

Programming Merit Badge

Badges at Barton, 2023

Counselor: David J. Jones

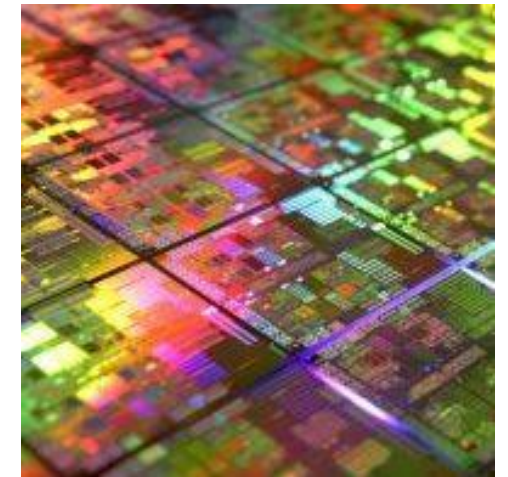


**BADGES AT
BARTON**
ADVENTURE • ADVANCEMENT • TRADITION



Introduction

- Instructor: David J. Jones
- Scouting:
 - Camp Barton Staff
 - 2004: Handicraft (Woodcarving, Metalwork)
 - 2005, 2006: Waterfront (Swimming)
 - Eagle Scout 2005. Troop 48, Lansing, NY.
- Professional:
 - BS Chemical Engineering, Clarkson University (2011)
 - MSE Nanotechnology, U. of Penn (2021)
 - MCIT (computer science), U. of Penn. (ongoing)
 - Semiconductor/nanotech engineer since 2011.
Currently a nanofabrication engineer at U. of Penn



**Penn
Engineering**
UNIVERSITY of PENNSYLVANIA

Safety

(Req. 1)

1a – Cyber Chip



- Show instructor your cyber chip!

1b – Safety hazards



- Injuries can happen when programming for many hours!
- Eye strain
 - First aid: Take a break.
 - Prevention. Consider using:
 - Dark mode in your code editor.
 - Blue light glasses or blue light screen filter
- Repetitive stress injuries
 - First aid
 - Apply an ice pack
 - Wrap the area with an elastic bandage
 - Stop coding
 - Prevention:
 - Good ergonomics
 - Take breaks if you feel you need them.



History of Programming (Req. 2)

2 - History of Programming Binary and Assembly

- In this course, we'll focus on digital computers (as opposed to mechanical or analog computers).
- Ultimately, digital computers “understand” binary (0's and 1's).
 - All code that we write ultimately gets translated to 0's and 1's.
 - We call this machine code.
- Instead of writing code in 0's and 1's, programmers developed “assembly” language, which can be assembled (translated) into machine code.

```
110101010101010010100010
100101111000001000100101
010101010101001011100011
010001010010101110001001
010101010000111110101010
010101010101010100100011
```

Assembly Language

```
mov ecx, ebx
mov esp, edx
mov edx, r9d
mov rax, rdx
```

Programmer

Assembler + Linker

Machine Language

```
100101011001
010011111011
111010101101
01010101010
```

Processor

2 - History of Programming

Unix, C



- Unix was an early operating system (first release in 1969). Widely considered the “grandfather” of operating systems that are used today.
- C programming language released in 1972, and Unix was re-written in C.
- C compiles directly to assembly, and is very fast. C is still widely used today in “low-level” applications, e.g. operating systems, and game engines.
 - Linux is written in C. Technically, Linux is a “kernel” (software that runs the hardware).

2 - History of Programming

Object-Oriented Programming (OOP)

- C++ was released in the 1980s. It has all of the same core features of C, but it supports “object-oriented programming” (OOP).
 - OOP allows the programmer to create “objects” which can interact with other “objects”, which are entities that contain data and code.
- Java was released in the early 1990s. It has similar syntax to C and C++. But, Java has some key features:
 - It’s designed to be platform-independent. “Write once, run anywhere”.
 - It has automated memory management (called “garbage collection”).
 - We’ll use Java in this course!



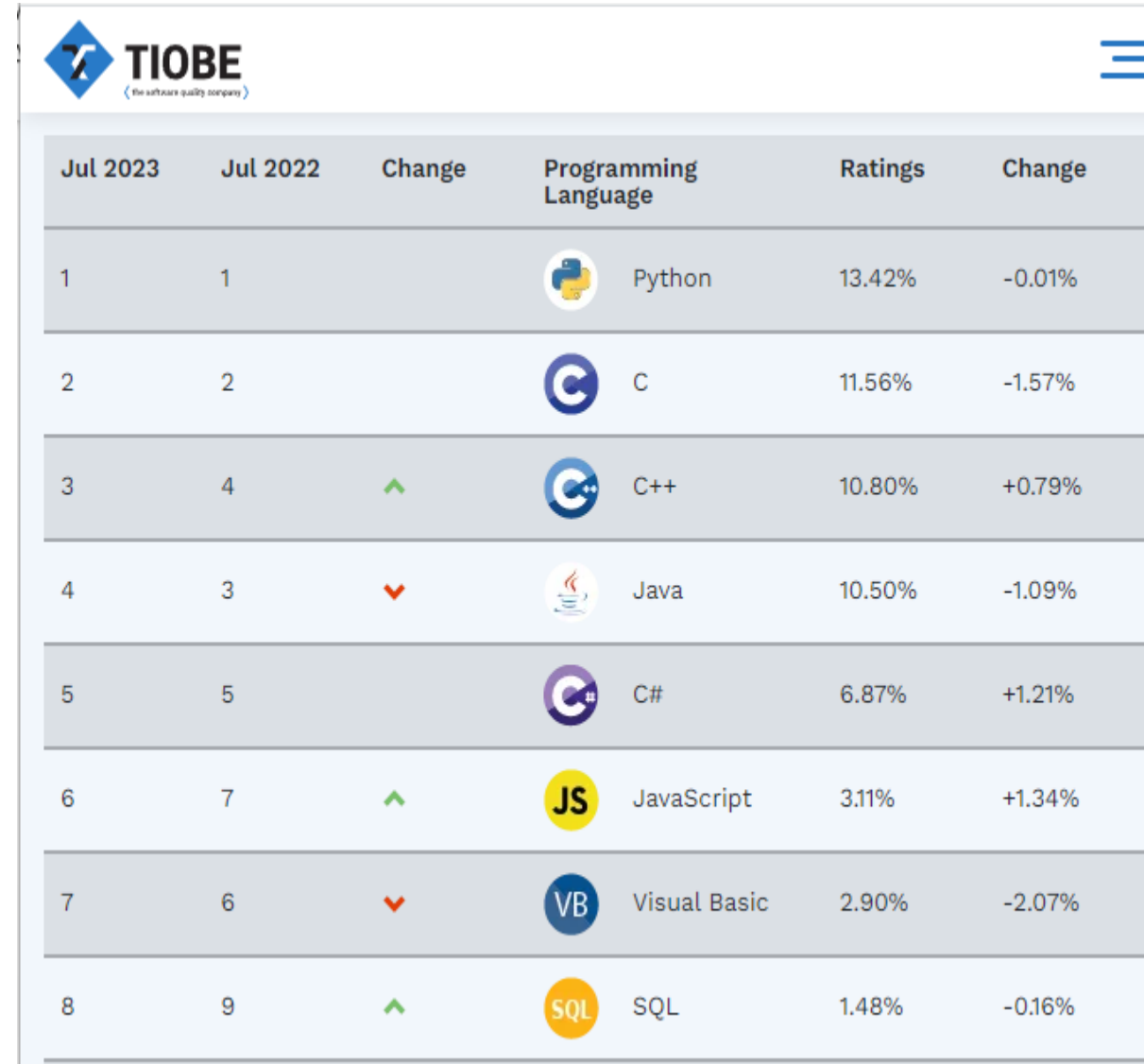
2 - History of Programming World Wide Web












- HTTP (HyperText Transfer Protocol) was introduced in 1991. Made the internet possible.
 - Gave everyone a standard for data transfer on the World Wide Web, and makes the internet possible.
 - HTTP isn't a programming language, but rather a set of rules for everyone to follow.
- HTML (HyperText Markup Language) was introduced in 1993. It's the first language written specifically for the internet. Used for websites.
- JavaScript was introduced in 1995. It's used to enhance web pages in ways that HTML can't.
 - We'll use JavaScript in this course!

2 - History of Programming Modern Languages

- Computing is constantly updating and evolving!
- TIOBE Index tracks the most popular languages in use!
 - <https://www.tiobe.com/tiobe-index/>
- In this course, we'll use:
 - Python
 - JavaScript
 - Java
- C and C++ are great languages to learn, but they're difficult for beginners (in the instructor's opinion).



 TIOBE <small>the software quality company</small>						
Jul 2023	Jul 2022	Change	Programming Language		Ratings	Change
1	1			Python	13.42%	-0.01%
2	2			C	11.56%	-1.57%
3	4	▲		C++	10.80%	+0.79%
4	3	▼		Java	10.50%	-1.09%
5	5			C#	6.87%	+1.21%
6	7	▲		JavaScript	3.11%	+1.34%
7	6	▼		Visual Basic	2.90%	-2.07%
8	9	▲		SQL	1.48%	-0.16%

2 - History of Programming

Turing Award

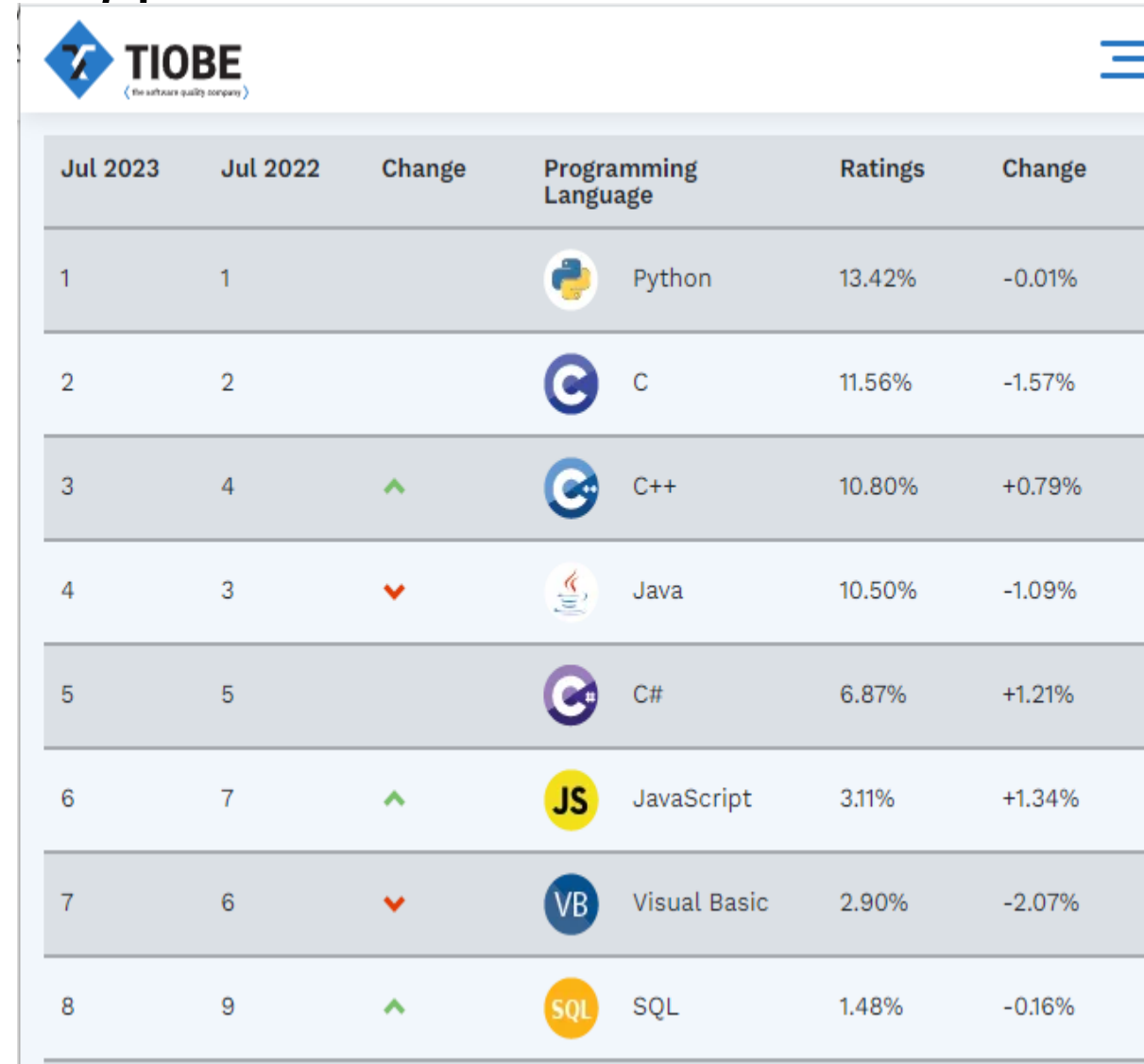
- If you're interested to learn more about the history of computing, a good place to start is to look at the Turing Award winners.
 - The Turing Award is the Computer Science version of the Nobel Prize
- Things we've covered that have won the Turing award:
 - UNIX (1983 – Ken Thompson & Dennis Ritchie)
 - Object oriented-programming (2001 – Ole-Johan Dahl & Kristen Nygaard)
 - HTTP (2016 – Tim Berners-Lee)











General Knowledge (Req. 3)

3a – Programming Languages

- TIOBE tracks most popular languages.
 - TIOBE index:
<https://www.tiobe.com/tiobe-index/>
- As of July 2023, Python is #1
 - We'll use Python in this course!



The screenshot shows the TIOBE index website. At the top, there is a logo for TIOBE (the software quality company) and a hamburger menu icon. Below the header is a table with the following columns: Jul 2023, Jul 2022, Change, Programming Language, Ratings, and Change. The table lists the top 8 programming languages. Python is at the top with a rating of 13.42% and a change of -0.01%. C is second with 11.56% and -1.57%. C++ is third with 10.80% and +0.79%. Java is fourth with 10.50% and -1.09%. C# is fifth with 6.87% and +1.21%. JavaScript is sixth with 3.11% and +1.34%. Visual Basic is seventh with 2.90% and -2.07%. SQL is eighth with 1.48% and -0.16%.

Jul 2023	Jul 2022	Change	Programming Language	Ratings	Change
1	1		 Python	13.42%	-0.01%
2	2		 C	11.56%	-1.57%
3	4	▲	 C++	10.80%	+0.79%
4	3	▼	 Java	10.50%	-1.09%
5	5		 C#	6.87%	+1.21%
6	7	▲	 JavaScript	3.11%	+1.34%
7	6	▼	 Visual Basic	2.90%	-2.07%
8	9	▲	 SQL	1.48%	-0.16%

3a – Programming Languages (Python)

- Python is very popular for several reasons:
 - Beginner-friendly, syntax is usually compact
 - Can be used for a wide variety of applications:
 - Data processing
 - Artificial intelligence and machine learning
 - Web and software development
 - Think of it as a “swiss-army knife” of programming languages!



Top 10 Python Frameworks

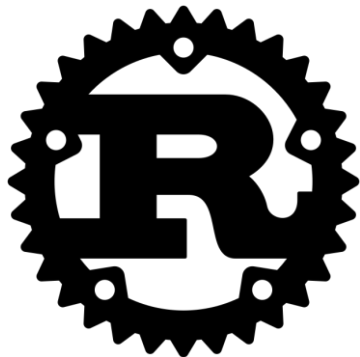
✓ django	✓ WEB2PY
✓  CherryPy	✓  Falcon
✓ GIOTTO	✓  Flask <small>web development, one drop at a time</small>
✓ 	✓ Bottle
✓  CubicWeb	✓ Dash

Advantages 

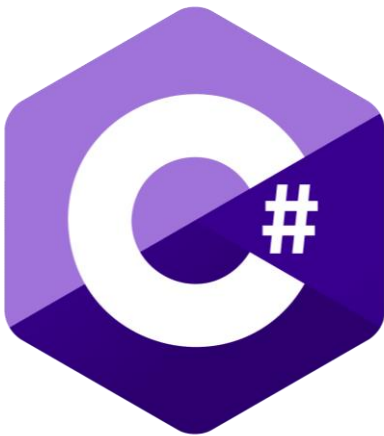
Disadvantages 

Extensive support libraries	Slow speed
Portable and interactive	Runtime errors
User friendly data structure	Bad for memory consumption

3a – Programming Languages (Low-level, and enterprise)



- C : Used in applications where performance is critical. E.g. operating systems, video games, compilers.
- C++ : Similar use cases as C
- Rust: similar use cases as C and C++, but is memory safe. Newer language, so not as well developed as C or C++ .
- Java : Game development, cloud computing, big data, artificial intelligence, internet of things
- C# (C-sharp) : similar use cases as Java. By Microsoft.



3a – Programming Languages (Web applications)

- HTML – Used to write web pages
- CSS (Cascading Style Sheets) – Used to style web pages. Used with HTML.
- JavaScript – Used to enhance webpages.
- PHP - web development scripting language. Was used more in the early days of the web.

HTML



CSS



JS



3a – Programming Languages

- Some are used in scientific computing
 - C++
 - MATLAB (MATrix LABoratory): Popular choice for engineering.
 - User-friendly, but slow (in the instructor's opinion). Also, not free to use ☹ .
 - FORTRAN (FORMula TRANslation): One of the earliest programming languages (released in the 1950s), but is very fast.
 - R: Popular choice for statistics



3b – Programmed Devices

- What devices do you use that run programs?
 - This computer!
 - Smartphones.
 - Most modern cars.
 - Thermostat.

Intellectual Property (Req. 4)

4a – Intellectual Property

- Copyright protection:
 - You can't copy a copyrighted work without the author's consent (while it's under copyright protection). For example, suppose it cost you 4 years of your life and many thousands of dollars to develop a game, and people could just copy it for free. Is that right? Where would be the incentive to sink that kind of cost into making the game?
- Patents:
 - Patents allow you to essentially get copyright protection on an idea or an invention. Similar to the copyright problem – helps preserve the financial incentive to innovate and invent.



4a – Intellectual Property

- Trademarks:
 - Used to protect the name of something. E.g. if I invent the “DaveScript” programming language, I wouldn’t want someone else to be able to call their new programming language “DaveScript”. That would get very confusing...
- Non-disclosure agreement (NDA):
 - An agreement where one party agrees to not give away information to another.
 - For example: businesses will often have employees sign an NDA as a condition of working there. If it’s discovered that the employee gave company secrets away (i.e. to a competitor), then the company could sue the employee.



4b – Licensing vs. Owning Software

- Licensing: Think of it as renting the software instead of owning it. You'd be able to use the software for an agreed-upon length of time.
- Owning: Just like you'd think. You get to use the software as long as you want.

4c – Freeware vs. open source vs. commercial

- Freeware – Free to use, but possible to allow limited use (not unlimited use).
- Open source – A type of freeware, but usually unlimited use. All of the source code is freely available too.
- Commercial software – Pay to use.
- Why should you respect the terms of use of each?

Let's do some programming!
(Req. 5)

Careers (Req. 6)

6 - Careers

- Software Engineer
 - Build software systems and applications.
 - Qualifications: Usually bachelors in Computer Science. Web developers can often take a coding bootcamp (with no college degree).
- Data Scientist
 - Examine and analyze large sets of data to draw insights for data. Machine learning falls under data science.
 - Qualifications: Bachelors in Computer Science, Data Science, or similar program.
- Engineer/Scientist
 - Engineers and scientists in other fields program too!
 - Qualification: Bachelors in an engineering field (e.g. Mechanical, Chemical, Electrical, Civil, Materials).

6 - Careers

- An example Computer Science curriculum (Cornell University)

CS REQUIRED COURSES

CS 111x	Introducton to Computing
CS 2110	Object-Oriented Programming and Data Structures
CS 2800	Discrete Structures
CS 3110	Data Structures and Functional Programming
CS 3410	Computer System Organization and Programming
or	
CS 3420	Embedded Systems
CS 4410	Operating Systems
CS 4820	Introduction to Analysis of Algorithms

CS SAMPLE ELECTIVE COURSES

CS 1300	Introductory Design and Programming for the Web
CS 1620	Visual Imaging in the Electronic Age
CS 1710	Introduction to Cognitive Science
CS 2024	C++ Programming
CS 2043	UNIX Tools and Scripting
CS 2300	Intermediate Design and Programming for the Web
CS 2850	Networks
CS 3152	Introduction to Computer Game Architecture
CS 3758	Autonomous Mobile Robots
CS 4120	Introduction to Compilers
CS 4220	Numerical Analysis: Linear and Nonlinear Problems
CS 4300	Language and Information
CS 4320	Introduction to Database Systems
CS 4620	Introduction to Computer Graphics
CS 4700	Foundations of Artificial Intelligence
CS 4740	Natural Language Processing
CS 4780	Machine Learning for Intelligent Systems
CS 4812	Quantum Information Processing
CS 4860	Applied Logic

Thank You!



**BADGES AT
BARTON**
ADVENTURE • ADVANCEMENT • TRADITION



Extra Slides

2 - History of Programming

Early languages

- Assembly is fairly difficult to write, so higher level languages were developed to “compile” to assembly. Some early examples include (developed in the 1950s):
 - FORTRAN (FORmula TRANslation) – developed by IBM. Still sees some use in scientific computing.
 - COBOL (COmmon Business Oriented Language) – was popular for business data processing. Saw use in large organizations: e.g. banks, government, manufacturing, etc.
 - Pascal – introduced ‘structural programming’. Made programs easier to test and debug.



COBOL

PASCAL