```
append_is_member_lemma  THM

⊢  ∀eq:{T=₂}, L:T List, x:T.
|  (∃M,N:T List. L = (M @ (x.N))) ⇒ x(∈eq) L
| by RepeatFor 2 (Analyze 0) THEN DiscreteEq -1 THEN UnivCD
|
1. T : Type
2. eq : {T=₂}
3. eq ∈ T→T→𝔹
4. ∀x,y:T. eq(x,y) ⟺ x = y
5. L : T List
6. x : T
7. ∃M,N:T List. L = (M @ (x.N))
⊢  x(∈eq) L
| by ListInd 5
|    THEN
|    Rewrite
|    (HigherC is_member_unrollC THENC TryC (HigherC assert_evalC))
|    0
|
|\
| 6. x : T
| ⊢  (∃M,N:T List. nil = (M @ (x.N))) ⇒ False  by Auto
| |
| 7. ∃M,N:T List. nil = (M @ (x.N))
| ⊢  False  by Analyze -1 THEN Analyze -1
| |
| 7. M : T List
| 8. N : T List
| 9. nil = (M @ (x.N))
| ⊢  False  by ListInd 7
| |
| |\
| | 8. N : T List
| | ⊢  nil = (nil @ (x.N)) ⇒ False  by AbReduce 0 THEN Analyze 0
| | |
| | 9. nil = x.N
| | ⊢  False
| |   by ApFunToHypEquands ‘z’ (Case of z; nil → 0 ; h.t → 1) ℤ 9
| |      THEN
| |      Reduce -1
| |
| .\
| . 9. u : T
| . 10. v : T List
| . 11. nil = (v @ (x.N)) ⇒ False
| . ⊢  nil = ((u.v) @ (x.N)) ⇒ False  by AbReduce 0 THEN Analyze 0
| . |
| . 12. nil = u.(v @ (x.N))
| . ⊢  False
| .   by ApFunToHypEquands ‘z’ (Case of z; nil → 0 ; h.t → 1) ℤ -1
| .      THEN
| .      Reduce -1
|
.\
```

```
. 7. u : T
. 8. v : T List
. 9. (∃M,N:T List. v = (M @ (x.N))) ⇒ x(∈eq) v
. ⊢  (∃M,N:T List. u.v = (M @ (x.N)))
. |  ⇒
. |  if eq(x,u)→ true₂ else x(∈eq) v fi
. | by GenConcl (eq(x,u) = b) THEN BoolInd -2
. |    THEN
. |    Rewrite
. |    (HigherC ifthenelse_evalC THENC TryC (HigherC assert_evalC))
. |    0
. |
. |\
. | 10. eq(x,u) = true₂
. | ⊢  (∃M,N:T List. u.v = (M @ (x.N))) ⇒ True
. |   by Analyze 0 THEN Trivial
. |
. .\
. . 10. eq(x,u) = false₂
. . ⊢  (∃M,N:T List. u.v = (M @ (x.N))) ⇒ x(∈eq) v  by Analyze 0
. . |
. . 11. ∃M,N:T List. u.v = (M @ (x.N))
. . ⊢  x(∈eq) v  by HypBackchain
. . |
. . ⊢  ∃M,N:T List. v = (M @ (x.N))  by Analyze 11 THEN Analyze -1
. . |
. . 11. M : T List
. . 12. N : T List
. . 13. u.v = (M @ (x.N))
. . ⊢  ∃M,N:T List. v = (M @ (x.N))
. . | by ListInd 11 THEN AbReduce 0 THEN Analyze 0
. . |
. . |\
. . | 13. u.v = x.N
. . | ⊢  ∃M,N:T List. v = (M @ (x.N))
. . | | by ApFunToHypEquands ‘X’ (Case of X; nil → x ; h.t → h) T 13
. . | |    THEN
. . | |    Reduce -1
. . | |
. . | 14. u = x
. . | ⊢  ∃M,N:T List. v = (M @ (x.N))  by Assert (x = u)
. . | |
. . | 15. x = u
. . | ⊢  ∃M,N:T List. v = (M @ (x.N))  by FwdThru 4 [-1]
. . | |
. . | 16. eq(x,u)
. . | ⊢  ∃M,N:T List. v = (M @ (x.N))
. . |   by HypSubst -7 -1 THEN Reduce -1 THEN Trivial
. . |
. . .\
. . . 13. u1 : T
. . . 14. v1 : T List
. . . 15. u.v = (v1 @ (x.N)) ⇒ (∃M,N:T List. v = (M @ (x.N)))
. . . 16. u.v = u1.(v1 @ (x.N))
. . . ⊢  ∃M,N:T List. v = (M @ (x.N))
```

```
. . . | by ApFunToHypEquands ‘X’ (Case of X; nil → nil ; h.t → t)
. . . |    (T List)
. . . |    16
. . . |
. . . 17. (Case of u.v; nil → nil ; h.t → t)
. . . |   =
. . . |   (Case of u1.(v1 @ (x.N)); nil → nil ; h.t → t)
. . . ⊢  ∃M,N:T List. v = (M @ (x.N))  by Analyze -1 THEN Reduce 0
. . . |
. . . ⊢  ∃M,N:T List. v = (M @ (x.N))  by Witness v1 THEN Witness N
. . . |
. . . ⊢  v = (v1 @ (x.N))
. . . | by ApFunToHypEquands ‘zz’ (Case of zz; nil → zz ; h.t → t)
. . . |    (T List)
. . . |    -2
. . . |    THEN
. . . |    Reduce -1
. . . |
. . . 18. v = (v1 @ (x.N))
. . . ⊢  v = (v1 @ (x.N))  by Trivial
END
```