```
range_sublist  THM

⊢  ∀T:Type, L:T List, n:ℕ, f:(ℕn→ℕ||L||).
|  increasing(f;n)
|  ⇒
|  (∃L1:T List. ||L1|| = n & sublist_occurence(T;L1;L;f))
| by Unfold ‘sublist_occurence‘ 0 THEN InductionOnList THEN Auto
|
|\
| 1. T : Type
| 2. T List
| 3. n : ℕ
| 4. f : ℕn→ℕ||nil||
| 5. increasing(f;n)
| ⊢  ∃L1:T List.
| |  ||L1|| = n
| |  & increasing(f;||L1||) & (∀j:ℕ||L1||. L1[j] = nil[(f(j))])
| | by Subst n ∼ 0 0
| |
| |\
| | .....equality .....
| | 1. T : Type
| | 2. T List
| | 3. n : ℕ
| | 4. f : ℕn→ℕ||nil||
| | 5. increasing(f;n)
| | ⊢  n ∼ 0  by Assert n = 0
| | |
| | |\
| | | .....assertion .....
| | | 1. T : Type
| | | 2. T List
| | | 3. n : ℕ
| | | 4. f : ℕn→ℕ||nil||
| | | 5. increasing(f;n)
| | | ⊢  n = 0  by SupposeNot THEN Assert f(0) ∈ ℕ||nil|| THEN Auto
| | | |
| | | 1. T : Type
| | | 2. T List
| | | 3. n : ℕ
| | | 4. f : ℕn→ℕ||nil||
| | | 5. increasing(f;n)
| | | 6. ¬n = 0
| | | 7. f(0) ∈ ℕ||nil||
| | | ⊢  n = 0  by All Reduce THEN Auto’
| | |
| | .\
| | . 1. T : Type
| | . 2. T List
| | . 3. n : ℕ
| | . 4. f : ℕn→ℕ||nil||
| | . 5. increasing(f;n)
| | . 6. n = 0
| | . ⊢  n ∼ 0  by Auto
| |
```

```
| .\
| . 1. T : Type
| . 2. T List
| . 3. n : ℕ
| . 4. f : ℕn→ℕ||nil||
| . 5. increasing(f;n)
| . ⊢ ∃L1:T List.
| . |  ||L1|| = 0
| . |  & increasing(f;||L1||) & (∀j:ℕ||L1||. L1[j] = nil[(f(j))])
| . | by With nil (D 0) THEN All Reduce THEN Auto
| . |
| . 1. T : Type
| . 2. T List
| . 3. n : ℕ
| . 4. f : ℕn→ℕ0
| . 5. increasing(f;n)
| . ⊢ increasing(f;0)  by Unfold ‘increasing‘ 0 THEN Auto’
|
.\
. 1. T : Type
. 2. T List
. 3. u : T
. 4. v : T List
. 5. ∀n:ℕ, f:(ℕn→ℕ||v||).
. |  increasing(f;n)
. |  ⇒
. |  (∃L1:T List.
. |   ||L1|| = n
. |   & increasing(f;||L1||) & (∀j:ℕ||L1||. L1[j] = v[(f(j))]))
. 6. n : ℕ
. 7. f : ℕn→ℕ||[u / v]||
. 8. increasing(f;n)
. ⊢ ∃L1:T List.
. |  ||L1|| = n
. |  & increasing(f;||L1||) & (∀j:ℕ||L1||. L1[j] = [u / v][(f(j))])
. | by CaseNat 0 ‘n’
. |
. |\
. | 1. T : Type
. | 2. T List
. | 3. u : T
. | 4. v : T List
. | 5. ∀n:ℕ, f:(ℕn→ℕ||v||).
. | |  increasing(f;n)
. | |  ⇒
. | |  (∃L1:T List.
. | |   ||L1|| = n
. | |   & increasing(f;||L1||) & (∀j:ℕ||L1||. L1[j] = v[(f(j))]))
. | 6. n : ℕ
. | 7. f : ℕn→ℕ||[u / v]||
. | 8. increasing(f;n)
. | 9. n = 0
. | ⊢ ∃L1:T List.
. | |  ||L1|| = 0
. | |  & increasing(f;||L1||) & (∀j:ℕ||L1||. L1[j] = [u / v][(f(j))])
```

```
. | | by HypSubstSq (-1) 0 THEN InstConcl [nil
. | |    ] THEN Reduce 0 THEN Auto'
. | |
. | 1. T : Type
. | 2. T List
. | 3. u : T
. | 4. v : T List
. | 5. ∀n:ℕ, f:(ℕn→ℕ||v||).
. |    increasing(f;n)
. |    ⇒
. |    (∃L1:T List.
. |     ||L1|| = n
. |     & increasing(f;||L1||) & (∀j:ℕ||L1||. L1[j] = v[(f(j))]))
. | 6. n : ℕ
. | 7. f : ℕn→ℕ||[u / v]||
. | 8. increasing(f;n)
. | 9. n = 0
. | ⊢  increasing(f;0)  by Unfold `increasing` 0 THEN Auto'
. |
. .\
. . 1. T : Type
. . 2. T List
. . 3. u : T
. . 4. v : T List
. . 5. ∀n:ℕ, f:(ℕn→ℕ||v||).
. . |  increasing(f;n)
. . |  ⇒
. . |  (∃L1:T List.
. . |   ||L1|| = n
. . |   & increasing(f;||L1||) & (∀j:ℕ||L1||. L1[j] = v[(f(j))]))
. . 6. n : ℕ
. . 7. f : ℕn→ℕ||[u / v]||
. . 8. increasing(f;n)
. . 9. ¬n = 0
. . ⊢  ∃L1:T List.
. . |  ||L1|| = n
. . |  & increasing(f;||L1||) & (∀j:ℕ||L1||. L1[j] = [u / v][(f(j))])
. . | by Decide f(0) = 0 THENA Auto
. . |
. . |\
. . | 1. T : Type
. . | 2. T List
. . | 3. u : T
. . | 4. v : T List
. . | 5. ∀n:ℕ, f:(ℕn→ℕ||v||).
. . | |  increasing(f;n)
. . | |  ⇒
. . | |  (∃L1:T List.
. . | |   ||L1|| = n
. . | |   & increasing(f;||L1||) & (∀j:ℕ||L1||. L1[j] = v[(f(j))]))
. . | 6. n : ℕ
. . | 7. f : ℕn→ℕ||[u / v]||
. . | 8. increasing(f;n)
. . | 9. ¬n = 0
. . | 10. f(0) = 0
```

```
. . | ⊢  ∃L1:T List.
. . | |  ||L1|| = n
. . | |  & increasing(f;||L1||)
. . | |    & (∀j:ℕ||L1||. L1[j] = [u / v][(f(j))])
. . | | by InstHyp [n-1;λi.f(i+1)-1] (-6) THENA Auto'
. . | |
. . | |\
. . | | .....subterm .....
. . | | 1. T : Type
. . | | 2. T List
. . | | 3. u : T
. . | | 4. v : T List
. . | | 5. ∀n:ℕ, f:(ℕn→ℕ||v||).
. . | |    increasing(f;n)
. . | |    ⇒
. . | |    (∃L1:T List.
. . | |     ||L1|| = n
. . | |     & increasing(f;||L1||) & (∀j:ℕ||L1||. L1[j] = v[(f(j))]))
. . | | 6. n : ℕ
. . | | 7. f : ℕn→ℕ||[u / v]||
. . | | 8. increasing(f;n)
. . | | 9. ¬n = 0
. . | | 10. f(0) = 0
. . | | 11. i : ℕ(n-1)
. . | | ⊢  f(i+1)-1 ∈ ℕ||v||
. . | |   by InstLemma `increasing_lower_bound` [n;f;i+1] THEN Auto'
. . | |
. . | |\
. . | | .....antecedent .....
. . | | 1. T : Type
. . | | 2. T List
. . | | 3. u : T
. . | | 4. v : T List
. . | | 5. ∀n:ℕ, f:(ℕn→ℕ||v||).
. . | |    increasing(f;n)
. . | |    ⇒
. . | |    (∃L1:T List.
. . | |     ||L1|| = n
. . | |     & increasing(f;||L1||) & (∀j:ℕ||L1||. L1[j] = v[(f(j))]))
. . | | 6. n : ℕ
. . | | 7. f : ℕn→ℕ||[u / v]||
. . | | 8. increasing(f;n)
. . | | 9. ¬n = 0
. . | | 10. f(0) = 0
. . | | ⊢  increasing(λi.f(i+1)-1;n-1)
. . | |   by ParallelOp (-3) THEN Reduce 0 THEN D 0 THENA Auto
. . | |
. . | |     THEN InstHyp [i+1] (-4) THEN Auto'
. . | |
. . | .\
. . | . 1. T : Type
. . | . 2. T List
. . | . 3. u : T
. . | . 4. v : T List
. . | . 5. ∀n:ℕ, f:(ℕn→ℕ||v||).
```

```
. . | . |  increasing(f;n)
. . | . |  ⇒
. . | . |  (∃L1:T List.
. . | . |   ||L1|| = n
. . | . |   & increasing(f;||L1||) & (∀j:ℕ||L1||. L1[j] = v[(f(j))]))
. . | . 6. n : ℕ
. . | . 7. f : ℕn→ℕ||[u / v]||
. . | . 8. increasing(f;n)
. . | . 9. ¬n = 0
. . | . 10. f(0) = 0
. . | . 11. ∃L1:T List.
. . | . |   ||L1|| = n-1
. . | . |   & increasing(λi.f(i+1)-1;||L1||)
. . | . |     & (∀j:ℕ||L1||. L1[j] = v[((λi.f(i+1)-1)(j))])
. . | . ⊢  ∃L1:T List.
. . | . |  ||L1|| = n
. . | . |  & increasing(f;||L1||)
. . | . |    & (∀j:ℕ||L1||. L1[j] = [u / v][(f(j))])
. . | . | by ExRepD THEN InstConcl [[u / L1]] THEN Reduce 0 THEN Auto'
. . | . |
. . | . |
. . | . |\
. . | . | 1. T : Type
. . | . | 2. T List
. . | . | 3. u : T
. . | . | 4. v : T List
. . | . | 5. ∀n:ℕ, f:(ℕn→ℕ||v||).
. . | . | |  increasing(f;n)
. . | . | |  ⇒
. . | . | |  (∃L1:T List.
. . | . | |   ||L1|| = n
. . | . | |   & increasing(f;||L1||)
. . | . | |     & (∀j:ℕ||L1||. L1[j] = v[(f(j))]))
. . | . | 6. n : ℕ
. . | . | 7. f : ℕn→ℕ||[u / v]||
. . | . | 8. increasing(f;n)
. . | . | 9. ¬n = 0
. . | . | 10. f(0) = 0
. . | . | 11. L1 : T List
. . | . | 12. ||L1|| = n-1
. . | . | 13. increasing(λi.f(i+1)-1;||L1||)
. . | . | 14. ∀j:ℕ||L1||. L1[j] = v[((λi.f(i+1)-1)(j))]
. . | . | ⊢  increasing(f;||L1||+1)  by HypSubstSq (-3) 0
. . | . | |
. . | . | 1. T : Type
. . | . | 2. T List
. . | . | 3. u : T
. . | . | 4. v : T List
. . | . | 5. ∀n:ℕ, f:(ℕn→ℕ||v||).
. . | . |    increasing(f;n)
. . | . |    ⇒
. . | . |    (∃L1:T List.
. . | . |     ||L1|| = n
. . | . |     & increasing(f;||L1||)
. . | . |       & (∀j:ℕ||L1||. L1[j] = v[(f(j))]))
```

```
. . | . | 6. n : ℕ
. . | . | 7. f : ℕn→ℕ||[u / v]||
. . | . | 8. increasing(f;n)
. . | . | 9. ¬n = 0
. . | . | 10. f(0) = 0
. . | . | 11. L1 : T List
. . | . | 12. ||L1|| = n-1
. . | . | 13. increasing(λi.f(i+1)-1;||L1||)
. . | . | 14. ∀j:ℕ||L1||. L1[j] = v[((λi.f(i+1)-1)(j))]
. . | . | ⊢  increasing(f;n-1+1)  by Subst (n-1+1) ∼ n 0 THEN Auto'
. . | . |
. . | . .\
. . | . . 1. T : Type
. . | . . 2. T List
. . | . . 3. u : T
. . | . . 4. v : T List
. . | . . 5. ∀n:ℕ, f:(ℕn→ℕ||v||).
. . | . . |  increasing(f;n)
. . | . . |  ⇒
. . | . . |  (∃L1:T List.
. . | . . |   ||L1|| = n
. . | . . |   & increasing(f;||L1||)
. . | . . |     & (∀j:ℕ||L1||. L1[j] = v[(f(j))]))
. . | . . 6. n : ℕ
. . | . . 7. f : ℕn→ℕ||[u / v]||
. . | . . 8. increasing(f;n)
. . | . . 9. ¬n = 0
. . | . . 10. f(0) = 0
. . | . . 11. L1 : T List
. . | . . 12. ||L1|| = n-1
. . | . . 13. increasing(λi.f(i+1)-1;||L1||)
. . | . . 14. ∀j:ℕ||L1||. L1[j] = v[((λi.f(i+1)-1)(j))]
. . | . . 15. j : ℕ(||L1||+1)
. . | . . ⊢  [u / L1][j] = [u / v][(f(j))]
. . | . . | by Decide j = 0 THENA Auto
. . | . . |
. . | . . |\
. . | . . | 1. T : Type
. . | . . | 2. T List
. . | . . | 3. u : T
. . | . . | 4. v : T List
. . | . . | 5. ∀n:ℕ, f:(ℕn→ℕ||v||).
. . | . . |    increasing(f;n)
. . | . . |    ⇒
. . | . . |    (∃L1:T List.
. . | . . |     ||L1|| = n
. . | . . |     & increasing(f;||L1||)
. . | . . |       & (∀j:ℕ||L1||. L1[j] = v[(f(j))]))
. . | . . | 6. n : ℕ
. . | . . | 7. f : ℕn→ℕ||[u / v]||
. . | . . | 8. increasing(f;n)
. . | . . | 9. ¬n = 0
. . | . . | 10. f(0) = 0
. . | . . | 11. L1 : T List
. . | . . | 12. ||L1|| = n-1
```

```
. . | . . | 13. increasing(λi.f(i+1)-1;||L1||)
. . | . . | 14. ∀j:ℕ||L1||. L1[j] = v[((λi.f(i+1)-1)(j))]
. . | . . | 15. j : ℕ(||L1||+1)
. . | . . | 16. j = 0
. . | . . | ⊢  [u / L1][j] = [u / v][(f(j))]
. . | . . |   by ...
. . | . . |
. . | . . .\
. . | . . . 1. T : Type
. . | . . . 2. T List
. . | . . . 3. u : T
. . | . . . 4. v : T List
. . | . . . 5. ∀n:ℕ, f:(ℕn→ℕ||v||).
. . | . . . |  increasing(f;n)
. . | . . . |  ⇒
. . | . . . |  (∃L1:T List.
. . | . . . |   ||L1|| = n
. . | . . . |   & increasing(f;||L1||)
. . | . . . |     & (∀j:ℕ||L1||. L1[j] = v[(f(j))]))
. . | . . . 6. n : ℕ
. . | . . . 7. f : ℕn→ℕ||[u / v]||
. . | . . . 8. increasing(f;n)
. . | . . . 9. ¬n = 0
. . | . . . 10. f(0) = 0
. . | . . . 11. L1 : T List
. . | . . . 12. ||L1|| = n-1
. . | . . . 13. increasing(λi.f(i+1)-1;||L1||)
. . | . . . 14. ∀j:ℕ||L1||. L1[j] = v[((λi.f(i+1)-1)(j))]
. . | . . . 15. j : ℕ(||L1||+1)
. . | . . . 16. ¬j = 0
. . | . . . ⊢  [u / L1][j] = [u / v][(f(j))]
. . | . . . | by RWO "select_cons_tl" 0 THEN Auto'
. . | . . . |
. . | . . . |\
. . | . . . | .....rewrite subgoal .....
. . | . . . | 1. T : Type
. . | . . . | 2. T List
. . | . . . | 3. u : T
. . | . . . | 4. v : T List
. . | . . . | 5. ∀n:ℕ, f:(ℕn→ℕ||v||).
. . | . . . |    increasing(f;n)
. . | . . . |    ⇒
. . | . . . |    (∃L1:T List.
. . | . . . |     ||L1|| = n
. . | . . . |     & increasing(f;||L1||)
. . | . . . |       & (∀j:ℕ||L1||. L1[j] = v[(f(j))]))
. . | . . . | 6. n : ℕ
. . | . . . | 7. f : ℕn→ℕ||[u / v]||
. . | . . . | 8. increasing(f;n)
. . | . . . | 9. ¬n = 0
. . | . . . | 10. f(0) = 0
. . | . . . | 11. L1 : T List
. . | . . . | 12. ||L1|| = n-1
. . | . . . | 13. increasing(λi.f(i+1)-1;||L1||)
. . | . . . | 14. ∀j:ℕ||L1||. L1[j] = v[((λi.f(i+1)-1)(j))]
```

```
. . | . . . | 15. j : ℕ(||L1||+1)
. . | . . . | 16. ¬j = 0
. . | . . . | ⊢  0<f(j)
. . | . . . |   by InstLemma ‘increasing_lower_bound‘ [n;f;j
. . | . . . |      ] THEN Auto’
. . | . . . |
. . | . . . .\
. . | . . . . 1. T : Type
. . | . . . . 2. T List
. . | . . . . 3. u : T
. . | . . . . 4. v : T List
. . | . . . . 5. ∀n:ℕ, f:(ℕn→ℕ||v||).
. . | . . . . |  increasing(f;n)
. . | . . . . |  ⇒
. . | . . . . |  (∃L1:T List.
. . | . . . . |   ||L1|| = n
. . | . . . . |   & increasing(f;||L1||)
. . | . . . . |     & (∀j:ℕ||L1||. L1[j] = v[(f(j))]))
. . | . . . . 6. n : ℕ
. . | . . . . 7. f : ℕn→ℕ||[u / v]||
. . | . . . . 8. increasing(f;n)
. . | . . . . 9. ¬n = 0
. . | . . . . 10. f(0) = 0
. . | . . . . 11. L1 : T List
. . | . . . . 12. ||L1|| = n-1
. . | . . . . 13. increasing(λi.f(i+1)-1;||L1||)
. . | . . . . 14. ∀j:ℕ||L1||. L1[j] = v[((λi.f(i+1)-1)(j))]
. . | . . . . 15. j : ℕ(||L1||+1)
. . | . . . . 16. ¬j = 0
. . | . . . . ⊢  L1[(j-1)] = v[(f(j)-1)]
. . | . . . . | by InstHyp [j-1] (-3) THEN Auto
. . | . . . . |
. . | . . . . 1. T : Type
. . | . . . . 2. T List
. . | . . . . 3. u : T
. . | . . . . 4. v : T List
. . | . . . . 5. ∀n:ℕ, f:(ℕn→ℕ||v||).
. . | . . . . |  increasing(f;n)
. . | . . . . |  ⇒
. . | . . . . |  (∃L1:T List.
. . | . . . . |   ||L1|| = n
. . | . . . . |   & increasing(f;||L1||)
. . | . . . . |     & (∀j:ℕ||L1||. L1[j] = v[(f(j))]))
. . | . . . . 6. n : ℕ
. . | . . . . 7. f : ℕn→ℕ||[u / v]||
. . | . . . . 8. increasing(f;n)
. . | . . . . 9. ¬n = 0
. . | . . . . 10. f(0) = 0
. . | . . . . 11. L1 : T List
. . | . . . . 12. ||L1|| = n-1
. . | . . . . 13. increasing(λi.f(i+1)-1;||L1||)
. . | . . . . 14. ∀j:ℕ||L1||. L1[j] = v[((λi.f(i+1)-1)(j))]
. . | . . . . 15. j : ℕ(||L1||+1)
. . | . . . . 16. ¬j = 0
. . | . . . . 17. L1[(j-1)] = v[((λi.f(i+1)-1)(j-1))]
```

```
. . | . . . . ⊢  L1[(j-1)] = v[(f(j)-1)]  by Reduce (-1)
. . | . . . . |
. . | . . . . 1. T : Type
. . | . . . . 2. T List
. . | . . . . 3. u : T
. . | . . . . 4. v : T List
. . | . . . . 5. ∀n:ℕ, f:(ℕn→ℕ||v||).
. . | . . . .    increasing(f;n)
. . | . . . .    ⇒
. . | . . . .    (∃L1:T List.
. . | . . . .     ||L1|| = n
. . | . . . .     & increasing(f;||L1||)
. . | . . . .       & (∀j:ℕ||L1||. L1[j] = v[(f(j))]))
. . | . . . . 6. n : ℕ
. . | . . . . 7. f : ℕn→ℕ||[u / v]||
. . | . . . . 8. increasing(f;n)
. . | . . . . 9. ¬n = 0
. . | . . . . 10. f(0) = 0
. . | . . . . 11. L1 : T List
. . | . . . . 12. ||L1|| = n-1
. . | . . . . 13. increasing(λi.f(i+1)-1;||L1||)
. . | . . . . 14. ∀j:ℕ||L1||. L1[j] = v[((λi.f(i+1)-1)(j))]
. . | . . . . 15. j : ℕ(||L1||+1)
. . | . . . . 16. ¬j = 0
. . | . . . . 17. L1[(j-1)] = v[(f(j-1+1)-1)]
. . | . . . . ⊢  L1[(j-1)] = v[(f(j)-1)]
. . | . . . .   by Subst (j-1+1) ∼ j (-1) THEN Auto'
. . |
. . .\
. . . 1. T : Type
. . . 2. T List
. . . 3. u : T
. . . 4. v : T List
. . . 5. ∀n:ℕ, f:(ℕn→ℕ||v||).
. . . |  increasing(f;n)
. . . |  ⇒
. . . |  (∃L1:T List.
. . . |   ||L1|| = n
. . . |   & increasing(f;||L1||) & (∀j:ℕ||L1||. L1[j] = v[(f(j))]))
. . . 6. n : ℕ
. . . 7. f : ℕn→ℕ||[u / v]||
. . . 8. increasing(f;n)
. . . 9. ¬n = 0
. . . 10. ¬f(0) = 0
. . . ⊢  ∃L1:T List.
. . . |  ||L1|| = n
. . . |  & increasing(f;||L1||)
. . . |    & (∀j:ℕ||L1||. L1[j] = [u / v][(f(j))])
. . . | by InstHyp [n;fadd(f;λi.0-1)] (-6) THENA Auto
. . . |
. . . |\
. . . | .....wf .....
. . . | 1. T : Type
. . . | 2. T List
. . . | 3. u : T
```

```
. . . | 4. v : T List
. . . | 5. ∀n:ℕ, f:(ℕn→ℕ||v||).
. . . | |  increasing(f;n)
. . . | |  ⇒
. . . | |  (∃L1:T List.
. . . | |   ||L1|| = n
. . . | |   & increasing(f;||L1||) & (∀j:ℕ||L1||. L1[j] = v[(f(j))]))
. . . | 6. n : ℕ
. . . | 7. f : ℕn→ℕ||[u / v]||
. . . | 8. increasing(f;n)
. . . | 9. ¬n = 0
. . . | 10. ¬f(0) = 0
. . . | ⊢  fadd(f;λi.0-1) ∈ ℕn→ℕ||v||
. . . | | by Unfold `fadd` 0 THEN Auto THEN Reduce 0 THEN Auto'
. . . | |
. . . | 1. T : Type
. . . | 2. T List
. . . | 3. u : T
. . . | 4. v : T List
. . . | 5. ∀n:ℕ, f:(ℕn→ℕ||v||).
. . . |    increasing(f;n)
. . . |    ⇒
. . . |    (∃L1:T List.
. . . |     ||L1|| = n
. . . |     & increasing(f;||L1||) & (∀j:ℕ||L1||. L1[j] = v[(f(j))]))
. . . | 6. n : ℕ
. . . | 7. f : ℕn→ℕ||[u / v]||
. . . | 8. increasing(f;n)
. . . | 9. ¬n = 0
. . . | 10. ¬f(0) = 0
. . . | 11. i : ℕn
. . . | ⊢  0≤f(i)+-1
. . . |   by InstLemma `increasing_lower_bound` [n;f;i] THEN Auto'
. . . |
. . . |\
. . . | .....antecedent .....
. . . | 1. T : Type
. . . | 2. T List
. . . | 3. u : T
. . . | 4. v : T List
. . . | 5. ∀n:ℕ, f:(ℕn→ℕ||v||).
. . . |    increasing(f;n)
. . . |    ⇒
. . . |    (∃L1:T List.
. . . |     ||L1|| = n
. . . |     & increasing(f;||L1||) & (∀j:ℕ||L1||. L1[j] = v[(f(j))]))
. . . | 6. n : ℕ
. . . | 7. f : ℕn→ℕ||[u / v]||
. . . | 8. increasing(f;n)
. . . | 9. ¬n = 0
. . . | 10. ¬f(0) = 0
. . . | ⊢  increasing(fadd(f;λi.0-1);n)
. . . |   by BackThruLemma `fadd_increasing` THEN Auto THEN Easy
. . . |
. . . .\
```

```
. . . . 1. T : Type
. . . . 2. T List
. . . . 3. u : T
. . . . 4. v : T List
. . . . 5. ∀n:ℕ, f:(ℕn→ℕ||v||).
. . . . |  increasing(f;n)
. . . . |  ⇒
. . . . |  (∃L1:T List.
. . . . |   ||L1|| = n
. . . . |   & increasing(f;||L1||) & (∀j:ℕ||L1||. L1[j] = v[(f(j))]))
. . . . 6. n : ℕ
. . . . 7. f : ℕn→ℕ||[u / v]||
. . . . 8. increasing(f;n)
. . . . 9. ¬n = 0
. . . . 10. ¬f(0) = 0
. . . . 11. ∃L1:T List.
. . . . |   ||L1|| = n
. . . . |   & increasing(fadd(f;λi.0-1);||L1||)
. . . . |     & (∀j:ℕ||L1||. L1[j] = v[(fadd(f;λi.0-1)(j))])
. . . . ⊢  ∃L1:T List.
. . . . |  ||L1|| = n
. . . . |  & increasing(f;||L1||)
. . . . |    & (∀j:ℕ||L1||. L1[j] = [u / v][(f(j))])
. . . . | by ExRepD THEN InstConcl [L1] THENA Auto'
. . . . |
. . . . 1. T : Type
. . . . 2. T List
. . . . 3. u : T
. . . . 4. v : T List
. . . . 5. ∀n:ℕ, f:(ℕn→ℕ||v||).
. . . . |  increasing(f;n)
. . . . |  ⇒
. . . . |  (∃L1:T List.
. . . . |   ||L1|| = n
. . . . |   & increasing(f;||L1||) & (∀j:ℕ||L1||. L1[j] = v[(f(j))]))
. . . . 6. n : ℕ
. . . . 7. f : ℕn→ℕ||[u / v]||
. . . . 8. increasing(f;n)
. . . . 9. ¬n = 0
. . . . 10. ¬f(0) = 0
. . . . 11. L1 : T List
. . . . 12. ||L1|| = n
. . . . 13. increasing(fadd(f;λi.0-1);||L1||)
. . . . 14. ∀j:ℕ||L1||. L1[j] = v[(fadd(f;λi.0-1)(j))]
. . . . ⊢  ||L1|| = n
. . . . |  & increasing(f;||L1||)
. . . . |    & (∀j:ℕ||L1||. L1[j] = [u / v][(f(j))])  by Auto
. . . . |
. . . . |\
. . . . | 1. T : Type
. . . . | 2. T List
. . . . | 3. u : T
. . . . | 4. v : T List
. . . . | 5. ∀n:ℕ, f:(ℕn→ℕ||v||).
. . . . |    increasing(f;n)
```

```
. . . . |    ⇒
. . . . |    (∃L1:T List.
. . . . |     ||L1|| = n
. . . . |     & increasing(f;||L1||)
. . . . |       & (∀j:ℕ||L1||. L1[j] = v[(f(j))]))
. . . . | 6. n : ℕ
. . . . | 7. f : ℕn→ℕ||[u / v]||
. . . . | 8. increasing(f;n)
. . . . | 9. ¬n = 0
. . . . | 10. ¬f(0) = 0
. . . . | 11. L1 : T List
. . . . | 12. ||L1|| = n
. . . . | 13. increasing(fadd(f;λi.0-1);||L1||)
. . . . | 14. ∀j:ℕ||L1||. L1[j] = v[(fadd(f;λi.0-1)(j))]
. . . . | ⊢  increasing(f;||L1||)
. . . . |   by ...
. . . . | Auto
. . . . |
. . . . .\
. . . . . 1. T : Type
. . . . . 2. T List
. . . . . 3. u : T
. . . . . 4. v : T List
. . . . . 5. ∀n:ℕ, f:(ℕn→ℕ||v||).
. . . . . |  increasing(f;n)
. . . . . |  ⇒
. . . . . |  (∃L1:T List.
. . . . . |   ||L1|| = n
. . . . . |   & increasing(f;||L1||)
. . . . . |     & (∀j:ℕ||L1||. L1[j] = v[(f(j))]))
. . . . . 6. n : ℕ
. . . . . 7. f : ℕn→ℕ||[u / v]||
. . . . . 8. increasing(f;n)
. . . . . 9. ¬n = 0
. . . . . 10. ¬f(0) = 0
. . . . . 11. L1 : T List
. . . . . 12. ||L1|| = n
. . . . . 13. increasing(fadd(f;λi.0-1);||L1||)
. . . . . 14. ∀j:ℕ||L1||. L1[j] = v[(fadd(f;λi.0-1)(j))]
. . . . . 15. j : ℕ||L1||
. . . . . ⊢  L1[j] = [u / v][(f(j))]
. . . . . | by Unfold ‘fadd‘ (-2) THEN Reduce (-2) THEN InstHyp [j
. . . . . |    ] (-2) THENA Auto
. . . . . |
. . . . . 1. T : Type
. . . . . 2. T List
. . . . . 3. u : T
. . . . . 4. v : T List
. . . . . 5. ∀n:ℕ, f:(ℕn→ℕ||v||).
. . . . . |  increasing(f;n)
. . . . . |  ⇒
. . . . . |  (∃L1:T List.
. . . . . |   ||L1|| = n
. . . . . |   & increasing(f;||L1||)
. . . . . |     & (∀j:ℕ||L1||. L1[j] = v[(f(j))]))
```

```
. . . . . 6. n : ℕ
. . . . . 7. f : ℕn→ℕ||[u / v]||
. . . . . 8. increasing(f;n)
. . . . . 9. ¬n = 0
. . . . . 10. ¬f(0) = 0
. . . . . 11. L1 : T List
. . . . . 12. ||L1|| = n
. . . . . 13. increasing(fadd(f;λi.0-1);||L1||)
. . . . . 14. ∀j:ℕ||L1||. L1[j] = v[(f(j)+-1)]
. . . . . 15. j : ℕ||L1||
. . . . . 16. L1[j] = v[(f(j)+-1)]
. . . . . ⊢ L1[j] = [u / v][(f(j))] by HypSubst (-1) 0 THENA Auto'
. . . . . |
. . . . . 1. T : Type
. . . . . 2. T List
. . . . . 3. u : T
. . . . . 4. v : T List
. . . . . 5. ∀n:ℕ, f:(ℕn→ℕ||v||).
. . . . . | increasing(f;n)
. . . . . | ⇒
. . . . . | (∃L1:T List.
. . . . . | ||L1|| = n
. . . . . | & increasing(f;||L1||)
. . . . . | & (∀j:ℕ||L1||. L1[j] = v[(f(j))]))
. . . . . 6. n : ℕ
. . . . . 7. f : ℕn→ℕ||[u / v]||
. . . . . 8. increasing(f;n)
. . . . . 9. ¬n = 0
. . . . . 10. ¬f(0) = 0
. . . . . 11. L1 : T List
. . . . . 12. ||L1|| = n
. . . . . 13. increasing(fadd(f;λi.0-1);||L1||)
. . . . . 14. ∀j:ℕ||L1||. L1[j] = v[(f(j)+-1)]
. . . . . 15. j : ℕ||L1||
. . . . . 16. L1[j] = v[(f(j)+-1)]
. . . . . ⊢ v[(f(j)+-1)] = [u / v][(f(j))] by Assert 0<f(j)
. . . . . |
. . . . . |\
. . . . . | .....assertion .....
. . . . . | 1. T : Type
. . . . . | 2. T List
. . . . . | 3. u : T
. . . . . | 4. v : T List
. . . . . | 5. ∀n:ℕ, f:(ℕn→ℕ||v||).
. . . . . | increasing(f;n)
. . . . . | ⇒
. . . . . | (∃L1:T List.
. . . . . | ||L1|| = n
. . . . . | & increasing(f;||L1||)
. . . . . | & (∀j:ℕ||L1||. L1[j] = v[(f(j))]))
. . . . . | 6. n : ℕ
. . . . . | 7. f : ℕn→ℕ||[u / v]||
. . . . . | 8. increasing(f;n)
. . . . . | 9. ¬n = 0
. . . . . | 10. ¬f(0) = 0
```

```
. . . . . | 11. L1 : T List
. . . . . | 12. ||L1|| = n
. . . . . | 13. increasing(fadd(f;λi.0-1);||L1||)
. . . . . | 14. ∀j:ℕ||L1||. L1[j] = v[(f(j)+-1)]
. . . . . | 15. j : ℕ||L1||
. . . . . | 16. L1[j] = v[(f(j)+-1)]
. . . . . | ⊢  0<f(j)
. . . . . |   by InstLemma ‘increasing_lower_bound‘ [n;f;j] THEN Auto’
. . . . . |
. . . . . |
. . . . . |
. . . . . .\
. . . . . . 1. T : Type
. . . . . . 2. T List
. . . . . . 3. u : T
. . . . . . 4. v : T List
. . . . . . 5. ∀n:ℕ, f:(ℕn→ℕ||v||).
. . . . . .    increasing(f;n)
. . . . . .    ⇒
. . . . . .    (∃L1:T List.
. . . . . .     ||L1|| = n
. . . . . .     & increasing(f;||L1||)
. . . . . .       & (∀j:ℕ||L1||. L1[j] = v[(f(j))]))
. . . . . . 6. n : ℕ
. . . . . . 7. f : ℕn→ℕ||[u / v]||
. . . . . . 8. increasing(f;n)
. . . . . . 9. ¬n = 0
. . . . . . 10. ¬f(0) = 0
. . . . . . 11. L1 : T List
. . . . . . 12. ||L1|| = n
. . . . . . 13. increasing(fadd(f;λi.0-1);||L1||)
. . . . . . 14. ∀j:ℕ||L1||. L1[j] = v[(f(j)+-1)]
. . . . . . 15. j : ℕ||L1||
. . . . . . 16. L1[j] = v[(f(j)+-1)]
. . . . . . 17. 0<f(j)
. . . . . . ⊢  v[(f(j)+-1)] = [u / v][(f(j))]
. . . . . .   by RW0 "select_cons_tl" 0 THEN Auto’
END
```