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

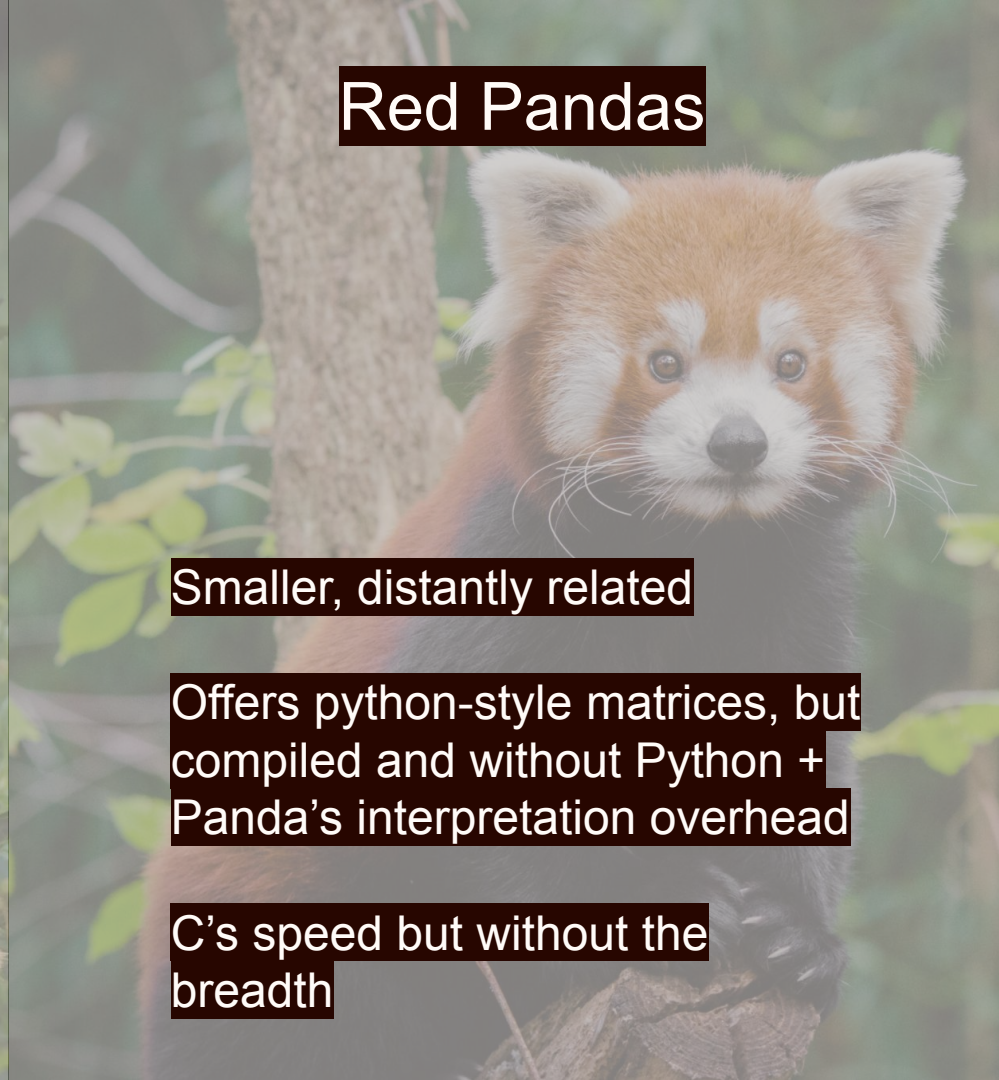
GitHub

zoom

Language Overview

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

Red Pandas

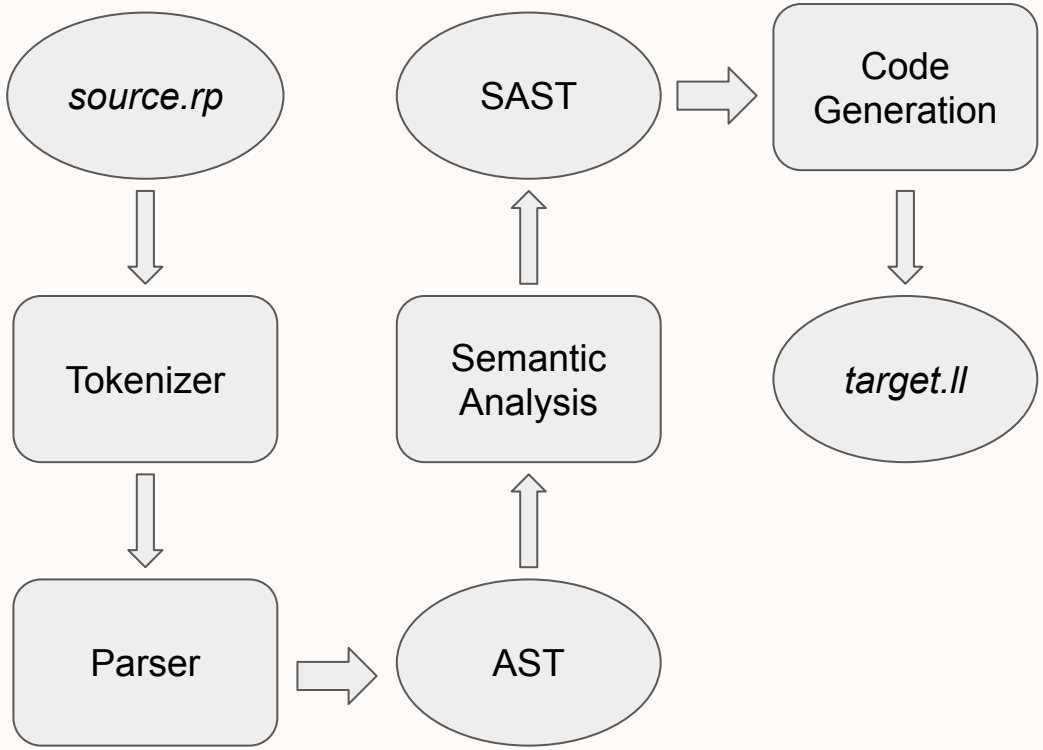


Smaller, distantly related

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:
  | LBRACK mat_opt RBRACK  { Mat($2) }

mat_opt:
  /* nothing */  { []          }
  | row_list     { List.rev $1 }

row_list:
    LBRACK row_expr RBRACK { [(List.rev $2)]     }
  | row_list COMMA LBRACK row_expr RBRACK
                           { (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