Rubric for Programming Language Implementation

1. Lexical Analysis

Sub-Criteria	Points	Description	Sample Input	Expected Output / Behavior
Tokenizer Correctness	7	Recognizes valid tokens	`let x = 10 + 5;`	Breaks into: `'let'`, variable `x`, `'='`, number `10`, `'+'`, number `5`, `';'`
Error Handling	5	Identifies invalid tokens	`let x = 10 @ 5;`	Displays error: `Invalid token @`
Efficiency	3	No redundant steps	`let x = 10; // comment`	Skips comments/whitespace, minimal processing time

2. Syntax Analysis

Sub-Criteria	Points	Description	Sample Input	Expected Output / Behavior
Parser Correctness	10	Accepts valid syntax forms	`if (x > 5) { print("hi"); }`	Builds correct internal structure
AST Generation	7	Accurate AST structure	`x = a + b * c;`	AST: `Assign(x, Add(a, Multiply(b, c)))`
Error Handling	3	Detects malformed syntax	`if x > 5 {`	Syntax error: missing parentheses

3. Semantic Analysis

Sub-Criteria	Points	Description	Sample Input	Expected Output / Behavior

Symbol Table Management	5	Tracks variable declarations & scope	`let x = 10;`
			`let x = 15;`
				Redeclaration warning or error

Type Checking	5	Enforces type rules	`let x = "hi" + 5;`	Error: cannot add string and number
Error Detection	5	Detects semantic violations	`print(y);`	Error: `Variable y is not defined`

4. Execution and Evaluation

Sub-Criteria	Points	Description	Sample Input	Expected Output / Behavior
Expression Evaluation	6	Computes expressions	`print(2 * (3 + 4));`	Prints: `14`
Control Structures	6	Flow of control: if, while, for	`while (i < 3) { print(i); i = i + 1; }`	Prints: `0 1 2`
Function Execution	4	Calls and returns from functions	`function square(x) { return x * x; } print(square(5));`	Prints: `25`
CLI/REPL Support	4	Accepts runtime input	`> let name = input(); print("Hello, " + name);`	Executes interactively

5. Memory Management

Sub-Criteria	Points	Description	Sample Input	Expected Output / Behavior
Variable Lifecycle	4	Proper initialization & updates	`let x = 5; x = 6;`	Variable `x` updated correctly
GC / Manual Memory Handling	4	Memory cleanup managed	`let x = [1, 2, 3]; x = null;`	Memory cleared or collected
Access to Deleted Object	2	Prevents use-after-delete	`let x = [1, 2, 3]; delete(x); print(x[0]);`	Error: access to deleted object

6. Error Handling and Debugging

Sub-Criteria	Points	Description	Sample Input	Expected Output / Behavior
Runtime Errors	4	No interpreter crash	`let x = 10 / 0;`	Error: division by zero
Debugging Feedback	3	Error message with line info	`let x = "hi" + 5;`	Shows helpful message with line number

Sub-Criteria	Points	Description	Sample Input	Expected Output / Behavior
Logging / Verbose Mode	3	Optional execution trace	`toy run file.toy --verbose`	Shows token stream, parse trace, eval steps

7. User Interface / CLI

Sub-Criteria	Points	Description	Sample Input	Expected Output / Behavior
Usability	3	Clean interaction with CLI	`toy run sample.toy`	Executes program
CLI Arguments	2	Supports flags and script input	`toy run test.toy --debug`	Executes with debug mode
Output Format	2	Neat and consistent display	`print("Hi")`	Output: `Hi`
Help / Docs	3	Displays usage/help	`toy --help`	Shows command list and usage

8. Advanced Features

Sub-Criteria	Points	Description	Sample Input	Expected Output / Behavior
Object-Oriented Support	4	Class/object handling	`class Person { greet() { print("Hello"); } } let p = new Person(); p.greet();`	Output: `Hello`
Lambda Functions	3	Anonymous functions supported	`let add = (x, y) => x + y; print(add(2,3));`	Output: `5`

Concurrency / Async	3	Parallel execution (e.g., threads)	`parallel { print(1); }`
			`parallel { print(2); }`
				Output: `1 2` (order may vary)

Extensibility	2	Feature plug-in flexibility	`repeat 3 times { print("Hi") }`	New feature runs with minimal core changes

9. Presentation and Demonstration

Sub-Criteria	Points	Description	Sample Input	Expected Output / Behavior
Feature Walkthrough	4	Demonstrates all major features	Demo using examples from each rubric section	Features shown with explanations
Clarity & Engagement	2	Clear and engaging delivery	Presenter walks through slides/code	Maintains interest and clarity
Code & Demo Quality	2	Error-free execution	Run test suite and live demo	All code runs without issue
Slides/Attire	2	Professional presentation style	Styled slides and professional dress	Meets standard of academic/tech demo

Total Points: