# **Pre-Coding Essentials (Component: crates/vm\_algo/Cargo.toml, Version/FormulaID: VM-ENGINE v0) — 35/89**

## **1) Goal & Success**

Goal: Manifest for **vm\_algo** (tabulation, allocation, gates/frontier helpers) that depends only on vm\_core (types/math/rng) and not on I/O/UI.

Success: Builds as rlib on all targets; unit tests run; no JSON/FS or UI deps; optional feature flags gate algorithm families without changing public types.

## **2) Scope**

In scope: package metadata, edition/rust-version, features to toggle families (ranked, condorcet, pr, mmp, gates/frontier), minimal deps.

Out of scope: CLI, JSON/schema, report rendering, persistence.

## **3) Inputs → Outputs**

Inputs: Workspace toolchain; vm\_core API.

Outputs: vm\_algo rlib with modules tabulation/\*, allocation/\*, mmp, gates\_frontier.

## **4) Entities/Tables (minimal)**

## **5) Variables (build/features)**

## **6) Functions**

(Manifest only.)

## **7) Algorithm Outline (manifest structure)**

[package] name vm\_algo, version 0.1.0, edition 2021, rust-version = pinned toolchain; dual license.

[lib] name="vm\_algo", path="src/lib.rs", crate-type=["rlib"].

[features]

default = ["std","tab\_ranked","tab\_score","pr\_methods","mmp","gates","frontier"]

std = []

tab\_ranked = []

tab\_score = []

pr\_methods = []

mmp = []

gates = []

frontier = []

[dependencies]

vm\_core = { path = "../vm\_core" }

*(No serde/json/fs; keep pure algorithmic.)*

[dev-dependencies]

proptest = "1" *(optional for property tests of rounding/allocations)*

rand\_chacha = "0.3" *(tests only if we simulate ties; runtime RNG comes from vm\_core)*

## **8) State Flow**

vm\_pipeline calls into vm\_algo functions (tabulate/allocate/gates) using vm\_core types; vm\_algo has no file I/O.

## **9) Determinism & Numeric Rules**

All numeric ops use vm\_core::rounding helpers (integer/rational; half-even policy).

RNG, when required by tie policy, is injected from vm\_core::rng::TieRng—no dependency on external RNG crates here.

## **10) Edge Cases & Failure Policy**

Ensure no accidental dependency pulls in std::fs/serde\_json.

Feature flags must not change type layouts or function signatures in breaking ways (only compile scope).

Keep default-features of all transitive deps off (we only depend on vm\_core).

## **11) Test Checklist (must pass)**

cargo check -p vm\_algo with defaults.

cargo check -p vm\_algo --no-default-features --features "std,pr\_methods,gates" builds (ranked/score/mmp/frontier off).

Unit tests compile without any I/O deps; property tests pass for allocation edge cases (if enabled).