



Introductory Python Programming Bootcamp

UAMS Biomedical Informatics

Tuesday, July 08, 2024



Cody Ashby, Ph.D.
Assistant Professor



Jonathan P. Bona, Ph.D.
Assistant Professor



Shaymaa Al-Shukri, Ph.D.
Instructor

This course is designed for first year biomedical informatics graduate students who have little or no prior exposure to computer programming, or who may have minimal programming experience in another language but would like to learn the basics of Python. The course covers materials taught in an introductory undergraduate course in computer science or related disciplines.



Course Webpage

Welcome

This is the landing page for the Summer 2024 UAMS DBMI Python Programming Bootcamp. The schedule of topics below contains links to pages for individual lab sessions.

Textbook

Python Crash Course, 3rd Edition: A Hands-On, Project-Based Introduction to Programming by Eric Matthes.

Please start by skimming Chapter 1, but *do not follow any instructions in Chapter 1 for setting up software*. We will use different software that needs to be configured differently.

Schedule of Topics

Date	Topic	Reading
Thursday, July 11 2024	Introduction, basic concepts	Read Ch 2
Tuesday, July 16 2024	Lists	Read Ch 3
Thursday, July 18 2024	Lists & Loops	Read Ch 4
Tuesday, July 23 2024	Selection statements	Read Ch 5
Thursday, July 25 2024	Dictionaries	Read Ch 6
Tuesday, July 30 2024	While loops & more	Read Ch 7
Thursday, August 1 2024	Functions	Read Ch 8
Tuesday, August 6 2024	Data science libraries & wrap-up	

Grading

Though you are not officially registered in a for-credit course for this bootcamp, we will ask you to submit homework assignments and complete quizzes. These are designed to help us and you to track your progress and assess how much you have learned, and we will grade these with feedback.

- | | |
|-----|--|
| 20% | Quizzes: short, frequent assessments of material you learned in the |
| 10% | Participation: this includes your contributions in-class discussions and group work on in-class exercises |
| 70% | Programming assignments: four to six short programming assignments that you will work on between sessions |



- MS Teams
 - Live sessions
- Pycharm
 - Coding
- Web page & Github
 - Accessing lab materials and assignment descriptions
- UAMS Box
 - Submitting assignments


<https://www.jetbrains.com/pycharm/download/>

← → ↻ 🔍 jetbrains.com/pycharm/download/?section=mac ☆


JETBRAINS PyCharm 🔍 ☰

Download ▾ Pricing


Windows macOS Linux

 **PyCharm** Unified Product

The only Python IDE you need

 Download .dmg (Apple Sili... ▾

Free forever, plus one month of Pro included

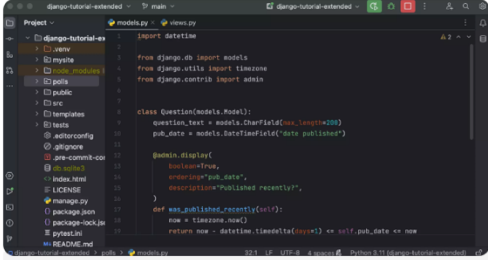
 Select an installer for Intel or Apple Silicon

PyCharm is now one unified product!

All users now automatically start with a free one-month Pro trial. After that, you can subscribe to Pro or keep using the core features for free – now with Jupyter support included.

Version: 2025.1.3
Build: 251.26927.74
2 July 2025

System requirements	Other versions
Installation instructions	Third-party software



← → ↻ 🔍 jetbrains.com/pycharm/download/?section=mac ☆

Download ▾ Pricing

unified PyCharm!

Switch to the unified PyCharm and get all core Community features for free, now with built-in Jupyter support.

You can upgrade to PyCharm Community 2025.1 as usual – no immediate changes are necessary. A seamless migration will follow in the next release. Either way, you keep everything and get more.

[Learn more](#)



PyCharm Community Edition

The IDE for Pure Python Development

Download .dmg (Apple Silicon) ▾

Free, built on open source

A HANDS-ON, PROJECT-BASED
INTRODUCTION TO PROGRAMMING

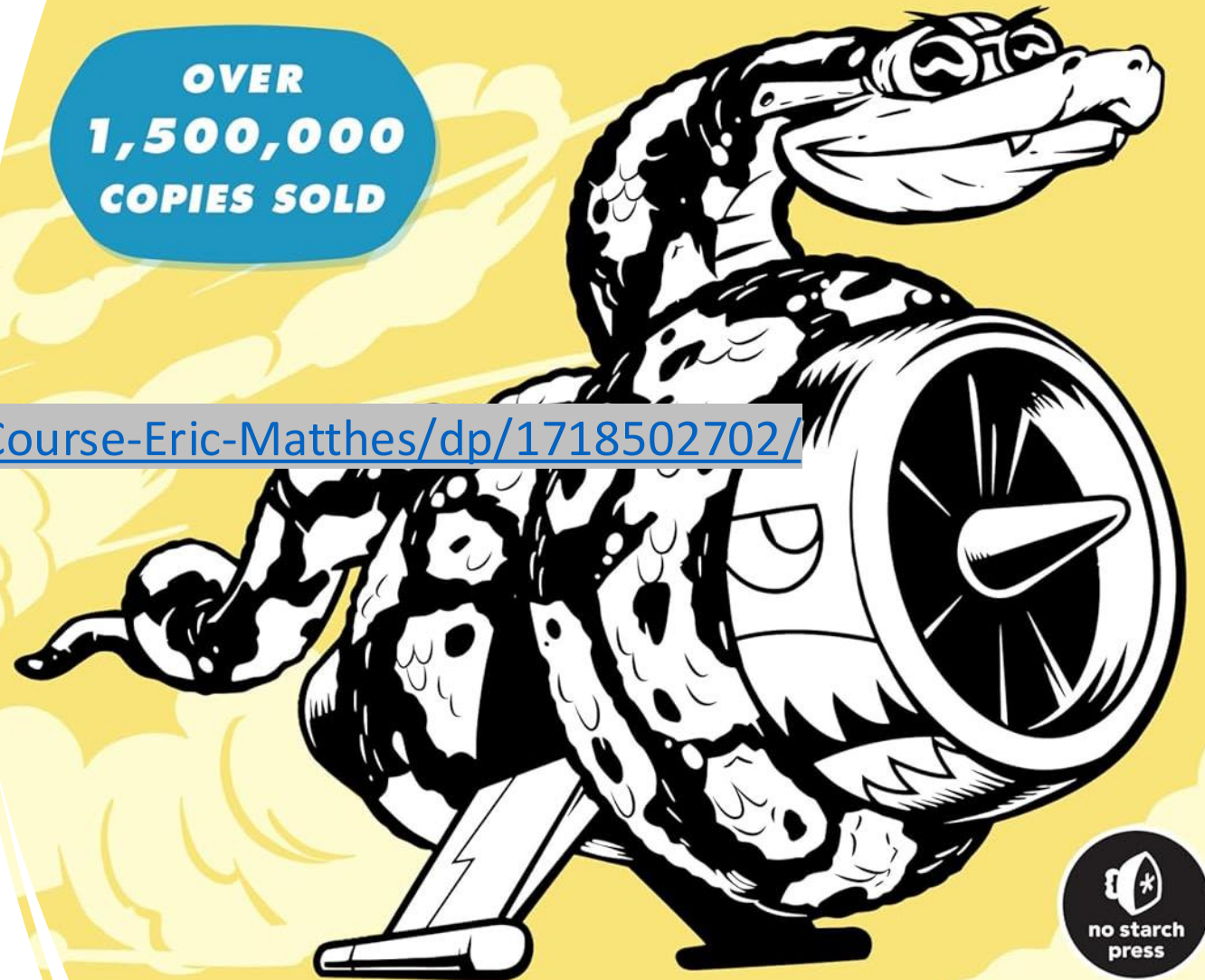
ERIC MATTHES

OVER
1,500,000
COPIES SOLD

Python Crash Course, 3rd Edition

by Eric Matthes

<https://www.amazon.com/Python-Crash-Course-Eric-Matthes/dp/1718502702/>

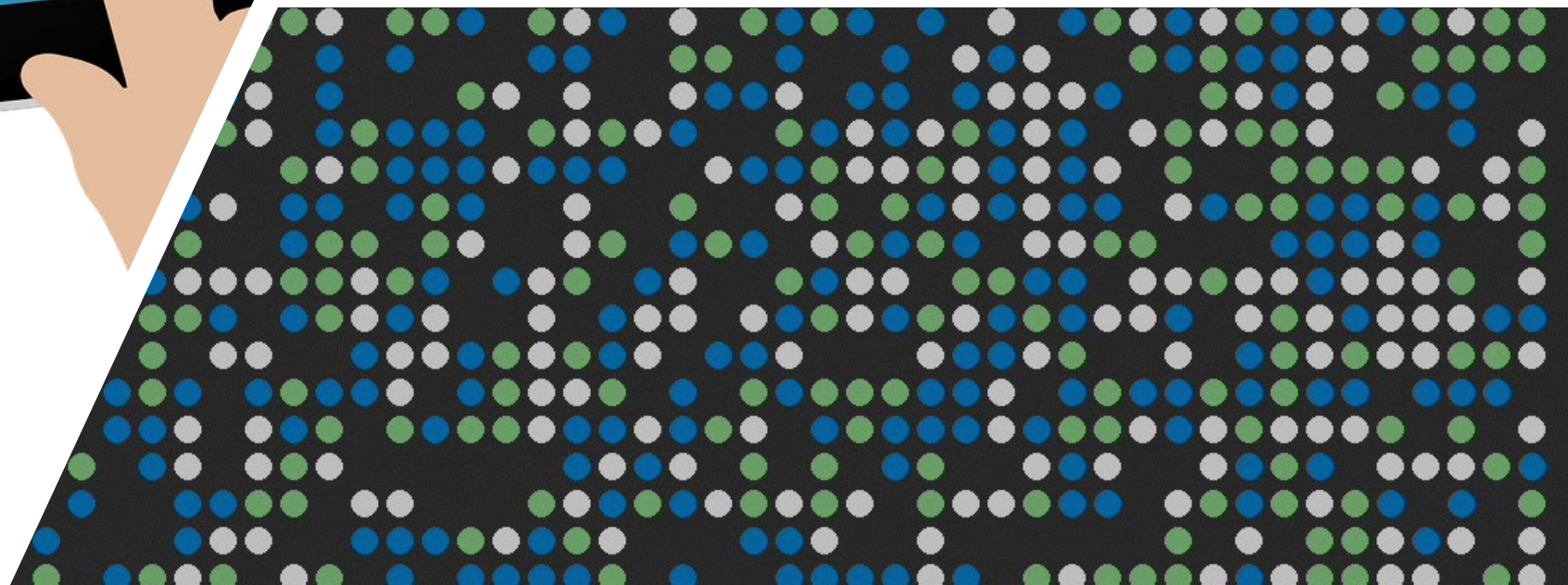
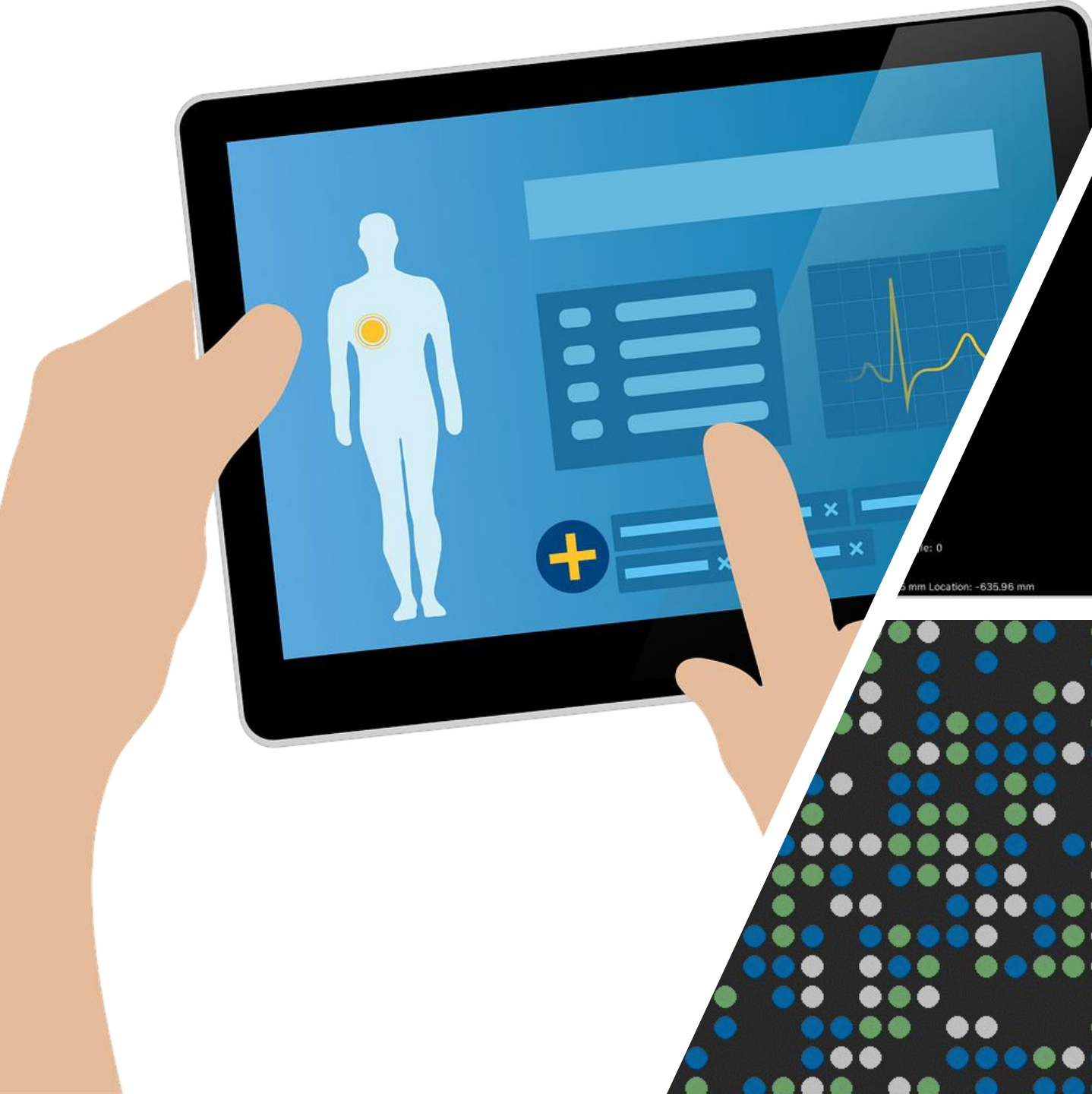


What about ChatGPT?

You may not use ChatGPT or other Generative AI tools for any purpose on assignments (quizzes, programming assignments, in-class assignments) in this course.

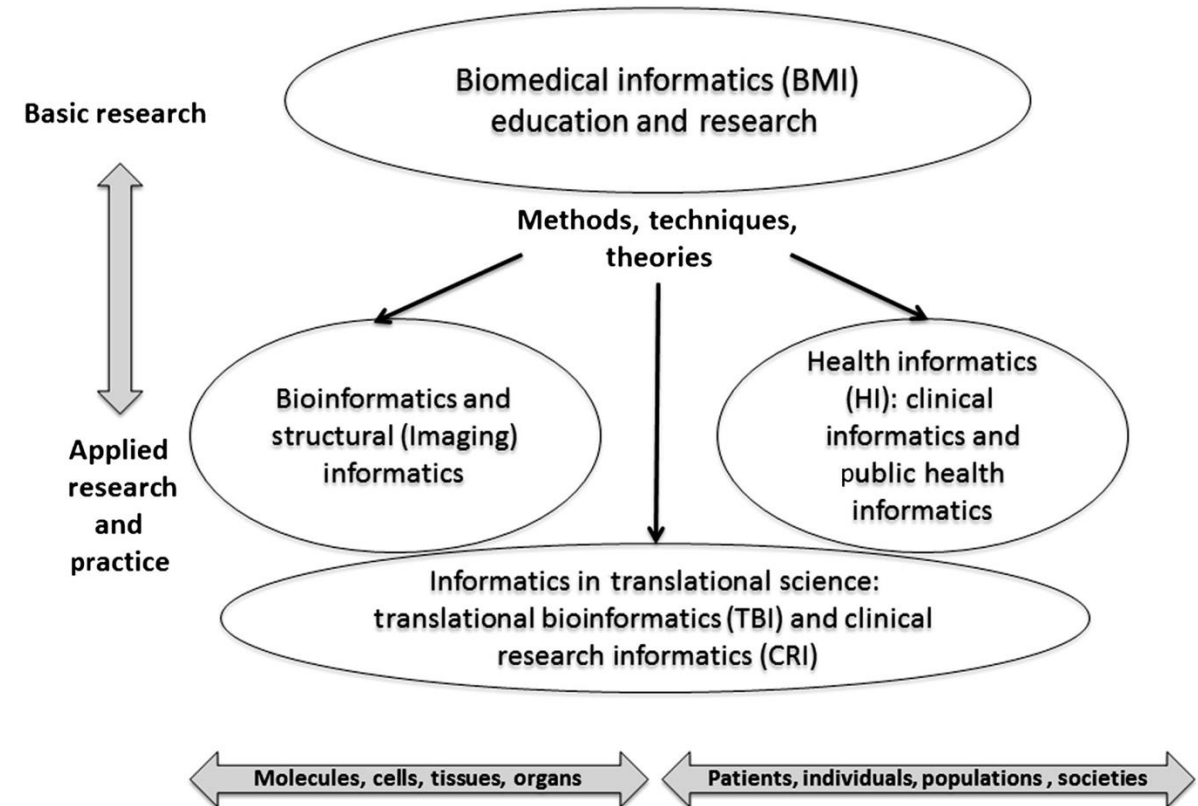
IV. The Use of GAI by Trainees and Teaching Faculty

- a. Use of GAI by students to satisfy course requirements must be authorized by the course faculty, trainer, or education supervisor.
- b. GAI tools can be useful assistive devices for learning. Trainees should use GAI tools wisely and intelligently, aiming to deepen understanding of subject matter and to support learning. Unless otherwise directed or specified by an instructor or mentor, helpful ways to use GAI include analysis, rephrasing, essentializing, synthesizing, and/or gathering information about the typical understanding of a topic to assist with learning. However, it must be the trainee who guides, verifies, and crafts the final answers on class assignments.
- c. Different courses and instructors will have different policies regarding the use of GAI tools and services for academic purposes. It is the trainee's responsibility to follow the GAI policies for each of the courses in which they are enrolled. Please ask the instructor any questions prior to utilizing a GAI tool for any assignment.
- d. Violating this policy or the instructor's directions about the use of GAI in their course or on an assignment might be considered a violation of the College's Academic Misconduct Policy and/or Academic Integrity Policy and could include disciplinary action up to and including dismissal.
- e. If the instructor allows trainees to utilize GAI in creating an assignment, please refer to the section above on how to appropriately cite and describe its use.



Biomedical Informatics

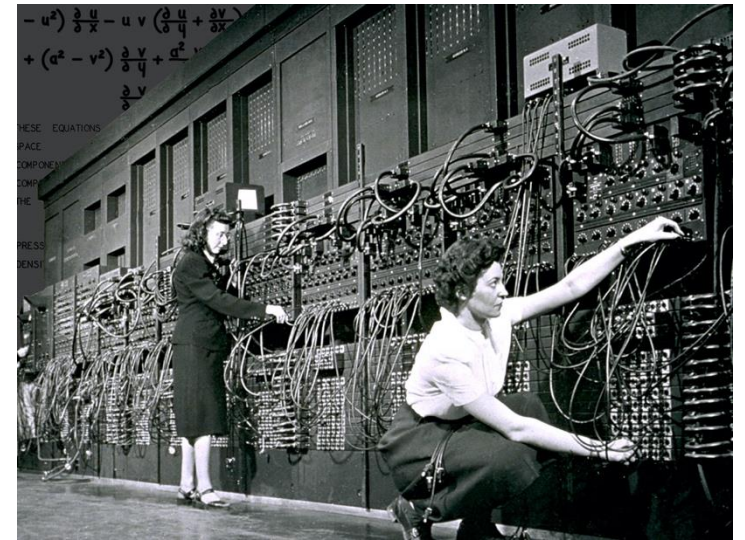
Biomedical informatics (BMI) is the interdisciplinary field that studies and pursues the effective uses of biomedical data, information, and knowledge for scientific inquiry, problem solving, and decision making, driven by efforts to improve human health.



Kulikowski CA, Shortliffe EH, Currie LM, Elkin PL, Hunter LE, Johnson TR, Kalet IJ, Lenert LA, Musen MA, Ozbolt JG, Smith JW. AMIA Board white paper: definition of biomedical informatics and specification of core competencies for graduate education in the discipline. Journal of the American Medical Informatics Association. 2012 Nov 1;19(6):931-8.

Programming Languages

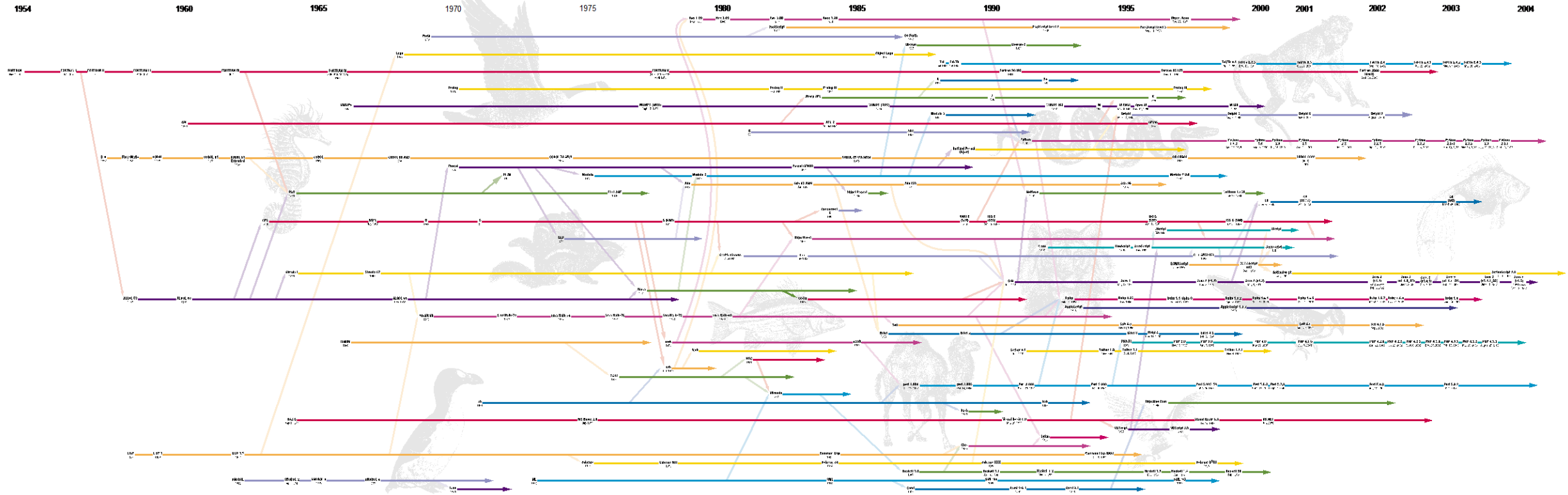
- Computer programming is the activity of expressing *algorithms* in a language that a computer can understand.
- A *programming language* is what we use to communicate these instructions to the computer.
- Ultimately, the computer converts instructions to its *machine code*.
 - Sequences of bits – easy for computers, hard for humans



<https://en.wikipedia.org/wiki/ENIAC>

History of Programming Languages

O'REILLY®



www.oreilly.com

For more than half of the fifty years computer programmers have been writing code, O'Reilly has provided developers with comprehensive, in-depth technical information. We've kept pace with rapidly changing technologies as new languages have emerged, developed, and mutated. Whether you want to learn something new or need answers to tough technical questions, you'll find what you need in O'Reilly books and on the O'Reilly Network.

This timeline includes fifty of the more than 2,500 documented programming languages. It is based on an original diagram created by Ericnieveez (www.levinez.com), augmented with suggestions from O'Reilly authors, friends, and conference attendees. For information and discussion on this poster, go to www.oreilly.com/go/languageposter.



©2004 O'Reilly books, Inc. O'Reilly logo is a registered trademark of O'Reilly books, Inc. All other trademarks are the property of their respective owners. pdf00147

http://oreilly.com/news/graphics/prog_lang_poster.pdf

What is an Algorithm?

- “... any well-defined computational procedure that takes some value, or set of values, as ***input*** and produces some value, or set of values, as ***output***. An algorithm is thus a sequence of computational steps that transform the input into the output.”

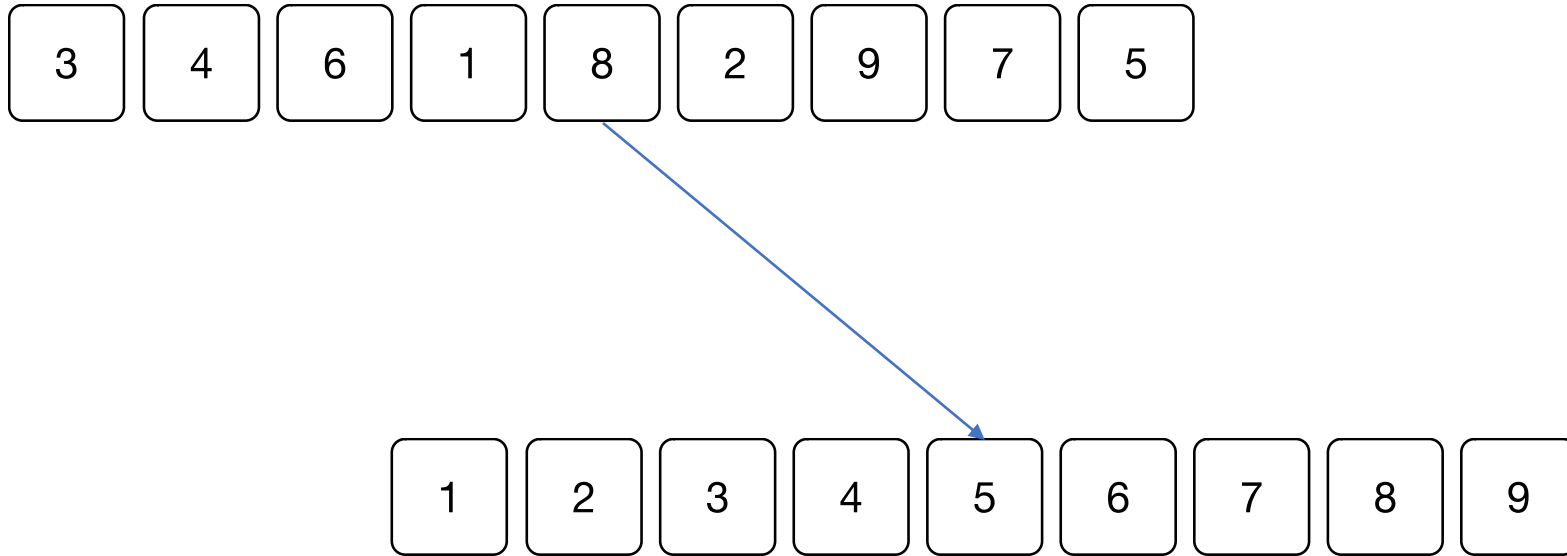
Introduction to Algorithms, Second Edition by Thomas H. Cormen, Charles E. Leiserson, Ronald L. Rivest and Clifford Stein

Algorithm Exercise: Sorting

- Problem: given a sequence of numbers in any order, re-order those numbers so the sequence is in ascending order.
- Input: a sequence of numbers
- Output: a sorted list of the numbers



Algorithm Exercise: Sorting



Some basic operations

- Go to the beginning of a sequence
- Look at the value stored in a particular position in a sequence
- Compare elements stored at two different positions in a sequence
- Make a choice based on the result of a comparison
- Output an answer like "yes" or "no"
- Remove an element from a particular place in a sequence
- Copy an element from one place to another
- Swap the position of two elements

is_sorted? Checking whether a sequence of numbers is sorted in ascending order

8 67 12 9 877 6

is_sorted? Checking whether a sequence of numbers is sorted in ascending order

- Go to the beginning of the sequence
- Compare the first number in the sequence to the second number in the sequence
 - If the first number is greater than the second, then the sequence is not in order, so stop and answer “NO”
- Compare the second number in the sequence to the third number in the sequence
 - If the second number is greater than the third, then the sequence is not in order, so stop and answer “NO”
- ...
- Compare the second-to-last number in the sequence to the last number in the sequence
 - If the second-to-last number is greater than the last number, then the sequence is not in order, so stop and answer “NO”
- If we didn’t answer no yet, then that means we looked at every pair of numbers in the sequence and no pair was out of order. That means the entire sequence is *in order*, so we stop and answer “YES”

sort_sequence: input a sequence, output a sorted version of that sequence

8 67 12 9 877 6

A Sorting Algorithm

- Go to the beginning of the sequence
- Compare the first number in the sequence to the second number in the sequence
 - If the first number is greater than the second, swap the two numbers.
- Compare the second number in the sequence to the third number in the sequence
 - If the second number is greater than the third, swap the two numbers.
- ...
- Compare the second-to-last number in the sequence to the last number in the sequence
 - If the second-to-last number is greater than the last number, swap the two numbers.
- Is the sequence sorted? If not, go back to the first step and repeat these instructions.