NIVEL 1: ENTRADA DE PARÁMETROS & VALIDACIÓN DE PRECONDICIONES

```
FUNCTION ExecuteCostEngine(periodo, tenant_id)
============================================================
INPUT: periodo_fiscal (mes=03, anio=2026)
INPUT: tenant_id = 'CORP_INDUSTRIAL_PE'
INPUT: moneda_base = 'PEN (S/)'
```

```
DATA FETCHING (SUPABASE POSTGREST API)
============================================================
OPs = SELECT * FROM orden_produccion WHERE periodo = P
BOM = SELECT * FROM formula_insumos_detalle
KARDEX = SELECT * FROM articulo_costo_estandar
NOMINA = SELECT * FROM utilizacion_horas_hombre
CIF = SELECT * FROM comprobante_gastos_fabricacion
DRIVERS = SELECT * FROM determinacion_cost_drivers
```

```
ASSERT & QUALITY GATE 0 (VALIDACIÓN DE COMPLETITUD)
============================================================
IF (COUNT(OPs) == 0 OR COUNT(BOM) == 0) THEN
    THROW Exception('Catálogo incompleto para liquidación')
END IF
```

NIVEL 2: BUCLE PRINCIPAL DE ABSORCIÓN POR ORDEN DE PRODUCCIÓN

ALGORITMO 1: MATERIA PRIMA (MP)

```
PASO 1: EXPLOSIÓN BOM & MERMAS TÉCNICAS
============================================================
FOR EACH op IN OPs:
  op.total_mp = 0.0
  FOR EACH item IN BOM[op.codigo_producto]:
    q_bruta = item.cant_neta * (1.0 + item.pct_merma/100.0)
    cu_std = KARDEX[item.insumo_id].costo_unitario
    costo_linea = q_bruta * cu_std * op.cant_programada
    op.total_mp += costo_linea
  END FOR
END FOR
```

ALGORITMO 2: MANO DE OBRA DIRECTA (MOD)

```
PASO 2: LIQUIDACIÓN DE TARIFAS EFECTIVAS HH
============================================================
FOR EACH cc IN CentrosDeCosto:
  total_hh_cc = SUM(NOMINA[cc].horas_efectivas)
  total_soles_cc = SUM(NOMINA[cc].costo_planilla)
  tarifa_hh[cc] = total_soles_cc / total_hh_cc
END FOR

FOR EACH op IN OPs:
  op.total_mod = 0.0
  FOR EACH parte IN NOMINA_OP[op.numero_op]:
    hh = parte.horas + (parte.minutos / 60.0)
    op.total_mod += hh * tarifa_hh[parte.centro_costo]
  END FOR
END FOR
```

ALGORITMO 3: PRORRATEO & ABSORCIÓN CIF

```
PASO 3: INDUCTORES & COST DRIVERS
============================================================
FOR EACH drv IN DRIVERS:
  tasa_inductor[drv] = drv.costo_pool / drv.capacidad_total
END FOR

total_cif_absorbible = 0.0
FOR EACH comp IN CIF_Comprobantes:
  total_cif_absorbible += comp.monto_real
END FOR

FOR EACH op IN OPs:
  factor_op = op.total_hh_consumidas / Total_HH_Planta
  op.total_cif = total_cif_absorbible * factor_op
END FOR
```

NIVEL 3: DETERMINACIÓN DE COSTO TOTAL & COSTOS UNITARIOS

```
PASO 4: TOTALIZACIÓN & DIVISOR DETERMINISTA
============================================================
FOR EACH op IN OPs:
  op.costo_total_produccion = op.total_mp + op.total_mod + op.total_cif

  IF op.cantidad_terminada > 0 THEN
    op.cu_total = op.costo_total_produccion / op.cantidad_terminada
    op.cu_md    = op.total_mp / op.cantidad_terminada
    op.cu_mod   = op.total_mod / op.cantidad_terminada
    op.cu_cif   = op.total_cif / op.cantidad_terminada
    op.estado   = 'LIQUIDADA'
  ELSE
    op.cu_total = 0.0
    op.estado   = 'EN_PROCESO_WIP'
  END IF
END FOR
```

NIVEL 4: GENERADOR DE ASIENTOS CONTABLES (PCGE AUTOMATIZADO)

```
PASO 5: GENERACIÓN DE DIARIOS & CUADRE DEBE = HABER (100%)
============================================================
Total_PT_Liquidado = SUM(OPs[estado=='LIQUIDADA'].costo_total_produccion)

// Asiento 1: Transferencia a Existencias PT
Asiento_1.Lineas = [
   (Cta: '211101', Debe: Total_PT_Liquidado, Haber: 0.0),
   (Cta: '711101', Debe: 0.0, Haber: Total_PT_Liquidado)
]
ASSERT (SUM(Asiento_1.Debe) == SUM(Asiento_1.Haber))

// Asiento 2: Ajuste de Variación CIF
Variacion_CIF = Total_CIF_Real - Total_CIF_Absorbido
IF (ABS(Variacion_CIF) > 0.01) THEN
  Asiento_2.Lineas = [
    (Cta: '694101', Debe: Variacion_CIF, Haber: 0.0),
    (Cta: '791100', Debe: 0.0, Haber: Variacion_CIF)
  ]
  ASSERT (SUM(Asiento_2.Debe) == SUM(Asiento_2.Haber))
END IF
```

NIVEL 5: PERSISTENCIA TRANSACCIONAL & CIERRE KARDEX

```
PASO 6: COMMIT INMUTABLE EN SUPABASE POSTGRESQL
============================================================
BEGIN TRANSACTION;
  UPDATE orden_produccion SET ... WHERE id = op.id;
  INSERT INTO asiento_diario (lineas) VALUES (...);
  UPDATE cierre_kardex SET cerrado = TRUE, fecha = NOW();
COMMIT;

RETURN {
  status: 'SUCCESS_200',
  total_liquidado: Total_PT_Liquidado,
  ordenes_procesadas: COUNT(OPs),
  hash_auditoria: SHA256(Asientos)
}
```

🏆 MOTOR DR. COST: PSEUDOCÓDIGO EJECUTADO & CERTIFICADO (0 ERRORES)