How to Implement a PL in < 10min

Programming languages are cool!

Devin Lehmacher

March 17, 2021

Background

- In addition to studying generally more useful computer science for software engineering in college: like algorithms, databases, operating systems, etc.
- I spent a great deal of time studying Programming Language theory (e.g. what makes a good programming language) and implementation (e.g. how is Rosalyn the C# compiler written)
- Language design opens possibilities
- Helps you think outside of the box when solving problems

What Language?

- Hint: we're not going to implement C# in 10 minutes.

What Language?

- Hint: we're not going to implement C# in 10 minutes.
- IMP (short for imperative), a very simple programming language

```
# Compute nth triangle number
n := 5;
i := n;
result := 0;
while i > 0 do (
  result := result + i;
  i := i - 1
);
print n
```

Basics

- For a typical interpreted programming language like Python or Javascript we have:

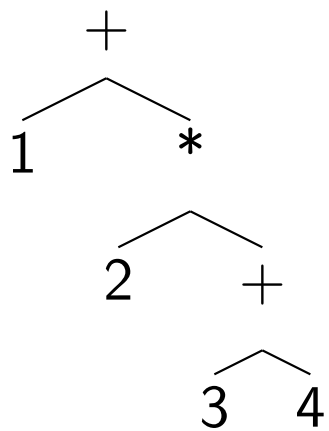
$$\text{Source Code} \xrightarrow{\text{Parser}} \text{AST} \xrightarrow{\text{Interpreter}} \text{Program Output}$$

Basics

- For a typical interpreted programming language like Python or Javascript we have:

$$\text{Source Code} \xrightarrow{\text{Parser}} \text{AST} \xrightarrow{\text{Interpreter}} \text{Program Output}$$

- Only going to implement an interpreter; so we'll start with Abstract Syntax Trees (ASTs)
- An expression like $1 + 2 * (3 + 4)$ would be represented as this AST:



Demo Outline

- Printing + Basic command stuff
- Assignment
- Conditionals (e.g. if)
- While

That's all!

- A more complete implementation of this simple language including a parser + repl can be found at https://github.com/lehmacdj/imp-lang. It is a little bit "better" of an implementation too:
  - It separates out booleans as a separate type from integers
  - It supports a fuller range of operations (e.g. all comparison operators and boolean operators; more arithmetic operations)
  - It is purer (it doesn't use IORef + IO in the evaluator)

That's all!

- A more complete implementation of this simple language including a parser + repl can be found at https://github.com/lehmacdj/imp-lang. It is a little bit "better" of an implementation too:
  - It separates out booleans as a separate type from integers
  - It supports a fuller range of operations (e.g. all comparison operators and boolean operators; more arithmetic operations)
  - It is purer (it doesn't use IORef + IO in the evaluator)
- Questions? (hopefully I didn't go overtime)