SuayLang Research Plan (v0.2)

Build: `make research-pdf` regenerates [docs/RESEARCH_PLAN.pdf](RESEARCH_PLAN.pdf).

Problem Statement

Many small languages ship a reference interpreter and a faster backend (VM/bytecode), but the two often diverge in subtle ways (semantics, diagnostics, determinism). SuayLang's goal is to make control flow explicit *and* make backend equivalence and diagnostic stability measurable and reproducible.

Research Questions

1) Can an expression-oriented language with explicit control-flow operators (`dispatch`, `cycle`) be specified and tested with a small, committee-reviewable contract?

2) Can we provide strong, reproducible evidence that two implementations (interpreter vs. bytecode VM) are observationally equivalent over a stated v1 scope?

Hypothesis (falsifiable)

Within the v1 scope, the interpreter and the VM are observationally equivalent under a fixed observation policy (termination class, normalized stdout, returned value when comparable, and error kind+span).

Method (how we will test it)

bullet**Semantic contract + golden diagnostics**: a small v1 contract document plus a suite of valid/invalid programs with golden snapshots for error kind/span/message shape.

bullet**Differential testing (interpreter vs. VM)**: deterministic generator by seed, multi-seed runs, size buckets (S/M/L), timeouts.

bullet**Comparator + normalization**: normalize paths and whitespace; compare outputs exactly after normalization; compare errors by kind + span (and stable formatting).

bullet**Minimization**: when a divergence is found, shrink the program and store a minimized regression case with metadata.

bullet**Coverage reporting**: record feature coverage (AST nodes and/or opcode families) to show exploration breadth.

bullet**Benchmarks**: measure parse/compile/interp/VM times with warmups and repeats; report median and p90 with raw samples.

Metrics & Success Criteria

All metrics are produced by repository commands and saved under `results/`.

Equivalence (primary)

bulletCI mode: seeds `0..9`, `N=500/seed` ⇒ **divergences = 0**.

bulletFull mode: seeds `0..99`, `N=2000/seed` ⇒ **divergences = 0**.

Diagnostics stability (primary)

bullet For the invalid-program contract corpus: **100% match** on error kind + span, and stable message prefix/shape.

Coverage (supporting)

bullet Coverage report includes counts by feature class (dispatch/cycle/functions/collections/errors). Success criterion: no major feature class is zero-covered in CI mode.

Benchmarks (supporting; not a correctness proof)

bullet Report median and p90 for parse, compile-to-bytecode, interpreter runtime, and VM runtime with ≥20 repeats + warmup. Success: the benchmark runner emits raw samples and environment metadata.

Experimental Protocol (repeatability)

bullet Runs are deterministic by **seed** and generator configuration.

bullet Profiles:

bullet **CI**: fast gate (seeds `0..9`, smaller `N`).

bullet **Full**: long local run (seeds `0..99`, larger `N`).

bullet Every run captures:

bullet Python version, OS, CPU info (best-effort), and git commit hash.

bullet Per-seed breakdown and size-bucket breakdown.

bullet Artifacts:

bullet `results/diff_report.json` + `results/diff_report.md`

bullet `results/coverage.json` + `results/coverage.md`

bullet `results/bench_raw.json` + `results/benchmarks.md`

Threats to Validity

bullet **Generator bias**: random generation may over/under-sample important features.

bullet **Observation policy limitations**: “equivalence” is with respect to the chosen observable outcomes.

bullet **Timeouts**: timeouts may hide non-termination differences.

bullet **Host effects**: Python runtime and OS scheduling noise affects benchmark timing.

bullet **Scope gaps**: behavior outside the declared v1 subset is intentionally unspecified.

Expected Results + What Would Falsify the Hypothesis

Expected: no divergences within v1 scope under CI and full differential testing profiles; stable diagnostics for the contract corpus.

Falsifiers:

bullet Any reproducible divergence in stdout/value/error kind+span for the same program input between interpreter and VM.

bullet Any change that breaks golden diagnostic snapshots for the v1 contract invalid-program corpus (without an explicit, documented version bump).

Timeline (milestones)

1) Finalize v1 contract and golden diagnostics corpus.

2) Scale differential testing to multi-seed CI + full profiles; enable minimization.

3) Add feature coverage reporting (AST/opcode) and publish reports to `results/`.

4) Add benchmark suite + runner with raw samples and noise controls.

5) Tooling polish: one-command install + `suay` CLI closed loop.

References

bullet Plotkin (1981), *A Structural Approach to Operational Semantics*.

bullet Kahn (1987), *Natural Semantics*.

bullet McKeeman (1998), *Differential Testing for Software*.

bullet Claessen & Hughes (2000), *QuickCheck*.

bullet Yang et al. (2011), *Finding and Understanding Bugs in C Compilers*.

bullet Le et al. (2014), *Compiler Validation via Equivalence Modulo Inputs*.

bullet Maranget (2008), *Compiling Pattern Matching to Good Decision Trees*.