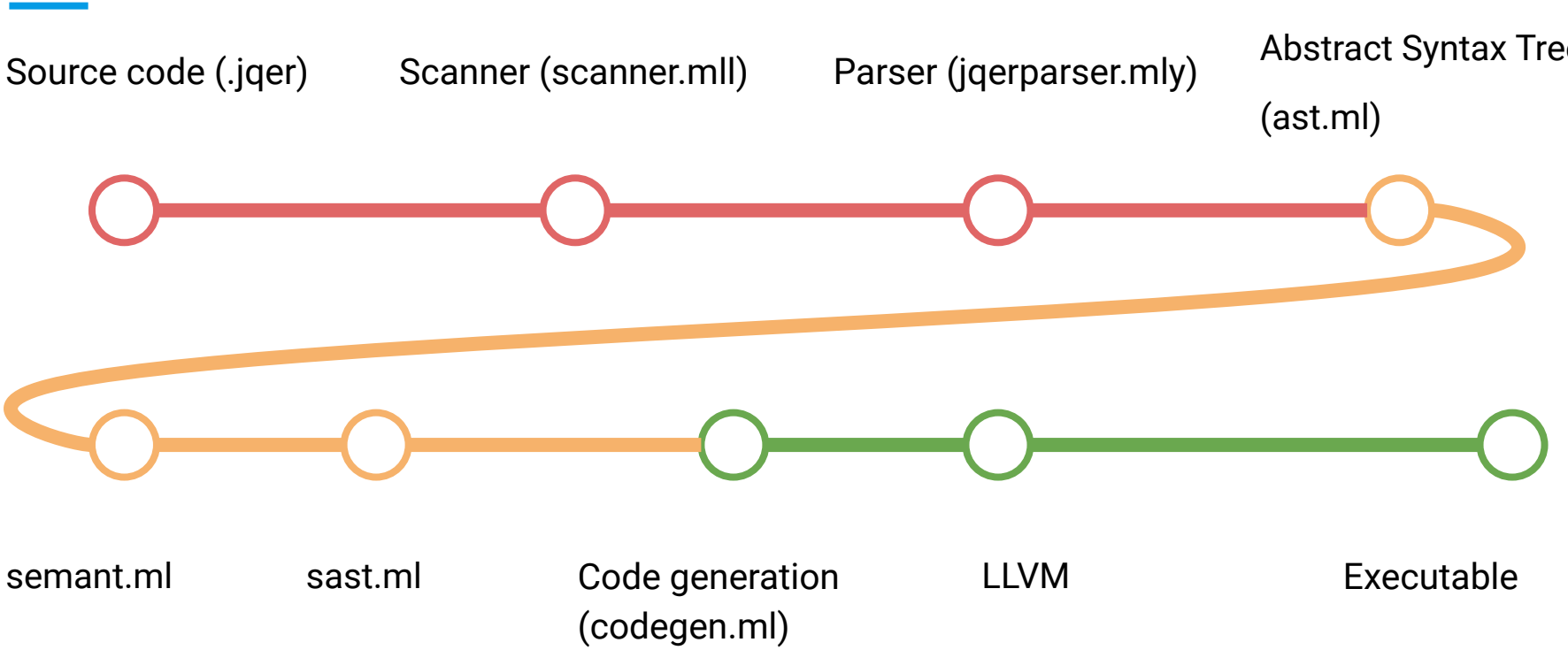
JQER

Python, but not real python

Jiaxuan Pan
Qianjun Chen
Eurey Noguchi
Roger Lu

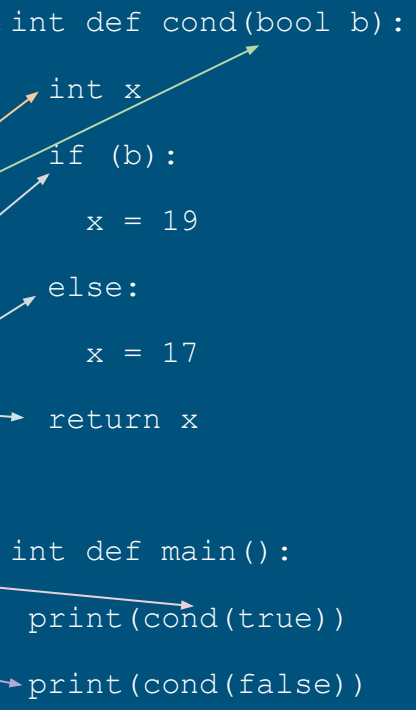
Motivation

- We like the Python syntax (indentation) for grouping
- We like strongly and static typed language
- Let’s combine them!


Source code (.jqer)
Scanner (scanner.mll)
Parser (jqerparser.mly)
Abstract Syntax Tree (ast.ml)
semant.ml
sast.ml
Code generation (codegen.ml)
LLVM
Executable

Syntax

- Function definition
  - Return type of function
  - Parameter(s) of function
  - Return statement
- Initialization of variable
- Main function
- Function call
- Print function call
- If else statement
- While and for loop



HOW DOES GROUPING WORK?

Functionalities

- Operators:
  - + - * % / = == > < >= <= ! and or
- Control Flow:
  - if (true): print(1) [else: print(0)]
  - while (true): print(1)
  - for (i = 0; i < 5; i = i + 1): print(1)
- Primitive Types:
  - int, bool, char, str, tuple
- Comments:
  - # comments

Testing

- Use shell script for suite automated testing and record keeping.
- Runs .jqer files for both passed and failed and record in .out files.
- Test suite composed with microC program rewrote in JQER and specific JQER features.
- Most bugs detected during compiler compile time (make) instead of suite compile time.

Challenges & Reflections & Future Work

- Challenges & Reflections:
  - Data structure like array is hard as the length for each index are not the same for different primitive types without pointer.
  - We initially were going to implement binary trees but it was hard to implement a dynamically changing type (and therefore could not finish)
    - We added a simpler structure: tuple
  - Aim low at first to build up on the basics.

- Future Work:
  - Struct like object-oriented programming without inheritance
  - Code linking to support module importing

DEMO TIME