# C++: Syllabus & Introduction

# Outline

- Course Introduction
- C++ Introduction

# Who I am

- Instructor: Keith Perkins
- Office & Office Hours: See syllabus, course shell and course website

- Email: keith.perkins@cnu.edu

# Notes, Lectures, Assignments, Videos …

- Scholar
  - Not much except for Grades and announcements
- Course Webpage ← Most content here
- Webpage: Note in particular;
  - The Lectures/Readings section
    - You are responsible for everything here
  - The Examples section
    - Please understand these

# Assignments

- Read All week 1 readings
- Please go to projects section of website
    - Complete Project 0 by due date

# Syllabus: Prerequisites

- CPSC 255 or equivalent
- Textbook – Any C++ text
- Suggestions:
    - Absolute C++, Walter Savitch
    - C++ Programming Language, Stroustrup
- References to make you a better programmer
    - Effective C++, Scott Meyers
    - More Effective C++ , Scott Meyers
    - Effective STL, Scott Meyers
    - Effective Modern C++, Scott Meyers

# Syllabus: Major Topics

## (Subject to change)

- Week 1 C++ Intro, Market share, Compilation, GIT, Linux introduction
- Week 2 compilation, headers intro, makefiles, Eclipse
- Week 3,4 Headers, functions, Streams, Structs, Enums
- Week 5,6 Standard Library, strings
- Week 7 Standard Library iterators and Lists, Preprocessor directives
- Week 8,9 Pointers, References, Memory
- Week 10 Classes, operators, memory management using RAII
- Week 11 Exceptions
- Week 12, 13 Inheritance, operator overloading, virtual heiarchys
- Week 14 Registers, Memory, profiling

# Syllabus : Evaluation

- 2 Midterm Tests
- 1 Final
- Numerous projects
- See Syllabus for details

- This will be a rigorous course. Please start projects early.

# Syllabus: Assignments

- Project 0 – 0  points – IDE
- Project 1 - 100 points – makefile
- Project 2 - 100 points – File I/O
- Project 3 - 100 points – Modeling a simple system
- Project 4 - 100 points – Static libraries and parsing strings
- Project 5 - 100 points – Polymorphism

- This may change as the semester progresses

# Development Environment

- Could use vim, g++, gdb, valgrind, tmux for a command line only dev environment
- Or an IDE,  Lots to choose from, Codeblocks, Netbeans,Ms Visual Studio, Eclipse CDT…Clion
- We will use Eclipse CDT

# Operating System

- Linux – Ubuntu
- Can install yourself or (see course website for tutorial)
- Also running on the Hunter Creech lab computers.

- Compiler – GNU toolchain

# What you will learn

- Standard C++ - to a level of proficiency so you can function professionally, you will not be an expert.
- Some of the C++ syntax
- Coding suggestions and Guidelines to make you a better programmer.
- how to use an IDE, how to use libraries, how to approach and solve programming problems

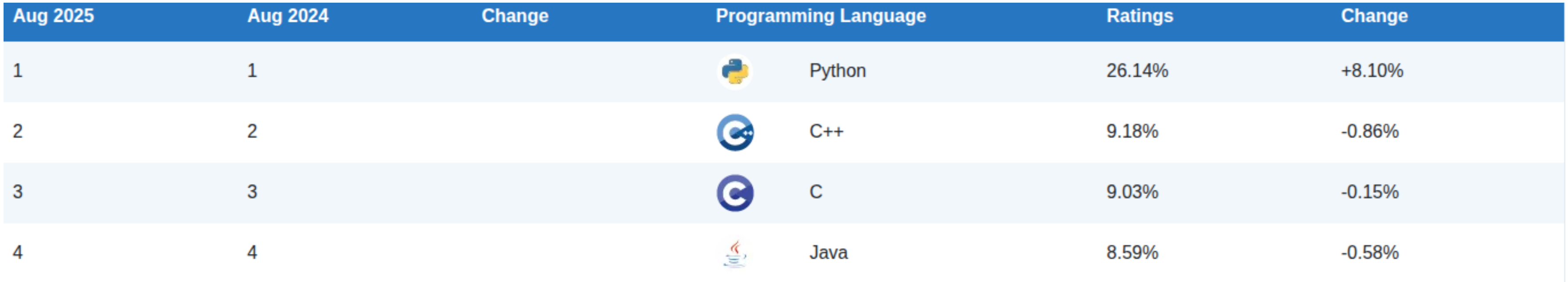
# What you will NOT learn

- User Interface (UI), networking– UI is platform dependent,  networking is too advanced for intro class (and is MUCH harder in C++ than Java)

# Outline

- Course Introduction
- C++ Introduction

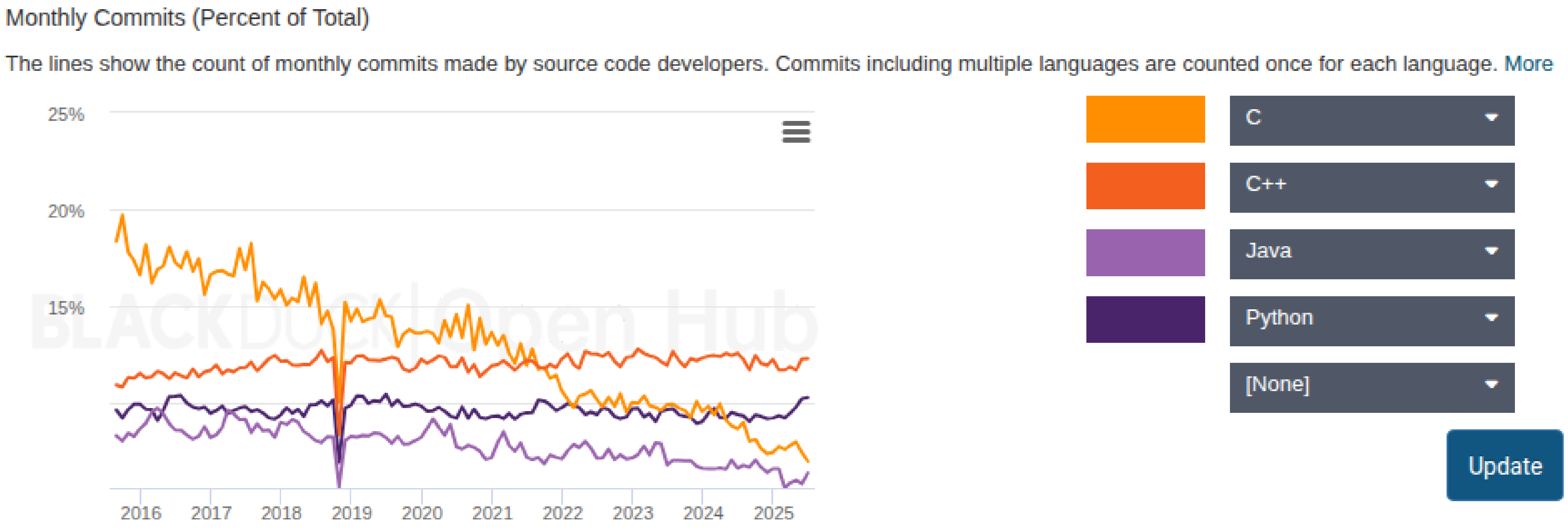
# C++ Usage



| Aug 2025 | Aug 2024 | Change | Programming Language | Ratings | Change |
|---|---|---|---|---|---|
| 1 | 1 | | Python | 26.14% | +8.10% |
| 2 | 2 | | C++ | 9.18% | -0.86% |
| 3 | 3 | | C | 9.03% | -0.15% |
| 4 | 4 | | Java | 8.59% | -0.58% |

See https://www.tiobe.com/tiobe-index/

# C++ Usage

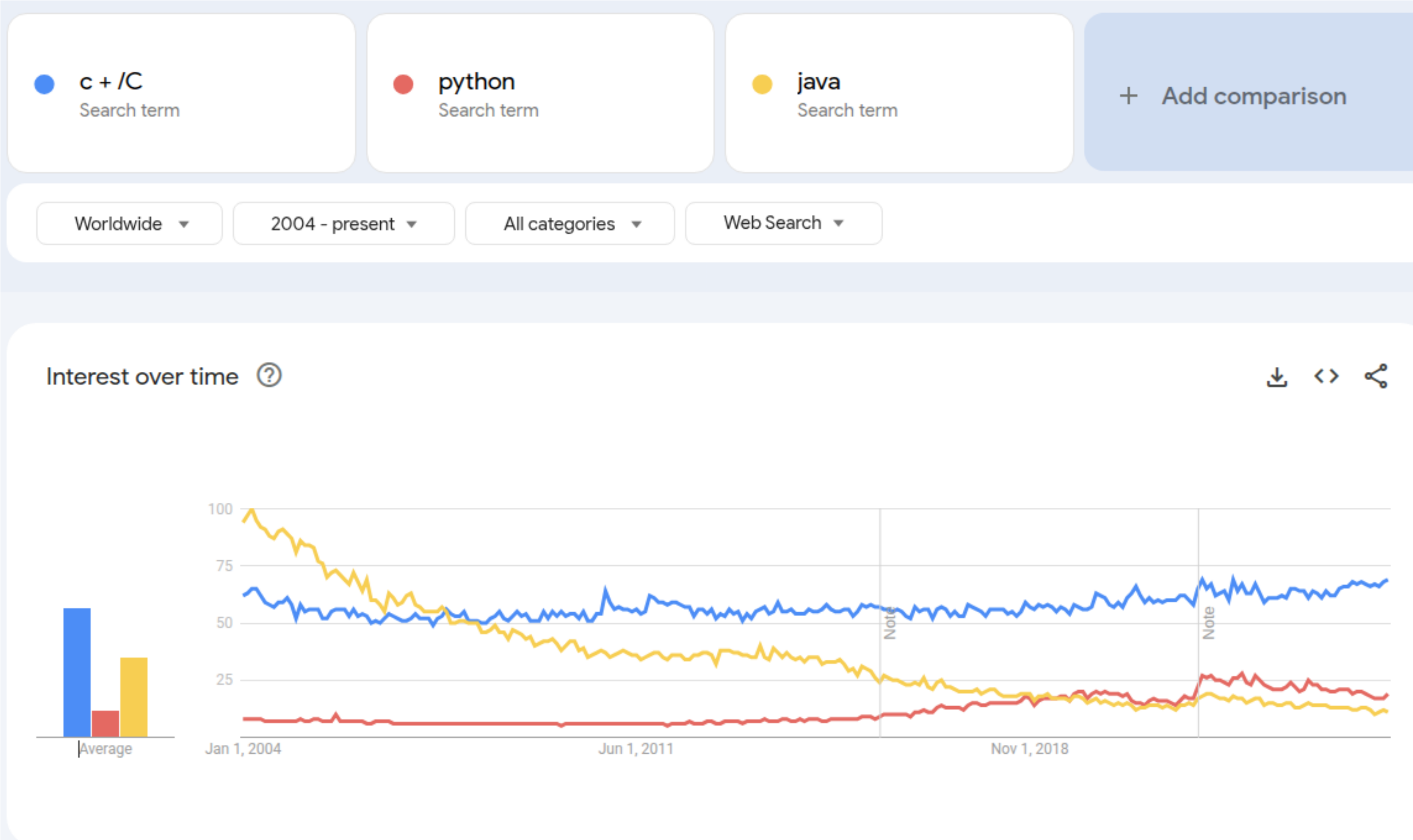


Monthly Commits (Percent of Total)

The lines show the count of monthly commits made by source code developers. Commits including multiple languages are counted once for each language. More

See  https://www.openhub.net/languages/compare

# C++ Usage



● c + /C
Search term

● python
Search term

● java
Search term

+ Add comparison

Worldwide ▾ | 2004 - present ▾ | All categories ▾ | Web Search ▾

Interest over time

See https://www.google.com/trends/explore?date=all&q=c%2B%2B%2FC,python,javascript

# C++ … Why?

- Fast
- You have absolute control over everything
- No need for virtual machine or interpreter
- Elegant when done well
- Only choice for some situations
  - High speed trading
  - Google search
  - Embedded systems
  - Real Time Processing
- Low level control

# C++ … Why not?

- Harder to code than languages that run on a VM (Java, C#)
- No garbage collection, pointers can be (and usually are) a problem
- Must be compiled to target platform, no portable bytecode
- My experience – My Python and Java applications are up and running faster than my C++ apps.

# C++ … Where is it used?

- Device driver development
- Video Games
- Advanced engines (audio, image processing, etc)
- Telecom
- Embedded software
- Financial - low latency market data feeds
- Google
- Real time video processing

# I know Python, why bother?

- Speed
- Software now targets distributed applications
  - Rich user interfaces
  - Cloud storage
  - Mobile Applications
  - Big Data
  - games
- Today, applications require expertise in multiple languages

# But… I don’t know most of that stuff

- Don’t worry, you aren’t expected to.
- You learn on the job (while getting paid)