, y:T. eq(x,y) & u.v = (M @ (y.N))
. | | by Witness nil THEN Witness v
. | |
. | ⊢  ∃y:T. eq(x,y) & u.v = (nil @ (y.v))  by Witness u
. | |
. | ⊢  eq(x,u) & u.v = (nil @ (u.v))  by Reduce 0
. | |
. | ⊢  eq(x,u)
. |   by BackThru Thm* ∀P:𝔹. P ⟺ P = true₂    {assert_iff_btrue}
. |
. .\
. . .....falsecase .....
. . 12. eq(x,u) = false₂
. . ⊢  x(∈eq) v ⇒ (∃M,N:T List, y:T. eq(x,y) & u.v = (M @ (y.N)))
. . | by Analyze 0
. . |
. . 13. x(∈eq) v
```

```
. . ⊢  ∃M,N:T List, y:T. eq(x,y) & u.v = (M @ (y.N))  by Analyze -3
. . |
. . 11. eq(x,u) = false₂
. . 12. x(∈eq) v
. . 13. ∃M,N:T List, y:T. eq(x,y) & v = (M @ (y.N))
. . ⊢  ∃M,N:T List, y:T. eq(x,y) & u.v = (M @ (y.N))
. . | by Repeat (Analyze -1)
. . |
. . 13. M : T List
. . 14. N : T List
. . 15. y : T
. . 16. eq(x,y)
. . 17. v = (M @ (y.N))
. . ⊢  ∃M,N:T List, y:T. eq(x,y) & u.v = (M @ (y.N))
. . | by Witness u.M THEN Witness N THEN Witness y
. . |
. . ⊢  eq(x,y) & u.v = ((u.M) @ (y.N))  by Auto
. . |
. . ⊢  u.v = ((u.M) @ (y.N))  by HypSubst -1 0 THEN Reduce 0
. . |
. . ⊢  u.(M @ (y.N)) = u.(M @ (y.N))  by Auto
END
```