Lab 0

Professor Sprenkle
sprenkles@wlu.edu
September 11, 2007

Objectives

- Why programming languages?
- Start lab #0
  - UNIX/Linux
  - Portal account (Blog for "Broader CS Issues")
  - Web page
  - Text Editor
  - Python

Why Do We Need Programming Languages?

- Computers can't understand English
  - Too ambiguous (PB&J)
- Humans can't easily write machine code

Problem Statement (English)

Machine code/Central Processing Unit (CPU)

000000 00001 00010 00110 00000 100000

Why Do We Need Programming Languages?

- Computers can't understand English
  - Too ambiguous (PB&J)
- Humans can't easily write machine code

Programmer (YOU!) **translates** from problem to algorithm (solution) to program

Problem Statement (English)
Algorithm/Pseudocode
High-level Programming Language (Python)
Bytecode

Python **interpreter** translates into bytecode

Machine code/Central Processing Unit (CPU)

Why Do We Need Programming Languages?

- Computers can't understand English
  - Too ambiguous (PB&J)
- Humans can't easily write machine code

Problem Statement (English)
Algorithm/Pseudocode
High-level Programming Language (Python)
Bytecode

Python **interpreter** executes the bytecode in a "virtual machine"
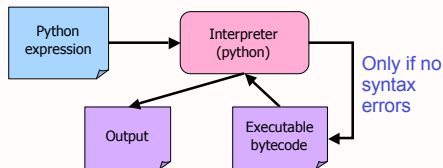
Machine code/Central Processing Unit (CPU)

Python Is

- A programming language
- An interpreter (a program)

Python Interpreter

1. Validates the Python programming language expression
   - Enforces Python syntax
   - Reports syntax errors ← Have a lot of these early on!
2. Simulates a computer (executes the expression)

Python expression → Interpreter (python)

Only if no syntax errors

Output ← Executable bytecode

Python Interpreter

1. Validates the Python programming language expression
   - Enforces Python syntax
   - Reports syntax errors ← Have a lot of these early on!
2. Simulates a computer (executes the expression)
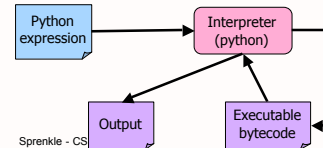   - Runtime errors (e.g., divide by 0)
   - Semantic errors (not what you *meant*)

- Good way to test expressions

Python expression → Interpreter (python)

Output ← Executable bytecode

Our Programming Process

1. Programmer types a program/script into a **text editor** (jEdit or IDLE).
2. An interpreter turns each expression into bytecode and then executes each expression

Text Editor (jEdit or IDLE) → Program text file program.py → Interpreter (python)

1 "line" at a time
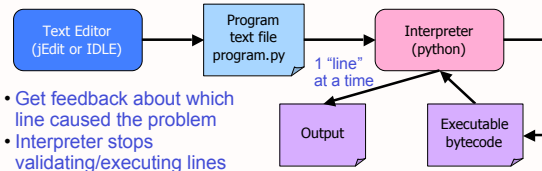
- Get feedback about which line caused the problem
- Interpreter stops validating/executing lines

Output ← Executable bytecode

Lab 0

- Start on paper --> Web, after log in
- Linux worksheet
  - Login
  - Open browser
  - Navigate to Lab 0, from "Schedule" page
- Continue on Web-based Lab 0