

Red Pandas

Amina Assal

Ivan Barral

Rafail Khalilov

Myric Lehner



Project Management

Amina____Tester

Ivan____System Architect

Rafail___Language Guru

Myric____Project Manager



Visual Studio Code



GitHub



zoom

Language Overview

A close-up photograph of a giant panda's face as it eats bamboo leaves. The panda has its characteristic black and white fur, with black patches around its eyes and on its ears. It is holding a piece of bamboo in its mouth, and its teeth are visible as it chews. The background is a soft-focus green, suggesting a natural habitat.

Inspired by Python's Pandas (or NumPy) used for manipulating the matrices that make up linear algebra and so much machine learning.

Red Pandas

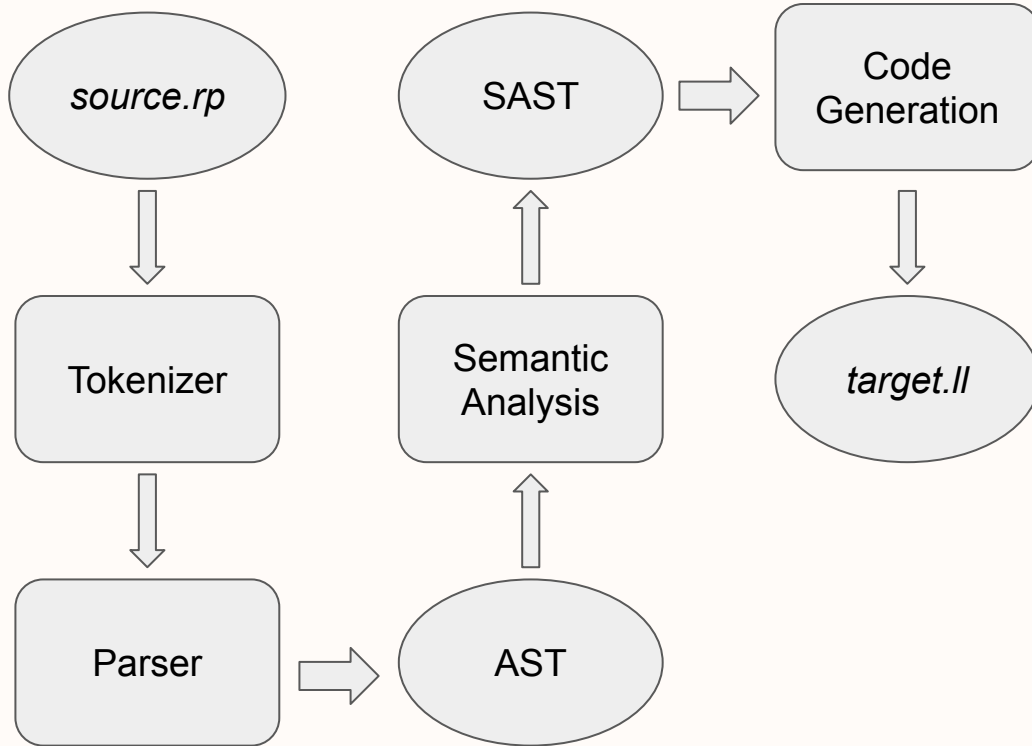
A photograph of a red panda looking directly at the camera. The red panda has reddish-brown fur with white markings on its face and chest. It has a long, bushy tail and is perched on a tree branch. The background is a soft-focus green, suggesting a natural habitat.

Smaller, distantly related

Offers python-style matrices, but compiled and without Python + Panda's interpretation overhead

C's speed but without the breadth

Compiler Architecture



Language Features

Core Features

- Strong Static Typing
- Matrices
- Lexical Scoping
- C's Operator Precedence

Core Functions

- Access
- Transpose

Primitives

int, float, string, bool

Matrix

[]; [[1][2]]; [[1,2,3,4,5],[8,7,6,2,10],[20,10,4,9,1]]

Control Flow Keywords

if, else, while, for, return

Arithmetic Operators + Assignment

+ - * / = .* ./

Logical Operators

! && ||

Conditional Operators

< > == != <= >=

Comments

// , /* ... */

Language Layout

```
matrix int [3][5] m;  
matrix int [3][5] test;  
printStr("All Values of matrix:");  
for(j = 0; j < m.row; j = j + 1 ){  
    for(i = 0; i < m.col; i = i + 1){  
        test[j][i] = m[j][i] * 2;  
        print(m[j][i]);  
def void scaleMatrix(int x) {  
    for(j = 0; j < m.row; j = j + 1 ){  
        for(i = 0; i < m.col; i = i + 1){  
            print(m[i][j] * x);  
        }  
    }  
}
```

Matrix type declaration

Required size declaration

Print function - strings

Row size attribute

Print for non-string values

Function declaration

Implementation Details

Each element of the Matrix is treated as an expression

As long as the types of the expressions match the declared matrix type, the compiler will accept the matrix.

parser.mly

```
...  
  
expr:  
  | LBRAK mat_opt RBRAK  { Mat($2) }  
  
mat_opt:  
  /* nothing */ { [] }  
  | row_list   { List.rev $1 }  
  
row_list:  
  LBRAK row_expr RBRAK { [(List.rev $2)] }  
  | row_list COMMA LBRAK row_expr RBRAK  
    { (List.rev $4) :: $1 }  
  
row_expr:  
  /* nothing */ { [] }  
  | expr       { [$1] }  
  | row_expr COMMA expr { $3 :: $1 }  
  
...
```

Future Work

More Matrix Functionality

- Declaring matrices without size
- More built-in functions for matrices
- Index out of bounds

Better Printing Visuals

- Escape Characters
- Formatted Matrix printing



Compilation & Code Demo