Non-constructive Constant Time Algorithm

Time limit: 1.0s **Memory limit:** 256M

Let p be a function from the natural numbers to the booleans. Suppose that for all x, if $p(x)$ is true, then $p(x+1)$ is true. Prove that either:

1. $p(x)$ is false for all x.
2. There exists n such that $p(x) = \begin{cases} \text{false} & \text{if } x < n \\ \text{true} & \text{if } x \geq n \end{cases}$.

This proof has applications, too. For example, let $q(x)$ denote the truth value of "π does not contain x consecutive 9's in its decimal representation." Either $q(x) = \text{false}$, or there exists n such that $q(x) = (x \geq n)$. Therefore, q has a constant time algorithm, but the exact algorithm remains unknown.

Definitions

```lean
-- header.lean
def Simple (f : Nat → Bool) : Prop :=
  f = (fun _ => false) ∨
  ∃ n, f = (fun x => if x < n then false else true)

def NonconstructiveConstantTime : Prop :=
  ∀ p : Nat → Bool,
  (∀ n, p n → p (n + 1)) →
  Simple p

macro "check0123456789abcdef" t:ident : command => `(
  example : NonconstructiveConstantTime := $t
  #print axioms $t
)
```

Note: The macro's name is randomly generated on each submission, and will follow the format `check[0-9a-f]{16}`.

Proof Format

Your goal is to define a term `proof` with the type `NonconstructiveConstantTime`. You may use this template for your submission:

```
-- submission.lean
open Classical

def proof : NonconstructiveConstantTime := by
  admit
```

The judge will automatically prepend `import header` to your submission.

You may use the following axioms: `Classical.choice`, `Quot.sound`, `propext`

Checker

```
-- entry.lean
import header
import submission
check0123456789abcdef proof  -- 'proof' depends on axioms: [Classical.choice, Quot.sound, propext]
```

If you find a way to fool the checker, please open a ticket by clicking the "Report an issue" button at the bottom of the page, and add a link to your submission in the ticket.