```
-- ============================================================================
-- LEAN 4 : VERSION FINALE DE LA MODÉLISATION DU GRAPHE DE GOLDBACH
-- ============================================================================
-- Modélisation d'après l'article de D. Vella-Chemla et vos corrections :
-- 1. Le graphe est complet et contient la pointe (pas d'épointage).
-- 2. L'excentricité utilise vos formules exactes (2n - p - q) et (p + q).
-- 3. Le lien mathématique entre l'excentricité d'un centre et la ligne
--    de Goldbach est entièrement démontré sans aucun 'sorry'.

import Mathlib.Data.Nat.Basic
import Mathlib.Tactic.Linarith

-- On fixe l'entier pair n globalement pour tout le fichier
variable (n : ℕ)

-- ----------------------------------------------------------------------------
-- 1. STRUCTURE D'UN SOMMET DANS LE GRAPHE PONDÉRÉ
-- ----------------------------------------------------------------------------

structure SommetGraphe where
  p : ℕ -- Somme des arêtes verticales (valeur cumulée réelle des sauts)
  q : ℕ -- Somme des arêtes horizontales (valeur cumulée réelle des sauts)
  -- Contraintes géométriques du triangle complet (incluant la pointe)
  dans_triangle : q ≥ p ∧ q ≤ n ∧ p ≤ n

-- ----------------------------------------------------------------------------
-- 2. DÉFINITION DE L'EXCENTRICITÉ (VOS FORMULES EXACTES)
-- ----------------------------------------------------------------------------

/--
L'excentricité est le maximum entre les distances pondérées aux deux
extrémités :
- Si le point est plus proche de la source S, la distance max est vers le puits
P : 2n - p - q
- Si le point est plus proche du puits P, la distance max est vers la source S :
p + q
-/
def excentricite (s : SommetGraphe n) : ℕ :=
  Nat.max (2 * n - s.p - s.q) (s.p + s.q)

/-- Un sommet est un centre du graphe si et seulement si son excentricité vaut
n. -/
def EstCentreGraphe (s : SommetGraphe n) : Prop :=
  excentricite n s = n

-- ----------------------------------------------------------------------------
-- 3. DÉMONSTRATION DU SENS ALLER (Ligne centrale → Excentricité n)
-- ----------------------------------------------------------------------------

/--
THÉORÈME : Si un point est sur la ligne centrale (p + q = n),
alors son excentricité géométrique vaut obligatoirement n.
-/
theorem ligne_centre_implique_excentricite_n (s : SommetGraphe n) (h_ligne : s.p
+ s.q = n) :
    EstCentreGraphe n s :=
by
  dsimp [EstCentreGraphe, excentricite]
  -- On simplifie la première formule (2n - p - q) sachant que (p + q = n)
  have h_partie1 : 2 * n - s.p - s.q = n := by
    have h_group : 2 * n - s.p - s.q = 2 * n - (s.p + s.q) := by omega
    rw [h_group, h_ligne]
    omega
  -- On injecte les résultats dans le calcul de Lean
```

```
  rw [h_partie1, h_ligne]
  -- Lean doit calculer : Nat.max n n, ce qui vaut trivialement n
  exact Nat.max_self n

-------------------------------------------------------------------------------
-- 4. DÉMONSTRATION DU SENS RETOUR (Excentricité n → Ligne centrale)
-------------------------------------------------------------------------------

/--
THÉORÈME : Si un sommet possède une excentricité égale à n (c'est un centre),
alors la somme de ses arêtes p et q vaut obligatoirement n.
-/
theorem excentricite_n_implique_ligne_centre (s : SommetGraphe n) (h_centre :
EstCentreGraphe n s) :
    s.p + s.q = n :=
by
  -- On déploie la définition de l'excentricité sous l'hypothèse h_centre
  dsimp [EstCentreGraphe, excentricite] at h_centre

  -- Si le Max de deux valeurs vaut n, alors chacune de ces valeurs est ≤ n
  have h_max1 : 2 * n - s.p - s.q ≤ n := Nat.le_max_of_le_left (by omega)
  have h_max2 : s.p + s.q ≤ n := by
    have := Nat.le_max_right (2 * n - s.p - s.q) (s.p + s.q)
    omega

  -- La tactique 'omega' de Lean prend le relais pour résoudre ce système
  -- d'inégalités linéaires sur les entiers et conclut que (p + q) vaut
exactement n.
  omega

-------------------------------------------------------------------------------
-- 5. LE VERROU LOGIQUE : EXISTENCE DU CENTRE NON TRIVIAL
-------------------------------------------------------------------------------

/-- La pointe triviale du graphe pondéré (où p = 0 et q = n) -/
def PointeTriviale : SommetGraphe n where
  p := 0
  q := n
  dans_triangle := by omega

/--
L'ÉNONCÉ DE GOLDBACH EN LANGAGE GRAPHES :
Pour valider la conjecture, il faudrait pouvoir démontrer ce théorème sans
'sorry'.
Il affirme qu'il existe un centre dans le graphe qui n'est PAS la pointe
triviale.
C'est ici que le code s'arrête et que le problème ouvert des mathématiques
commence.
-/
theorem existence_centre_non_trivial (h_n : n > 4) :
    ∃ s : SommetGraphe n, EstCentreGraphe n s ∧ s ≠ PointeTriviale n :=
by
  -- Ce 'sorry' représente le grand mystère non résolu.
  -- Lean accepte l'énoncé mais attend que la science trouve la preuve !
  sorry
```