```
assert-member-paren  THM

⊢  ∀T:Type, E:(T→T→𝔹).
|  (∀x,y:T. E(x,y) ⟺ x = y)
|  ⇒
|  (∀i:T, s:(T+T) List.
|   member-paren(x,y.E(x,y);i;s) ⟺ (inl(i) ∈ s) ∨ (inr(i) ∈ s))
| by UnivCD THEN (Unfold `member-paren` 0)
|    THEN
|    (RWO
|     Thm* ∀L:T List, P:(T→𝔹). (∃x∈L.P(x)) ⟺ (∃i:ℕ||L||. P(L[i]))
|     {assert_l_bexists}
|     0)
|    THEN
|    (Unfold `l_member` 0)
|    THEN
|    SplitOrHyps
|    THEN
|    ExRepD
|    THEN
|    (Try
|     ((((InstConcl [i@0]) THEN (RevHypSubst -1 0)) THEN (Reduce 0))
|      THEN
|      EasyHyp))
|
1. T : Type
2. E : T→T→𝔹
3. ∀x,y:T. E(x,y) ⟺ x = y
4. i : T
5. s : (T+T) List
6. i1 : ℕ||s||
7. InjCase(s[i1]; a. E(a,i), E(a,i))
⊢  (∃i@0:ℕ. i@0<||s|| & inl(i) = s[i@0])
|  ∨ (∃i@0:ℕ. i@0<||s|| & inr(i) = s[i@0])
| by (MoveToConcl -1) THEN (GenConcl (s[i1] = x)) THEN (Analyze -2)
|    THEN
|    (Reduce 0)
|
|\
| 1. T : Type
| 2. E : T→T→𝔹
| 3. ∀x,y:T. E(x,y) ⟺ x = y
| 4. i : T
| 5. s : (T+T) List
| 6. i1 : ℕ||s||
| 7. x1 : T
| 8. s[i1] = inl(x1)
| 9. E(x1,i)
| ⊢  (∃i@0:ℕ. i@0<||s|| & inl(i) = s[i@0])
|    ∨ (∃i@0:ℕ. i@0<||s|| & inr(i) = s[i@0])  by Obvious
|
.\
. 1. T : Type
. 2. E : T→T→𝔹
. 3. ∀x,y:T. E(x,y) ⟺ x = y
```

```
. 4. i : T
. 5. s : (T+T) List
. 6. i1 : ℕ||s||
. 7. y : T
. 8. s[i1] = inr(y)
. 9. E(y,i)
. ⊢  (∃i@0:ℕ. i@0<||s|| & inl(i) = s[i@0])
.    ∨ (∃i@0:ℕ. i@0<||s|| & inr(i) = s[i@0])  by Obvious
END
```