```
⊢ ∀n:ℕ+. ∀x:ℕ.  (∃r:{ℕ| ((r^n ≤ x) ∧ x < r + 1^n)})
|
BY ((D 0 THENA Auto)
|   THEN (Evaluate ⌈b = 2^n⌉·
|         THENA (Auto THEN InstLemma ‘exp_preserves_lt‘ [⌈n⌉;⌈1⌉;⌈2⌉]· THEN Auto)
|         )
|   THEN InstLemma ‘div_nat_induction‘ [⌈b⌉;⌈λ2x.∃r:{ℕ| ((r^n ≤ x) ∧ x < r + 1^n)}⌉]·
|   THEN Auto
|   THEN Try ((RWO "exp-fastexp<" 0 THEN Auto)·))
|\
| 1. n: ℕ+
| 2. b: {b:ℤ| 1 < b}
| 3. b = 2^n
| ⊢ ∃r:{ℕ| ((r^n ≤ 0) ∧ 0 < r + 1^n)}
| |
1 BY With ⌈0⌉ (D 0)·
|    THEN (Reduce 0 THEN Auto THEN RWO  "exp-zero exp-one" 0 THEN Auto)·
 \
  1. n: ℕ+
  2. b: {b:ℤ| 1 < b}
  3. b = 2^n
  4. i: ℕ+
  5. ∃r:{ℕ| ((r^n ≤ (i ÷ b)) ∧ i ÷ b < r + 1^n)}
  ⊢ ∃r:{ℕ| ((r^n ≤ i) ∧ i < r + 1^n)}
  |
  BY (D -1
  |   THEN (Evaluate ⌈r2 = (2 * r)⌉· THENA Auto)
  |   THEN (Evaluate ⌈r2’ = (r2 + 1)⌉· THENA Auto)
  |   THEN (InstLemma ‘exp-of-mul‘ [⌈2⌉;⌈r⌉;⌈n⌉]· THENA Auto)
  |   THEN (InstLemma ‘exp-of-mul‘ [⌈2⌉;⌈r + 1⌉;⌈n⌉]· THENA Auto)
  |   THEN ((InstLemma ‘div_rem_sum‘ [⌈i⌉; ⌈2^n⌉])· THENA Auto)
  |   THEN (InstLemma ‘rem_bounds_1‘ [⌈i⌉; ⌈2^n⌉])·
  |   THEN Auto
  |   THEN ((Decide ⌈r2’^n < i + 1⌉· THENA Auto) THEN All (RWO "exp-fastexp<") THEN Auto))
  |\
  | 5. r: ℕ
  | [6]. (r^n ≤ (i ÷ b)) ∧ i ÷ b < r + 1^n
  | 7. r2: ℤ
  | 8. r2 = (2 * r)
  | 9. r2’: ℤ
  | 10. r2’ = (r2 + 1)
  | 11. 2 * r^n = (2^n * r^n)
  | 12. 2 * (r + 1)^n = (2^n * r + 1^n)
  | 13. i = (((i ÷ 2^n) * 2^n) + (i rem 2^n))
  | 14. 0 ≤ (i rem 2^n)
  | 15. i rem 2^n < 2^n
  | 16. r2’^n < i + 1
  | ⊢ ∃r:{ℕ| ((r^n ≤ i) ∧ i < r + 1^n)}
  | |
  1 BY (With ⌈r2’⌉ (D 0)· THEN Auto’ THEN ElimVar ‘r2\’’ THEN ElimVar ‘r2’)
  | |
  | 6. r^n ≤ (i ÷ b)
  | 7. i ÷ b < r + 1^n
  | 8. r2: ℤ
  | 9. 2 * r ∈ ℤ
```

```
| 10. r2': ℤ
| 11. (2 * r) + 1 ∈ ℤ
| 12. 2 * r^n = (2^n * r^n)
| 13. 2 * (r + 1)^n = (2^n * r + 1^n)
| 14. i = (((i ÷ 2^n) * 2^n) + (i rem 2^n))
| 15. 0 ≤ (i rem 2^n)
| 16. i rem 2^n < 2^n
| 17. (2 * r) + 1^n < i + 1
| 18. (2 * r) + 1^n ≤ i
| ⊢ i < ((2 * r) + 1) + 1^n
| |
1 BY (Subst' ((2 * r) + 1) + 1 ~ 2 * (r + 1) 0 THEN Auto THEN HypSubst' (-5) 0)
| |
| ⊢ ((i ÷ 2^n) * 2^n) + (i rem 2^n) < 2 * (r + 1)^n
| |
1 BY (Assert ⌈(2^n * ((i ÷ 2^n) + 1)) ≤ (2^n * r + 1^n)⌉· THEN Auto')
| |
| ⊢ (2^n * ((i ÷ 2^n) + 1)) ≤ (2^n * r + 1^n)
| |
1 BY (BLemma `mul_preserves_le`  THEN Auto)
 \
  5. r: ℕ
  [6]. (r^n ≤ (i ÷ b)) ∧ i ÷ b < r + 1^n
  7. r2: ℤ
  8. r2 = (2 * r)
  9. r2': ℤ
  10. r2' = (r2 + 1)
  11. 2 * r^n = (2^n * r^n)
  12. 2 * (r + 1)^n = (2^n * r + 1^n)
  13. i = (((i ÷ 2^n) * 2^n) + (i rem 2^n))
  14. 0 ≤ (i rem 2^n)
  15. i rem 2^n < 2^n
  16. ¬r2'^n < i + 1
  ⊢ ∃r:{ℕ| ((r^n ≤ i) ∧ i < r + 1^n)}
  |
  BY (With ⌈r2⌉ (D 0)· THEN Auto' THEN ElimVar `r2\'' THEN ElimVar `r2' THEN Auto')·
  |
  6. r^n ≤ (i ÷ b)
  7. i ÷ b < r + 1^n
  8. r2: ℤ
  9. 2 * r ∈ ℤ
  10. r2': ℤ
  11. (2 * r) + 1 ∈ ℤ
  12. 2 * r^n = (2^n * r^n)
  13. 2 * (r + 1)^n = (2^n * r + 1^n)
  14. i = (((i ÷ 2^n) * 2^n) + (i rem 2^n))
  15. 0 ≤ (i rem 2^n)
  16. i rem 2^n < 2^n
  17. ¬(2 * r) + 1^n < i + 1
  ⊢ 2 * r^n ≤ i
  |
  BY Auto'
  |
  ⊢ 2 * r^n ≤ i
  |
  BY (Assert ⌈(2^n * r^n) ≤ (2^n * (i ÷ 2^n))⌉· THEN Auto')
  |
```

```
⊢ (2^n * r^n) ≤ (2^n * (i ÷ 2^n))
|
BY (BLemma `mul_preserves_le`  THEN Auto)
```

```
Extract:

λn.let b := 2^n in
  λx.letrec nth_root(x) =
    if x = 0 then 0
    else let z := x ÷ b in
         let r2 := 2 * (nth_root z) in
         let r3 := r2 + 1 in
           if (r3^n) < (x + 1)  then r3
           else r2 in
    nth_root(x)
```