CSCI 334: Principles of Programming Languages

Lecture 16: Program interpretation

Instructor: Dan Barowy

Williams

Announcements

David Jensen, UMass Amherst

- Class of 60's talk:
  What's So Important About Explanation? Science, Machine Learning, and Large Language Models
  Thu at 7:30pm in Wege Auditorium
- Friday's colloquium:
  Explanation, Causation, and Mechanism in AI systems
  Fri at 2:35pm in Wege Auditorium

Topics

Program interpretation

Your to-dos

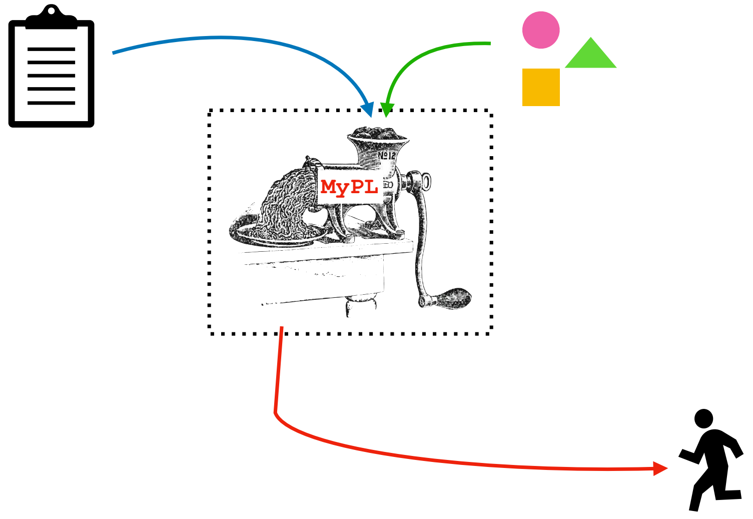
1. Watch *Growing a Language* before Thursday.
2. Lab #8, **due Sunday 11/12** (partner lab)

Final project timeline

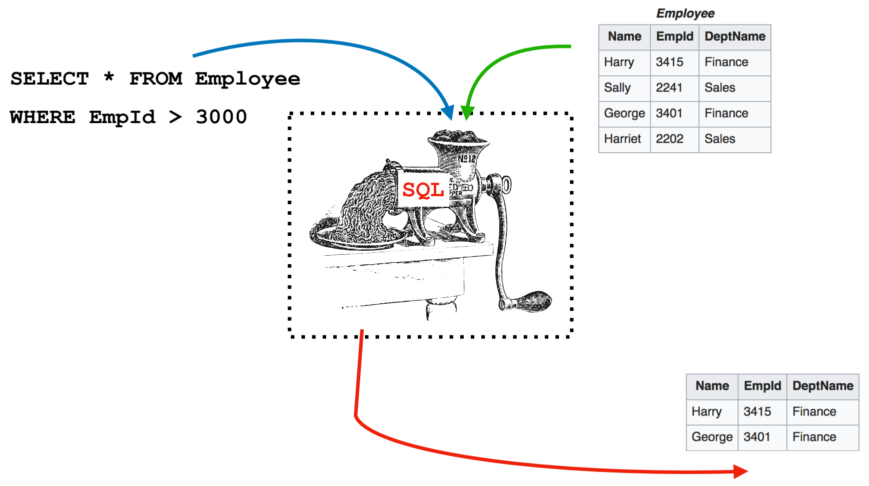
1. Project proposal (Lab 8), **due Sun 11/12**
2. Minimally working version (Lab 9), **due Sun 11/19**
3. Language specification doc (Lab 10), **due Sun 12/3**
4. Mostly working version (Lab 11), **due Sun 12/10**
5. Project + video presentation (Lab 12), **due Sun 12/17**

What is a programming language?

What is a programming language?



What is a programming language?



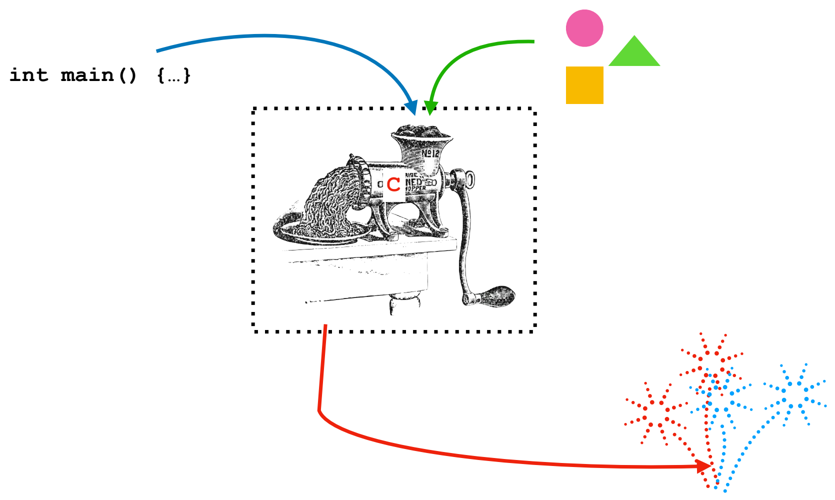
```
SELECT * FROM Employee
WHERE EmpId > 3000
```

Employee

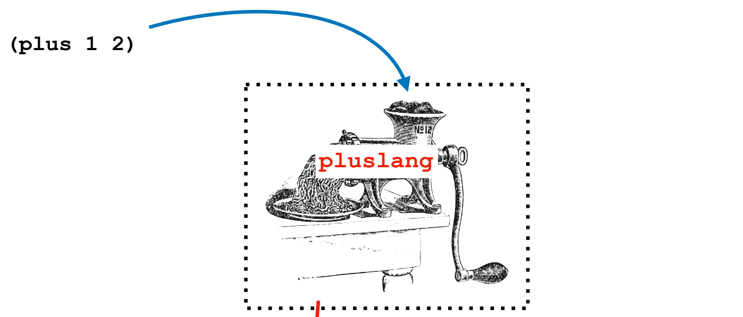
Name	EmpId	DeptName
Harry	3415	Finance
Sally	2241	Sales
George	3401	Finance
Harriet	2202	Sales

Name	EmpId	DeptName
Harry	3415	Finance
George	3401	Finance

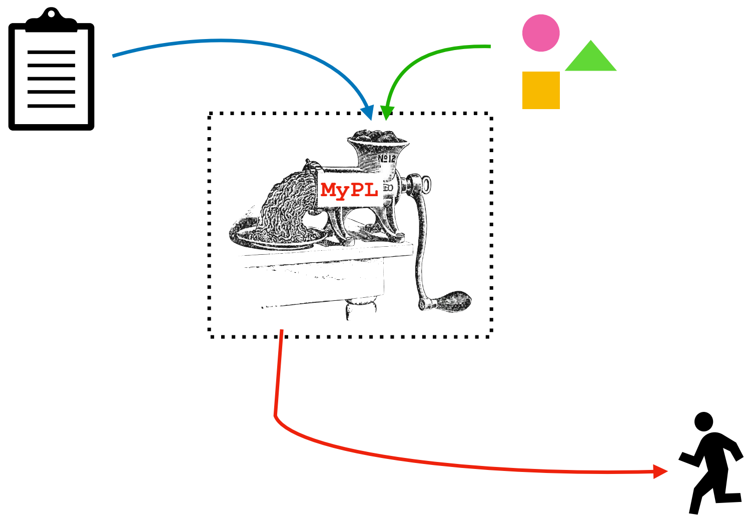
What is a programming language?



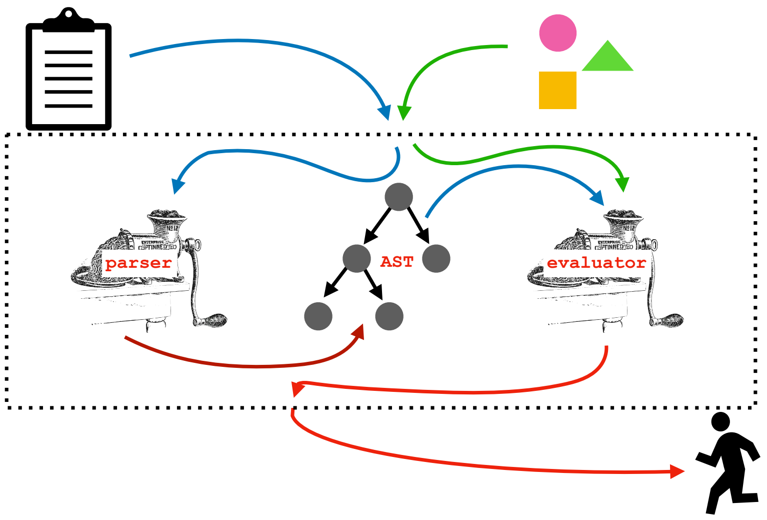
What is a programming language?



What is a programming language?



What is a programming language?



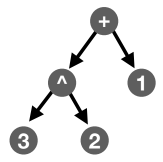
Program Interpreter

Program Interpreter

A **program interpreter** is a computer program that "interprets" given statements or expressions in a programming language. Unlike a compiler, an interpreter **directly** interprets code, often in the form of an **abstract syntax tree**.

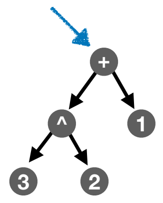
Example

```
3^2 + 1
```



Example

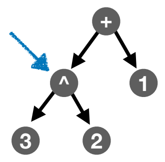
```
3^2 + 1
```



Eager evaluation: usually a **post-order traversal** of an AST.

Example

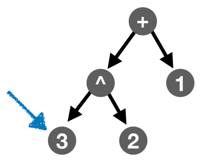
```
3^2 + 1
```



Eager evaluation: usually a **post-order traversal** of an AST.

Example

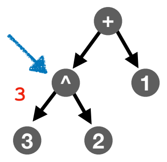
```
3^2 + 1
```



Eager evaluation: usually a **post-order traversal** of an AST.

Example

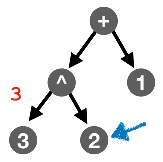
```
3^2 + 1
```



Eager evaluation: usually a **post-order traversal** of an AST.

Example

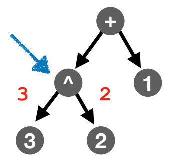
```
3^2 + 1
```



Eager evaluation: usually a **post-order traversal** of an AST.

Example

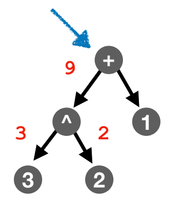
```
3^2 + 1
```



Eager evaluation: usually a **post-order traversal** of an AST.

Example

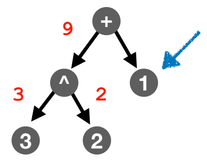
```
3^2 + 1
```



Eager evaluation: usually a **post-order traversal** of an AST.

Example

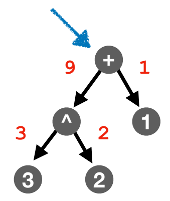
```
3^2 + 1
```



Eager evaluation: usually a **post-order traversal** of an AST.

Example

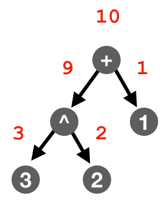
```
3^2 + 1
```



Eager evaluation: usually a **post-order traversal** of an AST.

Example

```
3^2 + 1
```



Eager evaluation: usually a **post-order traversal** of an AST.

This traversal is conveniently written as a **recursive function**.

pluslang

```
<expr> ::= (plus <expr> <expr>+)
         | n∈ℕ
```

```
(plus 1 2)

(plus 1 2 3 4 5)

(plus (plus 1 2) 3 4 5)
```

(code)

Recap & Next Class

Today:

Program interpretation

Next class:

Testing