```
cons_sublist_cons  THM

⊢  ∀T:Type, x1,x2:T, L1,L2:T List.
|  [x1 / L1] ⊆ [x2 / L2] ⟺ x1 = x2 & L1 ⊆ L2 ∨ [x1 / L1] ⊆ L2
| by Auto THEN All (Unfold `sublist`) THEN All Reduce THEN  ExRepD
|
|\
| 1. T : Type
| 2. x1 : T
| 3. x2 : T
| 4. L1 : T List
| 5. L2 : T List
| 6. f : ℕ(||L1||+1)→ℕ(||L2||+1)
| 7. increasing(f;||L1||+1)
| 8. ∀j:ℕ(||L1||+1). [x1 / L1][j] = [x2 / L2][(f(j))]
| ⊢  x1 = x2
| |  & (∃f:(ℕ||L1||→ℕ||L2||).
| |     increasing(f;||L1||) & (∀j:ℕ||L1||. L1[j] = L2[(f(j))]))
| |  ∨ (∃f:(ℕ(||L1||+1)→ℕ||L2||).
| |     increasing(f;||L1||+1)
| |     & (∀j:ℕ(||L1||+1). [x1 / L1][j] = L2[(f(j))]))
| | by InstHyp [0] (-1) THENA Auto'
| |
| 1. T : Type
| 2. x1 : T
| 3. x2 : T
| 4. L1 : T List
| 5. L2 : T List
| 6. f : ℕ(||L1||+1)→ℕ(||L2||+1)
| 7. increasing(f;||L1||+1)
| 8. ∀j:ℕ(||L1||+1). [x1 / L1][j] = [x2 / L2][(f(j))]
| 9. [x1 / L1][0] = [x2 / L2][(f(0))]
| ⊢  x1 = x2
| |  & (∃f:(ℕ||L1||→ℕ||L2||).
| |     increasing(f;||L1||) & (∀j:ℕ||L1||. L1[j] = L2[(f(j))]))
| |  ∨ (∃f:(ℕ(||L1||+1)→ℕ||L2||).
| |     increasing(f;||L1||+1)
| |     & (∀j:ℕ(||L1||+1). [x1 / L1][j] = L2[(f(j))]))
| | by Decide f(0)=₂0 THENA Auto
|
| |    THEN RW assert_pushdownC (-1) THENA Auto
| |
| |\
| | 1. T : Type
| | 2. x1 : T
| | 3. x2 : T
| | 4. L1 : T List
| | 5. L2 : T List
| | 6. f : ℕ(||L1||+1)→ℕ(||L2||+1)
| | 7. increasing(f;||L1||+1)
| | 8. ∀j:ℕ(||L1||+1). [x1 / L1][j] = [x2 / L2][(f(j))]
| | 9. [x1 / L1][0] = [x2 / L2][(f(0))]
| | 10. f(0) = 0
| | ⊢  x1 = x2
| | |  & (∃f:(ℕ||L1||→ℕ||L2||).
```

```
| | |     increasing(f;||L1||) & (∀j:ℕ||L1||. L1[j] = L2[(f(j))]))
| | |  ∨ (∃f:(ℕ(||L1||+1)→ℕ||L2||).
| | |     increasing(f;||L1||+1)
| | |     & (∀j:ℕ(||L1||+1). [x1 / L1][j] = L2[(f(j))]))
| | | by OrLeft THENA (Auto THEN Reduce 0 THEN Auto')
| | |
| | 1. T : Type
| | 2. x1 : T
| | 3. x2 : T
| | 4. L1 : T List
| | 5. L2 : T List
| | 6. f : ℕ(||L1||+1)→ℕ(||L2||+1)
| | 7. increasing(f;||L1||+1)
| | 8. ∀j:ℕ(||L1||+1). [x1 / L1][j] = [x2 / L2][(f(j))]
| | 9. [x1 / L1][0] = [x2 / L2][(f(0))]
| | 10. f(0) = 0
| | ⊢  x1 = x2
| | |  & (∃f:(ℕ||L1||→ℕ||L2||).
| | |     increasing(f;||L1||) & (∀j:ℕ||L1||. L1[j] = L2[(f(j))]))
| | | by Auto
| | |
| | |\
| | | 1. T : Type
| | | 2. x1 : T
| | | 3. x2 : T
| | | 4. L1 : T List
| | | 5. L2 : T List
| | | 6. f : ℕ(||L1||+1)→ℕ(||L2||+1)
| | | 7. increasing(f;||L1||+1)
| | | 8. ∀j:ℕ(||L1||+1). [x1 / L1][j] = [x2 / L2][(f(j))]
| | | 9. [x1 / L1][0] = [x2 / L2][(f(0))]
| | | 10. f(0) = 0
| | | ⊢  x1 = x2  by HypSubstSq (-1) (-2)
| | | |
| | | 1. T : Type
| | | 2. x1 : T
| | | 3. x2 : T
| | | 4. L1 : T List
| | | 5. L2 : T List
| | | 6. f : ℕ(||L1||+1)→ℕ(||L2||+1)
| | | 7. increasing(f;||L1||+1)
| | | 8. ∀j:ℕ(||L1||+1). [x1 / L1][j] = [x2 / L2][(f(j))]
| | | 9. [x1 / L1][0] = [x2 / L2][0]
| | | 10. f(0) = 0
| | | ⊢  x1 = x2  by RWO "select_cons_hd" (-2) THEN Auto
| | |
| | .\
| | . 1. T : Type
| | . 2. x1 : T
| | . 3. x2 : T
| | . 4. L1 : T List
| | . 5. L2 : T List
| | . 6. f : ℕ(||L1||+1)→ℕ(||L2||+1)
| | . 7. increasing(f;||L1||+1)
| | . 8. ∀j:ℕ(||L1||+1). [x1 / L1][j] = [x2 / L2][(f(j))]
```

```
| | . 9. [x1 / L1][0] = [x2 / L2][(f(0))]
| | . 10. f(0) = 0
| | . ⊢  ∃f:(ℕ||L1||→ℕ||L2||).
| | . |  increasing(f;||L1||) & (∀j:ℕ||L1||. L1[j] = L2[(f(j))])
| | . | by InstConcl [λi.f(i+1)-1] THEN Reduce 0 THEN Auto'
| | . |
| | . |\
| | . | .....subterm .....
| | . | 1. T : Type
| | . | 2. x1 : T
| | . | 3. x2 : T
| | . | 4. L1 : T List
| | . | 5. L2 : T List
| | . | 6. f : ℕ(||L1||+1)→ℕ(||L2||+1)
| | . | 7. increasing(f;||L1||+1)
| | . | 8. ∀j:ℕ(||L1||+1). [x1 / L1][j] = [x2 / L2][(f(j))]
| | . | 9. [x1 / L1][0] = [x2 / L2][(f(0))]
| | . | 10. f(0) = 0
| | . | 11. i : ℕ||L1||
| | . | ⊢  f(i+1)-1 ∈ ℕ||L2||
| | . |   by InstLemma `increasing_lower_bound` [||L1||+1;f;i+1
| | . |      ] THEN Auto'
| | . |
| | . |\
| | . | 1. T : Type
| | . | 2. x1 : T
| | . | 3. x2 : T
| | . | 4. L1 : T List
| | . | 5. L2 : T List
| | . | 6. f : ℕ(||L1||+1)→ℕ(||L2||+1)
| | . | 7. increasing(f;||L1||+1)
| | . | 8. ∀j:ℕ(||L1||+1). [x1 / L1][j] = [x2 / L2][(f(j))]
| | . | 9. [x1 / L1][0] = [x2 / L2][(f(0))]
| | . | 10. f(0) = 0
| | . | ⊢  increasing(λi.f(i+1)-1;||L1||)
| | . |   by  Unfold `increasing` 0 THEN Reduce 0 THEN Auto
| | . |
| | . |      THEN AllHyps (\j. Unfold `increasing` j THEN InstHyp [i+1
| | . |      ] j THEN Auto')
| | . |
| | . |
| | . .\
| | . . 1. T : Type
| | . . 2. x1 : T
| | . . 3. x2 : T
| | . . 4. L1 : T List
| | . . 5. L2 : T List
| | . . 6. f : ℕ(||L1||+1)→ℕ(||L2||+1)
| | . . 7. increasing(f;||L1||+1)
| | . . 8. ∀j:ℕ(||L1||+1). [x1 / L1][j] = [x2 / L2][(f(j))]
| | . . 9. [x1 / L1][0] = [x2 / L2][(f(0))]
| | . . 10. f(0) = 0
| | . . 11. j : ℕ||L1||
| | . . ⊢  L1[j] = L2[(f(j+1)-1)]
| | . . | by AllHyps (InstHyp [j+1]) THENA Auto
```

```
| | . . |
| | . . 1. T : Type
| | . . 2. x1 : T
| | . . 3. x2 : T
| | . . 4. L1 : T List
| | . . 5. L2 : T List
| | . . 6. f : ℕ(||L1||+1)→ℕ(||L2||+1)
| | . . 7. increasing(f;||L1||+1)
| | . . 8. ∀j:ℕ(||L1||+1). [x1 / L1][j] = [x2 / L2][(f(j))]
| | . . 9. [x1 / L1][0] = [x2 / L2][(f(0))]
| | . . 10. f(0) = 0
| | . . 11. j : ℕ||L1||
| | . . 12. [x1 / L1][(j+1)] = [x2 / L2][(f(j+1))]
| | . . ⊢  L1[j] = L2[(f(j+1)-1)]
| | . . | by RWO "select_cons_tl" (-1) THENA Auto'
| | . . |
| | . . |\
| | . . | .....rewrite subgoal .....
| | . . | 1. T : Type
| | . . | 2. x1 : T
| | . . | 3. x2 : T
| | . . | 4. L1 : T List
| | . . | 5. L2 : T List
| | . . | 6. f : ℕ(||L1||+1)→ℕ(||L2||+1)
| | . . | 7. increasing(f;||L1||+1)
| | . . | 8. ∀j:ℕ(||L1||+1). [x1 / L1][j] = [x2 / L2][(f(j))]
| | . . | 9. [x1 / L1][0] = [x2 / L2][(f(0))]
| | . . | 10. f(0) = 0
| | . . | 11. j : ℕ||L1||
| | . . | 12. [x1 / L1][(j+1)] = [x2 / L2][(f(j+1))]
| | . . | ⊢  0<f(j+1)
| | . . |   by InstLemma `increasing_lower_bound` [||L1||+1;f;j+1
| | . . |      ] THEN Auto'
| | . . |
| | . . .\
| | . . . ...
| |

| .\
| . 1. T : Type
| . 2. x1 : T
| . 3. x2 : T
| . 4. L1 : T List
| . 5. L2 : T List
| . 6. f : ℕ(||L1||+1)→ℕ(||L2||+1)
| . 7. increasing(f;||L1||+1)
| . 8. ∀j:ℕ(||L1||+1). [x1 / L1][j] = [x2 / L2][(f(j))]
| . 9. [x1 / L1][0] = [x2 / L2][(f(0))]
| . 10. f(0) ≠ 0
| . ⊢  x1 = x2
| . |  & (∃f:(ℕ||L1||→ℕ||L2||).
| . |     increasing(f;||L1||) & (∀j:ℕ||L1||. L1[j] = L2[(f(j))]))
| . |  ∨ (∃f:(ℕ(||L1||+1)→ℕ||L2||).
| . |     increasing(f;||L1||+1)
| . |     & (∀j:ℕ(||L1||+1). [x1 / L1][j] = L2[(f(j))]))
| . | by OrRight THENA Auto'
```

```
| . |
| . 1. T : Type
| . 2. x1 : T
| . 3. x2 : T
| . 4. L1 : T List
| . 5. L2 : T List
| . 6. f : ℕ(||L1||+1)→ℕ(||L2||+1)
| . 7. increasing(f;||L1||+1)
| . 8. ∀j:ℕ(||L1||+1). [x1 / L1][j] = [x2 / L2][(f(j))]
| . 9. [x1 / L1][0] = [x2 / L2][(f(0))]
| . 10. f(0) ≠ 0
| . ⊢  ∃f:(ℕ(||L1||+1)→ℕ||L2||).
| . |  increasing(f;||L1||+1)
| . |  & (∀j:ℕ(||L1||+1). [x1 / L1][j] = L2[(f(j))])
| . | by InstConcl [λj.f(j)-1] THEN Auto'
| . |
| . |\
| . | .....subterm .....
| . | 1. T : Type
| . | 2. x1 : T
| . | 3. x2 : T
| . | 4. L1 : T List
| . | 5. L2 : T List
| . | 6. f : ℕ(||L1||+1)→ℕ(||L2||+1)
| . | 7. increasing(f;||L1||+1)
| . | 8. ∀j:ℕ(||L1||+1). [x1 / L1][j] = [x2 / L2][(f(j))]
| . | 9. [x1 / L1][0] = [x2 / L2][(f(0))]
| . | 10. f(0) ≠ 0
| . | 11. j : ℕ(||L1||+1)
| . | ⊢  f(j)-1 ∈ ℕ||L2||
| . |   by InstLemma `increasing_lower_bound` [||L1||+1;f;j
| . |      ] THEN Auto'
| . |
| . |\
| . | 1. T : Type
| . | 2. x1 : T
| . | 3. x2 : T
| . | 4. L1 : T List
| . | 5. L2 : T List
| . | 6. f : ℕ(||L1||+1)→ℕ(||L2||+1)
| . | 7. increasing(f;||L1||+1)
| . | 8. ∀j:ℕ(||L1||+1). [x1 / L1][j] = [x2 / L2][(f(j))]
| . | 9. [x1 / L1][0] = [x2 / L2][(f(0))]
| . | 10. f(0) ≠ 0
| . | ⊢  increasing(λj.f(j)-1;||L1||+1)
| . |   by Unfold `increasing` 0 THEN Reduce 0 THEN Auto
| . |
| . |      THEN AllHyps (\j.Unfold `increasing` j THEN InstHyp [i
| . |      ] j THEN Auto')
| . |
| . .\
| . . 1. T : Type
| . . 2. x1 : T
| . . 3. x2 : T
| . . 4. L1 : T List
```

```
| . . 5. L2 : T List
| . . 6. f : ℕ(||L1||+1)→ℕ(||L2||+1)
| . . 7. increasing(f;||L1||+1)
| . . 8. ∀j:ℕ(||L1||+1). [x1 / L1][j] = [x2 / L2][(f(j))]
| . . 9. [x1 / L1][0] = [x2 / L2][(f(0))]
| . . 10. f(0) ≠ 0
| . . 11. j : ℕ(||L1||+1)
| . . ⊢  [x1 / L1][j] = L2[((λj.f(j)-1)(j))]
| . . | by Reduce 0 THEN AllHyps (InstHyp [j]) THENA Auto
| . . |
| . . 1. T : Type
| . . 2. x1 : T
| . . 3. x2 : T
| . . 4. L1 : T List
| . . 5. L2 : T List
| . . 6. f : ℕ(||L1||+1)→ℕ(||L2||+1)
| . . 7. increasing(f;||L1||+1)
| . . 8. ∀j:ℕ(||L1||+1). [x1 / L1][j] = [x2 / L2][(f(j))]
| . . 9. [x1 / L1][0] = [x2 / L2][(f(0))]
| . . 10. f(0) ≠ 0
| . . 11. j : ℕ(||L1||+1)
| . . 12. [x1 / L1][j] = [x2 / L2][(f(j))]
| . . ⊢  [x1 / L1][j] = L2[(f(j)-1)]
| . . | by InstLemma `increasing_lower_bound` [||L1||+1;f;j
| . . |    ] THENA Auto
| . . |
| . . 1. T : Type
| . . 2. x1 : T
| . . 3. x2 : T
| . . 4. L1 : T List
| . . 5. L2 : T List
| . . 6. f : ℕ(||L1||+1)→ℕ(||L2||+1)
| . . 7. increasing(f;||L1||+1)
| . . 8. ∀j:ℕ(||L1||+1). [x1 / L1][j] = [x2 / L2][(f(j))]
| . . 9. [x1 / L1][0] = [x2 / L2][(f(0))]
| . . 10. f(0) ≠ 0
| . . 11. j : ℕ(||L1||+1)
| . . 12. [x1 / L1][j] = [x2 / L2][(f(j))]
| . . 13. f(0)+j≤f(j)
| . . ⊢  [x1 / L1][j] = L2[(f(j)-1)]  by HypSubst (-2) 0 THEN Auto'
| . . |
| . . 1. T : Type
| . . 2. x1 : T
| . . 3. x2 : T
| . . 4. L1 : T List
| . . 5. L2 : T List
| . . 6. f : ℕ(||L1||+1)→ℕ(||L2||+1)
| . . 7. increasing(f;||L1||+1)
| . . 8. ∀j:ℕ(||L1||+1). [x1 / L1][j] = [x2 / L2][(f(j))]
| . . 9. [x1 / L1][0] = [x2 / L2][(f(0))]
| . . 10. f(0) ≠ 0
| . . 11. j : ℕ(||L1||+1)
| . . 12. [x1 / L1][j] = [x2 / L2][(f(j))]
| . . 13. f(0)+j≤f(j)
| . . ⊢  [x2 / L2][(f(j))] = L2[(f(j)-1)]
```

```
| . .   by RWO "select_cons_tl" 0 THEN Auto'
|
.\
. 1. T : Type
. 2. x1 : T
. 3. x2 : T
. 4. L1 : T List
. 5. L2 : T List
. 6. x1 = x2
. |  & (∃f:(ℕ||L1||→ℕ||L2||).
. |     increasing(f;||L1||) & (∀j:ℕ||L1||. L1[j] = L2[(f(j))]))
. |  ∨ (∃f:(ℕ(||L1||+1)→ℕ||L2||).
. |     increasing(f;||L1||+1)
. |     & (∀j:ℕ(||L1||+1). [x1 / L1][j] = L2[(f(j))]))
. ⊢  ∃f:(ℕ(||L1||+1)→ℕ(||L2||+1)).
. |  increasing(f;||L1||+1)
. |  & (∀j:ℕ(||L1||+1). [x1 / L1][j] = [x2 / L2][(f(j))])
. | by D (-1) THEN ExRepD
. |
. |\
. | 1. T : Type
. | 2. x1 : T
. | 3. x2 : T
. | 4. L1 : T List
. | 5. L2 : T List
. | 6. x1 = x2
. | 7. f : ℕ||L1||→ℕ||L2||
. | 8. increasing(f;||L1||)
. | 9. ∀j:ℕ||L1||. L1[j] = L2[(f(j))]
. | ⊢  ∃f:(ℕ(||L1||+1)→ℕ(||L2||+1)).
. | |  increasing(f;||L1||+1)
. | |  & (∀j:ℕ(||L1||+1). [x1 / L1][j] = [x2 / L2][(f(j))])
. | | by Unfold `increasing` 0 THEN
. | InstConcl [λi.if i=₂0→ 0
. | |                                               else f(i-1)+1 fi]
. | |
. | THEN Reduce 0
. | |
. | |\
. | | .....wf .....
. | | 1. T : Type
. | | 2. x1 : T
. | | 3. x2 : T
. | | 4. L1 : T List
. | | 5. L2 : T List
. | | 6. x1 = x2
. | | 7. f : ℕ||L1||→ℕ||L2||
. | | 8. increasing(f;||L1||)
. | | 9. ∀j:ℕ||L1||. L1[j] = L2[(f(j))]
. | | ⊢  (λi.if i=₂0→ 0 else f(i-1)+1 fi) ∈ ℕ(||L1||+1)→ℕ(||L2||+1)
. | | | by MemCD THENA Auto
. | | |
. | | .....subterm .....
. | | 1. T : Type
. | | 2. x1 : T
```

```
. | | 3. x2 : T
. | | 4. L1 : T List
. | | 5. L2 : T List
. | | 6. x1 = x2
. | | 7. f : ℕ||L1||→ℕ||L2||
. | | 8. increasing(f;||L1||)
. | | 9. ∀j:ℕ||L1||. L1[j] = L2[(f(j))]
. | | 10. i : ℕ(||L1||+1)
. | | ⊢  if i=₂0→ 0 else f(i-1)+1 fi ∈ ℕ(||L2||+1)
. | |   by SplitOnConclITE THEN Auto'
. | |
. | |\
. | | 1. T : Type
. | | 2. x1 : T
. | | 3. x2 : T
. | | 4. L1 : T List
. | | 5. L2 : T List
. | | 6. x1 = x2
. | | 7. f : ℕ||L1||→ℕ||L2||
. | | 8. increasing(f;||L1||)
. | | 9. ∀j:ℕ||L1||. L1[j] = L2[(f(j))]
. | | ⊢  (∀i:ℕ(||L1||+1-1).
. | | |   if i=₂0→ 0 else f(i-1)+1 fi<if i+1=₂0→ 0 else f(i+1-1)+1 fi)
. | | |  & (∀j:ℕ(||L1||+1).
. | | |      [x1 / L1][j] = [x2 / L2][if j=₂0→ 0 else f(j-1)+1 fi])
. | | | by Auto
. | | |
. | | |\
. | | | 1. T : Type
. | | | 2. x1 : T
. | | | 3. x2 : T
. | | | 4. L1 : T List
. | | | 5. L2 : T List
. | | | 6. x1 = x2
. | | | 7. f : ℕ||L1||→ℕ||L2||
. | | | 8. increasing(f;||L1||)
. | | | 9. ∀j:ℕ||L1||. L1[j] = L2[(f(j))]
. | | | 10. i : ℕ(||L1||+1-1)
. | | | ⊢  if i=₂0→ 0 else f(i-1)+1 fi<if i+1=₂0→ 0 else f(i+1-1)+1 fi
. | | | | by Repeat (SplitOnConclITE THEN Auto')
. | | | |
. | | | .....falsecase .....
. | | | 1. T : Type
. | | | 2. x1 : T
. | | | 3. x2 : T
. | | | 4. L1 : T List
. | | | 5. L2 : T List
. | | | 6. x1 = x2
. | | | 7. f : ℕ||L1||→ℕ||L2||
. | | | 8. increasing(f;||L1||)
. | | | 9. ∀j:ℕ||L1||. L1[j] = L2[(f(j))]
. | | | 10. i : ℕ(||L1||+1-1)
. | | | 11. ¬i = 0
. | | | 12. ¬i+1 = 0
. | | | ⊢  f(i-1)+1<f(i+1-1)+1
```

```
. | | |   by AllHyps (\j. Unfold `increasing` j THEN InstHyp [i-1
. | | |      ] j THEN Auto')
. | | |
. | | .\
. | | . 1. T : Type
. | | . 2. x1 : T
. | | . 3. x2 : T
. | | . 4. L1 : T List
. | | . 5. L2 : T List
. | | . 6. x1 = x2
. | | . 7. f : ℕ||L1||→ℕ||L2||
. | | . 8. increasing(f;||L1||)
. | | . 9. ∀j:ℕ||L1||. L1[j] = L2[(f(j))]
. | | . 10. j : ℕ(||L1||+1)
. | | . ⊢  [x1 / L1][j] = [x2 / L2][if j=_2 0→ 0 else f(j-1)+1 fi]
. | | . | by SplitOnConclITE THENA Auto
. | | . |
. | | . |\
. | | . | ...
. | | . |
. | | . .\
. | | . . .....falsecase .....
. | | . . 1. T : Type
. | | . . 2. x1 : T
. | | . . 3. x2 : T
. | | . . 4. L1 : T List
. | | . . 5. L2 : T List
. | | . . 6. x1 = x2
. | | . . 7. f : ℕ||L1||→ℕ||L2||
. | | . . 8. increasing(f;||L1||)
. | | . . 9. ∀j:ℕ||L1||. L1[j] = L2[(f(j))]
. | | . . 10. j : ℕ(||L1||+1)
. | | . . 11. ¬j = 0
. | | . . ⊢  [x1 / L1][j] = [x2 / L2][(f(j-1)+1)]
. | | . . | by RWO "select_cons_tl" 0 THEN Auto'
. | | . . |
. | | . . 1. T : Type
. | | . . 2. x1 : T
. | | . . 3. x2 : T
. | | . . 4. L1 : T List
. | | . . 5. L2 : T List
. | | . . 6. x1 = x2
. | | . . 7. f : ℕ||L1||→ℕ||L2||
. | | . . 8. increasing(f;||L1||)
. | | . . 9. ∀j:ℕ||L1||. L1[j] = L2[(f(j))]
. | | . . 10. j : ℕ(||L1||+1)
. | | . . 11. ¬j = 0
. | | . . ⊢  L1[(j-1)] = L2[(f(j-1)+1-1)]
. | | . . | by Subst (f(j-1)+1-1) ~ (f(j-1)) 0 THEN Auto'
. | | . . |
. | | . . 1. T : Type
. | | . . 2. x1 : T
. | | . . 3. x2 : T
. | | . . 4. L1 : T List
. | | . . 5. L2 : T List
```

```
. | | . . 6. x1 = x2
. | | . . 7. f : ℕ||L1||→ℕ||L2||
. | | . . 8. increasing(f;||L1||)
. | | . . 9. ∀j:ℕ||L1||. L1[j] = L2[(f(j))]
. | | . . 10. j : ℕ(||L1||+1)
. | | . . 11. ¬j = 0
. | | . . ⊢  L1[(j-1)] = L2[(f(j-1))]  by EasyHyp

. | |
. | .\
. | . .....wf .....
. | . 1. T : Type
. | . 2. x1 : T
. | . 3. x2 : T
. | . 4. L1 : T List
. | . 5. L2 : T List
. | . 6. x1 = x2
. | . 7. f : ℕ||L1||→ℕ||L2||
. | . 8. increasing(f;||L1||)
. | . 9. ∀j:ℕ||L1||. L1[j] = L2[(f(j))]
. | . 10. f1 : ℕ(||L1||+1)→ℕ(||L2||+1)
. | . ⊢  ((∀i:ℕ(||L1||+1-1). f1(i)<f1(i+1))
. | .     & (∀j:ℕ(||L1||+1). [x1 / L1][j] = [x2 / L2][(f1(j))]))
. | .    ∈ Prop  by Auto
. |
. .\
. . 1. T : Type
. . 2. x1 : T
. . 3. x2 : T
. . 4. L1 : T List
. . 5. L2 : T List
. . 6. f : ℕ(||L1||+1)→ℕ||L2||
. . 7. increasing(f;||L1||+1)
. . 8. ∀j:ℕ(||L1||+1). [x1 / L1][j] = L2[(f(j))]
. . ⊢  ∃f:(ℕ(||L1||+1)→ℕ(||L2||+1)).
. . |  increasing(f;||L1||+1)
. . |  & (∀j:ℕ(||L1||+1). [x1 / L1][j] = [x2 / L2][(f(j))])
. . | by InstConcl [λj.f(j)+1] THEN Auto’
. . |
. . |\
. . | 1. T : Type
. . | 2. x1 : T
. . | 3. T
. . | 4. L1 : T List
. . | 5. L2 : T List
. . | 6. f : ℕ(||L1||+1)→ℕ||L2||
. . | 7. increasing(f;||L1||+1)
. . | 8. ∀j:ℕ(||L1||+1). [x1 / L1][j] = L2[(f(j))]
. . | ⊢  increasing(λj.f(j)+1;||L1||+1)
. . |   by Unfold ‘increasing‘ 0 THEN Reduce 0 THEN Auto
. . |
. . |      THEN AllHyps (\j. Unfold ‘increasing‘ j THEN InstHyp [i
. . |      ] j THEN Auto’)
. . |
. . .\
. . . 1. T : Type
```

```
. . . 2. x1 : T
. . . 3. x2 : T
. . . 4. L1 : T List
. . . 5. L2 : T List
. . . 6. f : ℕ(||L1||+1)→ℕ||L2||
. . . 7. increasing(f;||L1||+1)
. . . 8. ∀j:ℕ(||L1||+1). [x1 / L1][j] = L2[(f(j))]
. . . 9. j : ℕ(||L1||+1)
. . . ⊢  [x1 / L1][j] = [x2 / L2][((λj.f(j)+1)(j))]  by Reduce 0
. . . |
. . . 1. T : Type
. . . 2. x1 : T
. . . 3. x2 : T
. . . 4. L1 : T List
. . . 5. L2 : T List
. . . 6. f : ℕ(||L1||+1)→ℕ||L2||
. . . 7. increasing(f;||L1||+1)
. . . 8. ∀j:ℕ(||L1||+1). [x1 / L1][j] = L2[(f(j))]
. . . 9. j : ℕ(||L1||+1)
. . . ⊢  [x1 / L1][j] = [x2 / L2][(f(j)+1)]
. . . | by Subst [x2 / L2][(f(j)+1)] = L2[(f(j))] 0
. . . |
. . . |\
. . . | .....equality .....
. . . | 1. T : Type
. . . | 2. x1 : T
. . . | 3. x2 : T
. . . | 4. L1 : T List
. . . | 5. L2 : T List
. . . | 6. f : ℕ(||L1||+1)→ℕ||L2||
. . . | 7. increasing(f;||L1||+1)
. . . | 8. ∀j:ℕ(||L1||+1). [x1 / L1][j] = L2[(f(j))]
. . . | 9. j : ℕ(||L1||+1)
. . . | ⊢  [x2 / L2][(f(j)+1)] = L2[(f(j))]
. . . |   by RWO "select_cons_tl" 0 THEN Auto'
. . . |
. . . |\
. . . | 1. T : Type
. . . | 2. x1 : T
. . . | 3. T
. . . | 4. L1 : T List
. . . | 5. L2 : T List
. . . | 6. f : ℕ(||L1||+1)→ℕ||L2||
. . . | 7. increasing(f;||L1||+1)
. . . | 8. ∀j:ℕ(||L1||+1). [x1 / L1][j] = L2[(f(j))]
. . . | 9. j : ℕ(||L1||+1)
. . . | ⊢  [x1 / L1][j] = L2[(f(j))]  by EasyHyp
. . . |
. . . .\
. . . . .....wf .....
. . . . 1. T : Type
. . . . 2. x1 : T

. . . . 3. T
. . . . 4. L1 : T List
. . . . 5. L2 : T List
```

```
. . . . 6. f : ℕ(||L1||+1)→ℕ||L2||
. . . . 7. increasing(f;||L1||+1)
. . . . 8. ∀j:ℕ(||L1||+1). [x1 / L1][j] = L2[(f(j))]
. . . . 9. j : ℕ(||L1||+1)
. . . . 10. z : T
. . . . ⊢  ([x1 / L1][j] = z) ∈ Prop  by Auto
. . . . |
. . . . 1. T : Type
. . . . 2. x1 : T
. . . . 3. T
. . . . 4. L1 : T List
. . . . 5. L2 : T List
. . . . 6. f : ℕ(||L1||+1)→ℕ||L2||
. . . . 7. increasing(f;||L1||+1)
. . . . 8. ∀j:ℕ(||L1||+1). [x1 / L1][j] = L2[(f(j))]
. . . . 9. j : ℕ(||L1||+1)
. . . . 10. T
. . . . ⊢  j<||[x1 / L1]||  by Reduce 0 THEN Auto'
END
```