Theorem 20a: $(C \vee (\sim C)) \Rightarrow (\exists x : T.\ True) \Rightarrow ((\sim (\forall x : T.\ (P\ x))) \Rightarrow (\exists x : T.\ (\sim (P\ x)))) \Rightarrow ((\forall x : T.\ (P\ x)) \Rightarrow C) \Rightarrow (\exists x : T.\ ((P\ x) \Rightarrow C))$

```
⊢ ∀[T:Type]. ∀[P:T → ℙ]. ∀[C:ℙ].
|     ((C ∨ (¬C))
|     ⇒ (∃x:T. True)
|     ⇒ ((¬(∀x:T. (P x))) ⇒ (∃x:T. (¬(P x))))
|     ⇒ ((∀x:T. (P x)) ⇒ C)
|     ⇒ (∃x:T. ((P x) ⇒ C)))
|
BY RepeatFor 3 ((UD THENA Auto))
|
[1]. T: Type
[2]. P: T → ℙ
[3]. C: ℙ
⊢ (C ∨ (¬C))
|   ⇒ (∃x:T. True)
|   ⇒ ((¬(∀x:T. (P x))) ⇒ (∃x:T. (¬(P x))))
|   ⇒ ((∀x:T. (P x)) ⇒ C)
|   ⇒ (∃x:T. ((P x) ⇒ C))
|
BY RepeatFor 4 ((D 0 THENA Auto))
|
4. C ∨ (¬C)
5. ∃x:T. True
6. (¬(∀x:T. (P x))) ⇒ (∃x:T. (¬(P x)))
7. (∀x:T. (P x)) ⇒ C
⊢ ∃x:T. ((P x) ⇒ C)
|
BY D 4
|\
| 4. C
| ⊢ ∃x:T. ((P x) ⇒ C)
| |
1 BY D 5
| |
| 5. x: T
| 6. True
| 7. (¬(∀x:T. (P x))) ⇒ (∃x:T. (¬(P x)))
| 8. (∀x:T. (P x)) ⇒ C
| ⊢ ∃x:T. ((P x) ⇒ C)
| |
1 BY (InstConcl [⌈x⌉]· THENA Auto)
| |
| ⊢ (P x) ⇒ C
| |
1 BY (D 0 THENA Auto)
| |
| 9. P x
| ⊢ C
| |
1 BY NthHyp 4
 \
  4. ¬C
  ⊢ ∃x:T. ((P x) ⇒ C)
  |
  BY D 6
  |\
  | 6. (∀x:T. (P x)) ⇒ C
```

```
| ⊢ ¬(∀x:T. (P x))
| |
1 BY (D 0 THENA Auto)
| |
| 7. ∀x:T. (P x)
| ⊢ False
| |
1 BY D 6
| |\
| | 6. ∀x:T. (P x)
| | ⊢ ∀x:T. (P x)
| | |
1 2 BY NthHyp 6
|  \
|   6. ∀x:T. (P x)
|   7. C
|   ⊢ False
|   |
1   BY (Unfold ‘not‘ 4 THEN D 4)
|   |\
|   | 4. ∃x:T. True
|   | 5. ∀x:T. (P x)
|   | 6. C
|   | ⊢ C
|   | |
1   2 BY NthHyp 6·
|    \
|     4. ∃x:T. True
|     5. ∀x:T. (P x)
|     6. C
|     7. False
|     ⊢ False
|     |
1     BY NthHyp 7
 \
  6. (∀x:T. (P x)) ⇒ C
  7. ∃x:T. (¬(P x))
  ⊢ ∃x:T. ((P x) ⇒ C)
  |
  BY D 7
  |
  7. x: T
  8. ¬(P x)
  ⊢ ∃x:T. ((P x) ⇒ C)
  |
  BY (InstConcl [⌈x⌉]· THENA Auto)
  |
  ⊢ (P x) ⇒ C
  |
  BY (D 0 THENA Auto)
  |
  9. P x
  ⊢ C
  |
  BY (Unfold ‘not‘ 8 THEN D 8)
  |\
  | 8. P x
```

```
    | ⊢ P x
    | |
    1 BY NthHyp 8·
     \
      8. P x
      9. False
      ⊢ C
      |
      BY FalseHD 9
```

Extract:

```
λd,e,f,g. case d of
    inl(c)  => let x,true = e in <x, λp.c>
  | inr(nc) => let x,np = f (λp1.(nc (g p1))) in <x, λp2.any (np p2)>

  where d : C ∨ (¬C)
        e : ∃x:T. True
        f : (¬(∀x:T. (P x))) ⇒ (∃x:T. (¬(P x)))
        g : (∀x:T. (P x)) ⇒ C
        c : C
        p : P x
       nc : ¬C ≡ (C ⇒ False)
       np : ¬(P x)
       p1 : ∀x:T. (P x)
       p2 : P x
```