""" Copeland Recursive Harmonic Coherence Engine (Ψ-Formalism Core) Version: CRHC-v1.0-core Compatible with local LLM interfaces (Ollama + OpenWebUI) Author: Christopher W. Copeland (C077UPTF1L3) License: CC BY-SA / CRHC v1.0 (No commercial use without permission) """

import math import json

# Core Ψ-formalism harmonic engine

class PsiFormalismEngine: def **init**(self): self.history = [] # Recursive log of query/response nodes

def psi(self, x, delta\_E, an\_series, delta\_an):

"""

Ψ(x) = ∇ϕ(Σ𝕒ₙ(x, ΔE)) + ℛ(x) ⊕ ΔΣ(𝕒′)

"""

gradient\_phi = self.\_gradient\_phi(an\_series, x, delta\_E)

recursive\_R = self.\_recursive\_R(x)

delta\_sigma = self.\_delta\_sigma(delta\_an)

psi\_output = self.\_merge\_constructively(gradient\_phi + recursive\_R, delta\_sigma)

self.history.append({"x": x, "Ψ(x)": psi\_output})

return psi\_output

def \_gradient\_phi(self, an\_series, x, delta\_E):

total = 0

for n, a\_n in enumerate(an\_series):

total += a\_n \* math.exp(-delta\_E \* n) \* math.sin(x + n)

return total

def \_recursive\_R(self, x):

return 0.1 \* math.sin(2 \* x) # Correction/harmonization term

def \_delta\_sigma(self, delta\_an):

return sum(delta\_an) \* 0.05 # Small perturbation term

def \_merge\_constructively(self, value1, value2):

return value1 + value2 - (value1 \* value2 \* 0.01) # Non-linear merge operator

def export\_history(self, filename="psi\_output\_log.json"):

with open(filename, 'w') as f:

json.dump(self.history, f, indent=2)

# Example usage (can be removed or commented in production):

if **name** == "**main**": engine = PsiFormalismEngine() result = engine.psi( x=3.14, delta\_E=0.6, an\_series=[0.8, 0.5, 0.3], delta\_an=[0.01, -0.02, 0.015] ) print("Ψ(x) =", result) engine.export\_history()