LECTURE 1: INTRODUCTION

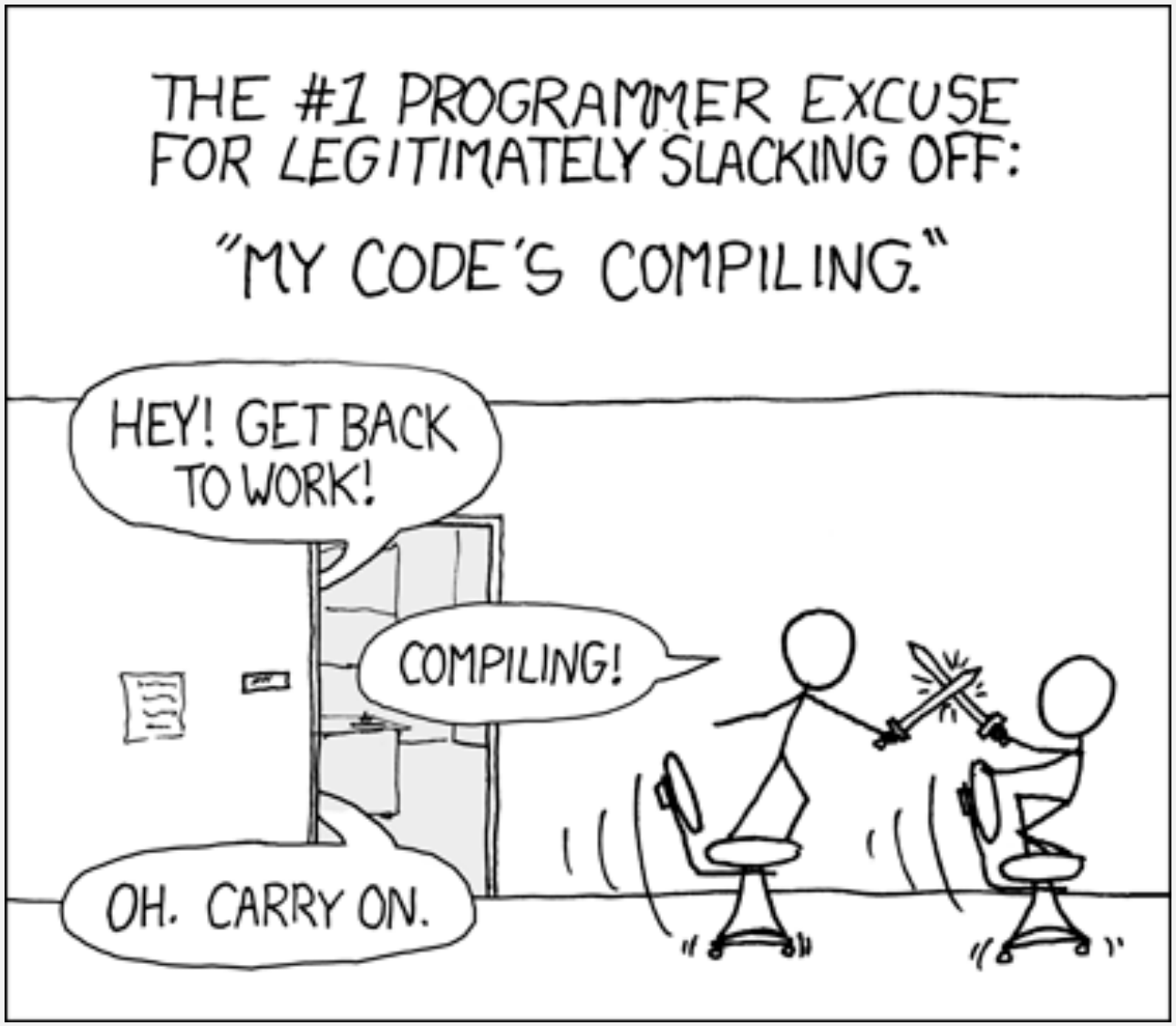
OUTLINE

- Scripting languages and popular applications
- Example scripting languages
- From low-level to high-level programming
  - Level of abstraction
  - Weakly or strongly typed

SCRIPTING LANGUAGES

Scripts are not too distinct from programs

- Scripting languages
  - are interpreted (vs compiled)
    - Rapid development: no compilation required
    - Portable: cross platform, no machine code
    - Slower: typically slower than compiled languages
- are good for controlling other applications
- are flexible
  - Vast modules/packages and extensions available



Source: https://xkcd.com/303/ (Licensed under CCA-Non Commercial 2.5)

INTERPRETER

- An interpreter is a program that executes scripts without first compiling them into machine code
  - How different compared to a compiler?
- An interpreter may
  - parse the source code and perform its behavior directly
  - translate source code into some efficient intermediate representation and immediately execute it
- Per statement (Read–Eval–Print Loop (REPL))

POPULAR APPLICATIONS OF SCRIPTING LANGUAGES

- System administration
- Web development: server side, client side
  - Many web frameworks
- Text processing / manipulation
- Gluing programs and tools
- Job control and automation
- Graphical User Interfaces

System Administration

- Deploying software and upgrading software
- Rotating logs
- Managing user accounts
- Performing backups and archives

Web Development

- Taking input from browser and constructing HTML pages online. Typical e-commerce site engine
- Creating web based applications: building a blog, implementing a wiki

Text Processing

- Effective and fast text processing

Gluing programs, modules, and tools

- Integration with other languages
- Connecting programs, modules, and tools
- E.g., calling C/C++ or shell commands within scripts

- Job control / automation
  - Running simulation experiments automatically
  - Checking storage status and notify users
  - TA marking of coding assignments
  - A Web crawler

- Graphic User Interfaces
  - Building a GUI front end of your own applications.

EXAMPLE SCRIPTING LANGUAGES

- There are many of them. Some examples:
  - Shell (many variants), PowerShell
  - sed, awk (Text processing)
  - Python, Perl, PHP
  - Ruby
  - Visual Basic (VBA, VBScript, VB.NET)
  - Javascript
  - Lua, R, …

FROM LOW-LEVEL TO HIGH-LEVEL PROGRAMMING

- Assembly languages: low-level programming
- System programming languages, e.g., C, Java
  - The compiler hides unnecessary details, have a higher level of abstraction compared to Assembly, results in increased productivity
  - Are strongly typed, i.e., meaning of information is specified before its use, enabling substantial error checking at compile time, may generate more efficient code

EVEN-HIGHER LEVEL PROGRAMMING

- Scripting languages provide an even higher- level of abstraction
  - The main goal is programming productivity, performance is a secondary consideration
  - Provide primitive operations with greater functionality
- Initial scripting languages were for job control
  - 1960s, IBM System 360 JCL (job control language)
  - Used to launch compilation, execution, and to check return codes

IBM 360 JCL EXAMPLE

```
$JOB    < Start of job
$FTN    < Load Fortran compiler
.
.       < Fortran instructions to be processed by compiler
.
$LOAD   < Load compiled instructions from tape
$RUN    < Transfer control to loaded program
.
.       < Data for program
.
$END    < End of job
```

A MORE RECENT EXAMPLE

```
#!/bin/bash
echo running $1
for (( c=1; c<=10; c++ ))
do
        python explicit_batch_train.py $1
done
```

AS GLUE LANGUAGES

- Often used to combine components
  - they assume a collection of useful components already exist in other languages
  - e.g., Unix shell scripts assemble filter programs into pipelines
    - ls –l | wc -l
  - e.g., interface between database and Web server
- Thus often called glue languages or system integration languages

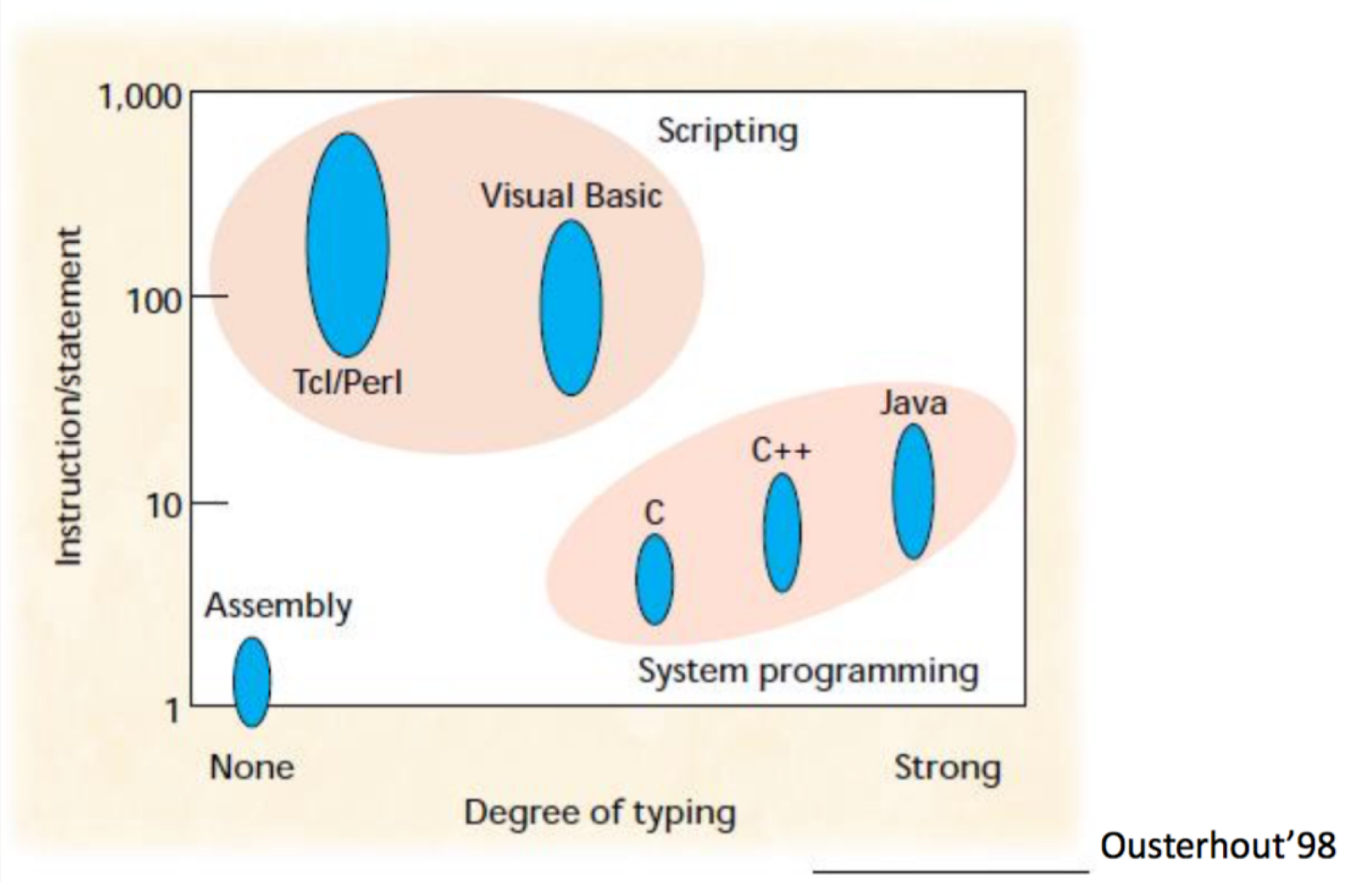
WEAKLY TYPED

- Scripting languages are weakly typed and generally type-less
  - Meaning of information is inferred
- It increases speed of development
  - Fewer lines of code
  - Flexibility. E.g., a variable can hold a string at a time and an integer the next moment

WEAKLY TYPED

```
use strict;
use warnings;
my $value = 42;
print $value + "3"
```

COMPARING LANGUAGES ON 2 DIMENSIONS



Ousterhout'98

READING FROM THE FIGURE

- Assembly language has no typing and very few instructions per statement
- Higher-level languages are weakly or strongly typed, execute more machine instructions for each language statement
  - System languages such as C has 5 to 10 instructions per statement
  - Scripting languages such as Tcl has 100 to 1000 instructions per statement

SUMMARY

- Interpreted, not compiled
- Popular applications
- A list of example scripting languages
- From low-level to high-level programming
  - Low: assembly, medium: system programming, high: scripting
  - Typeless, weakly typed, strongly typed
  - Low productivity to high productivity

FIN!