Applied Algorithms

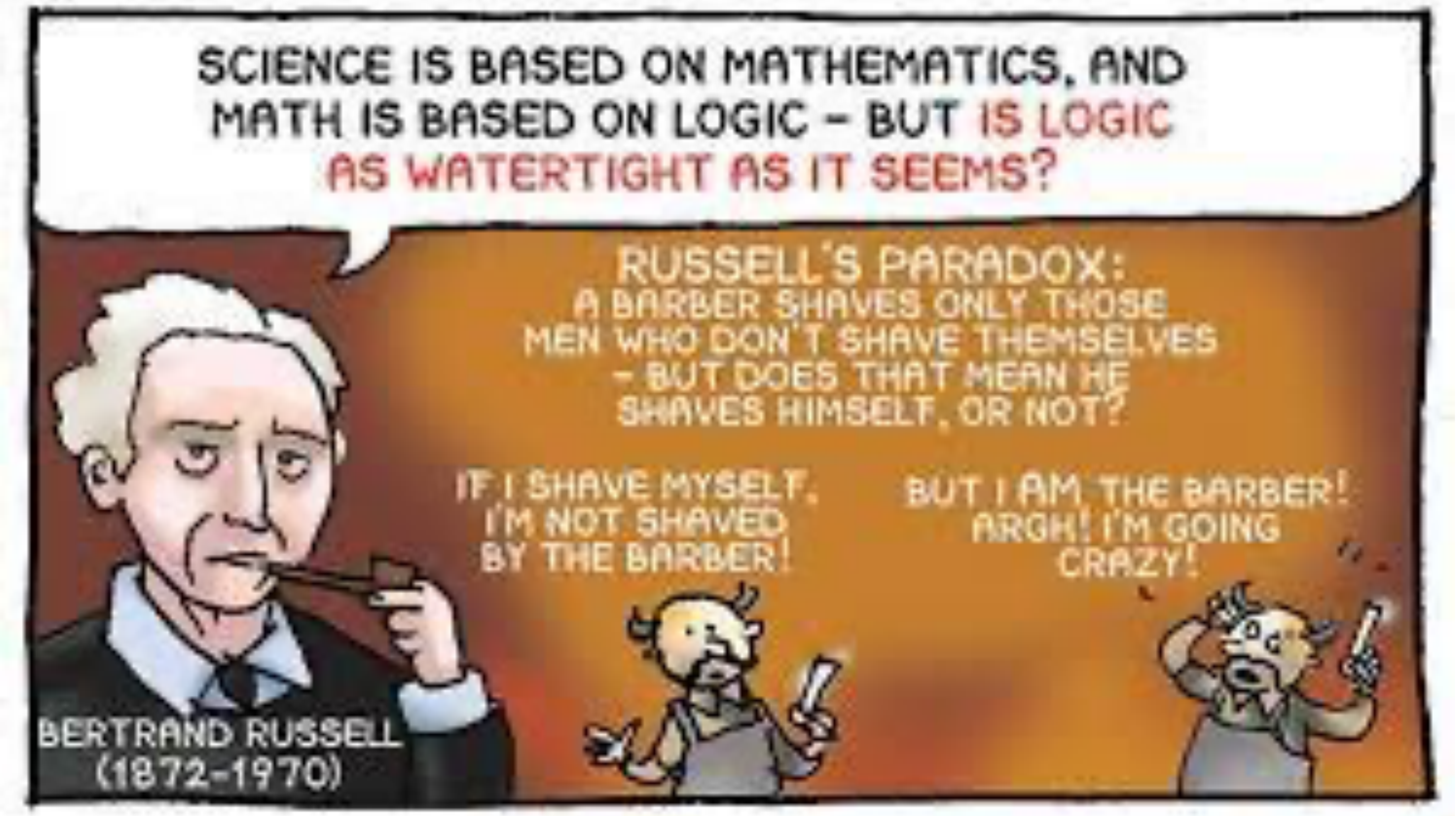
CSCI-B505 / INFO-I500

Lecture 1.

Introduction to Applied Algorithms and Course Overview

M. Oğuzhan Kulekci, August 22/23, 2022

THE QUESTION THAT LED TO THE DEVELOPMENT OF COMPUTING MACHINES ! *(I.M.O.)*

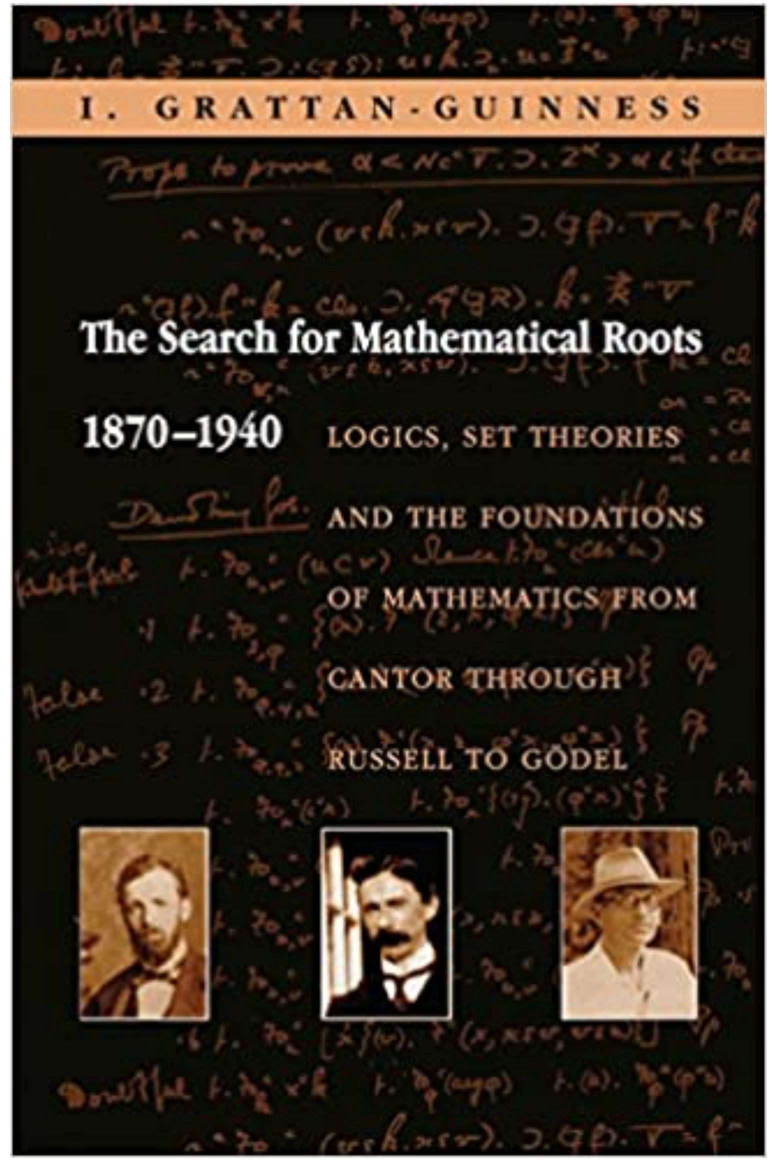


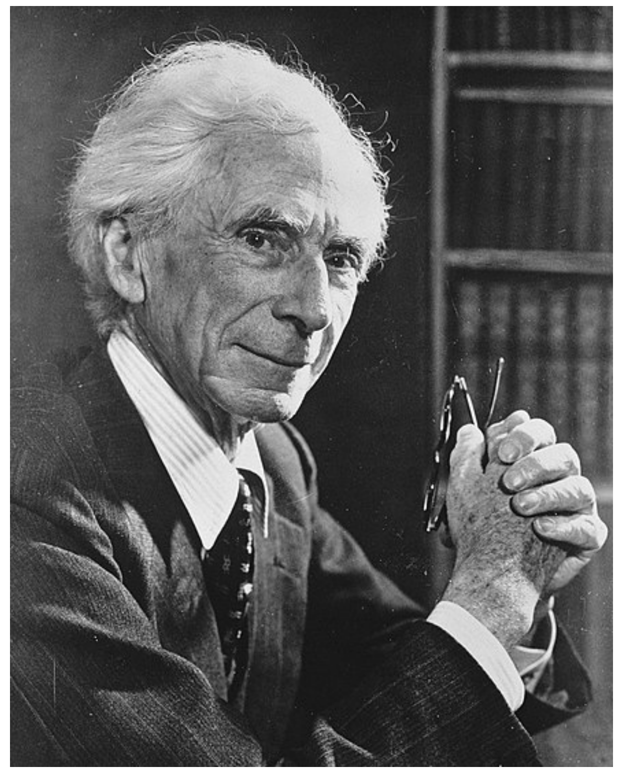
Mathematical Roots of Computing 1870 - 1940

Georg Cantor
1845-1917

SET THEORY - 1874

David Hilbert
1862-1943
MATHEMATICAL LOGIC - 1921



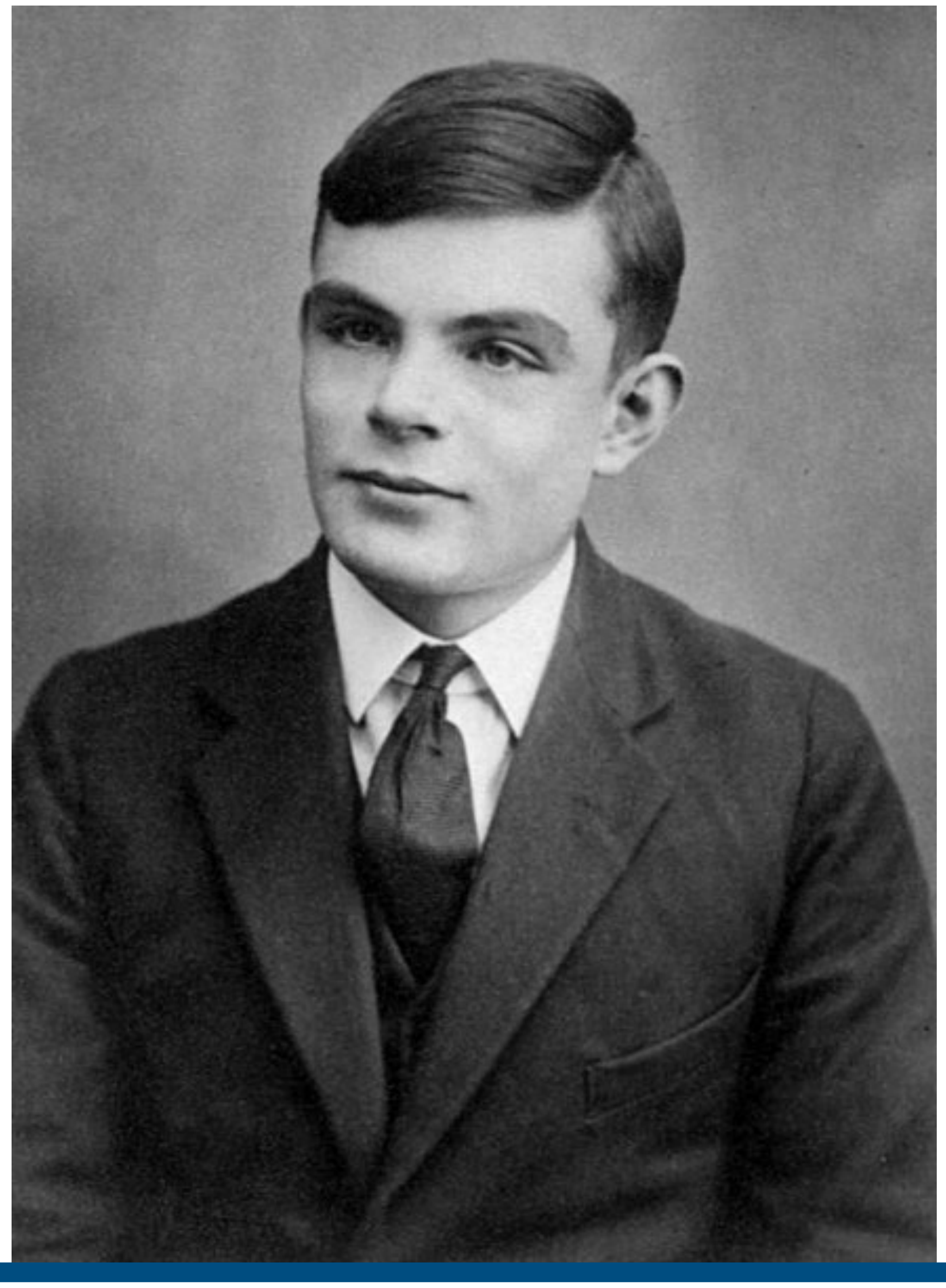


Bertrand Russell
1872-1970

RUSSEL PARADOX - 1901

Kurt Gödel
1906 - 1978

Incompleteness - 1931



Alan Turing
1912-1954

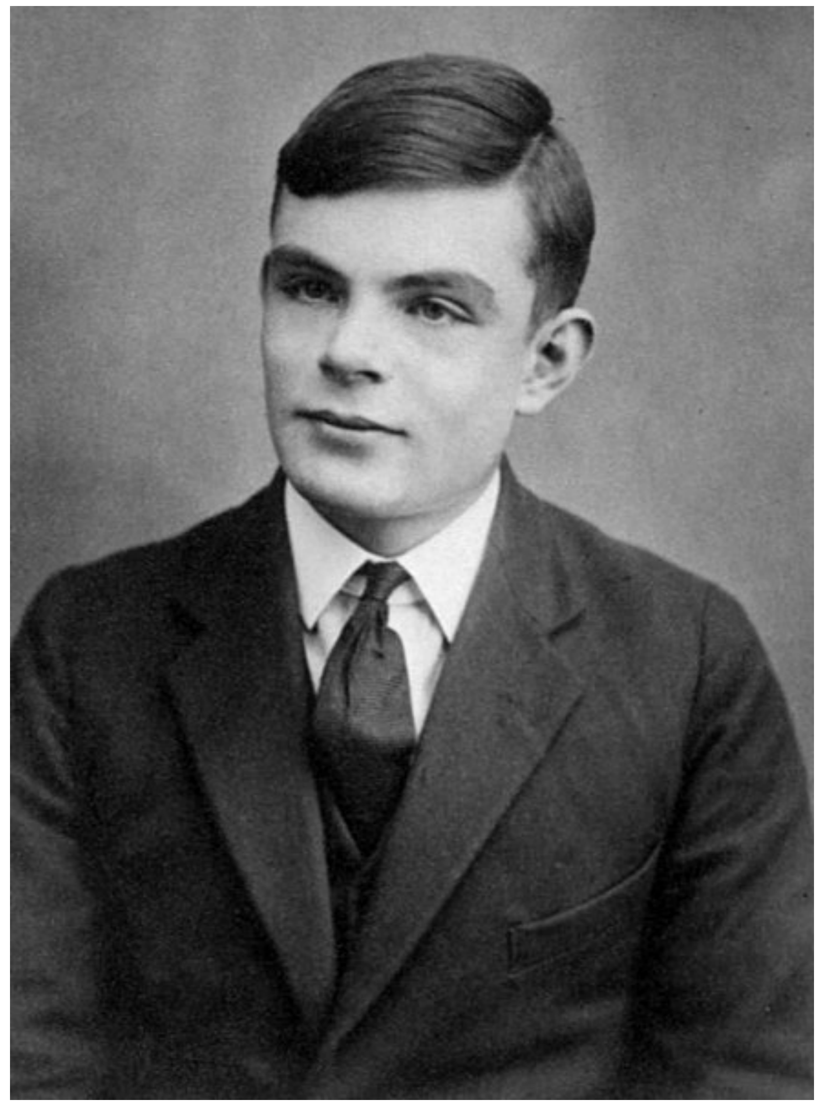
Undecidability - 1936

FIRST PROGRAMMABLE COMPUTING MACHINES 1940 - 1950

Alanzo Church
1903 - 1995

$\lambda - calculus$

1936

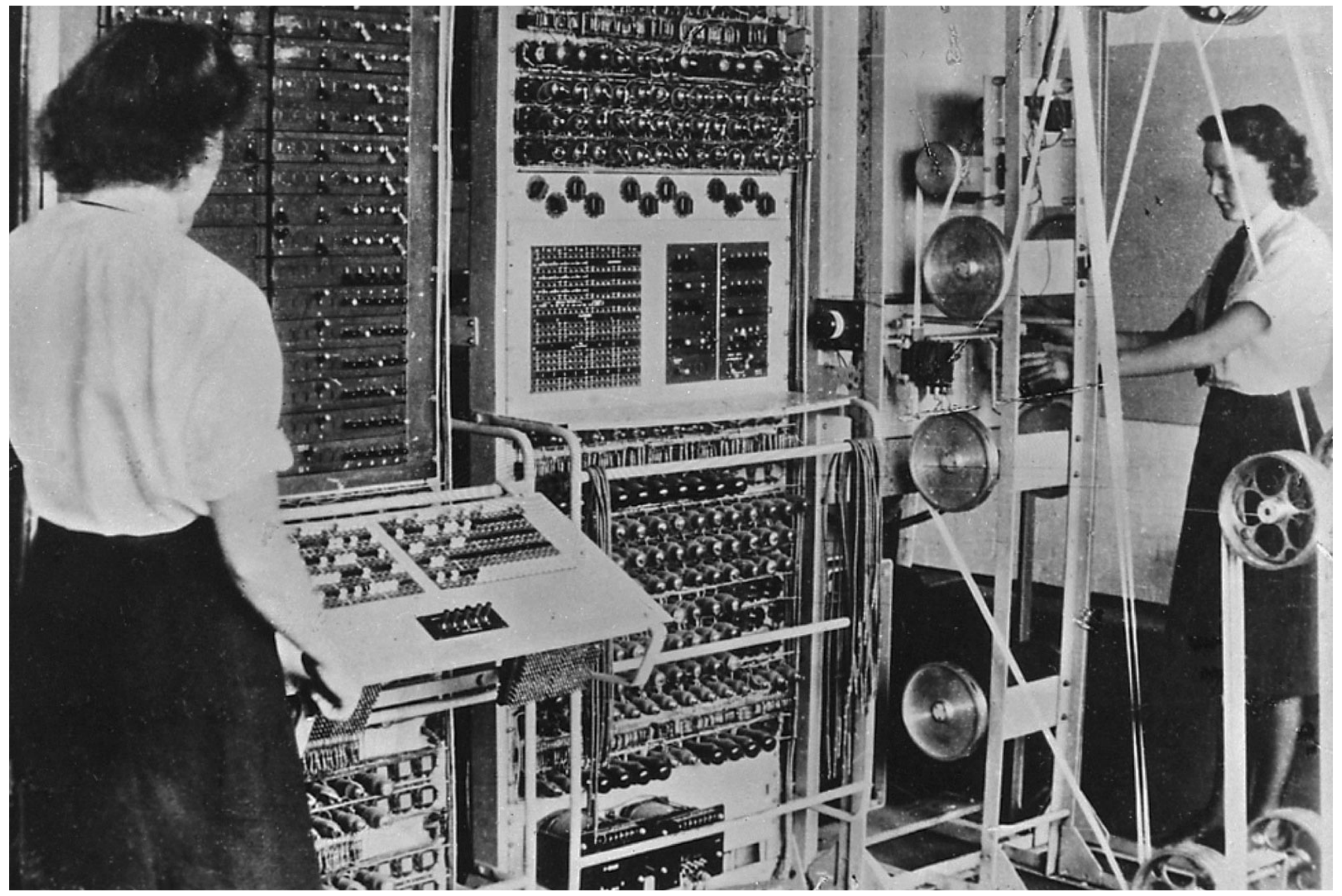


Alan Turing
1912-1954

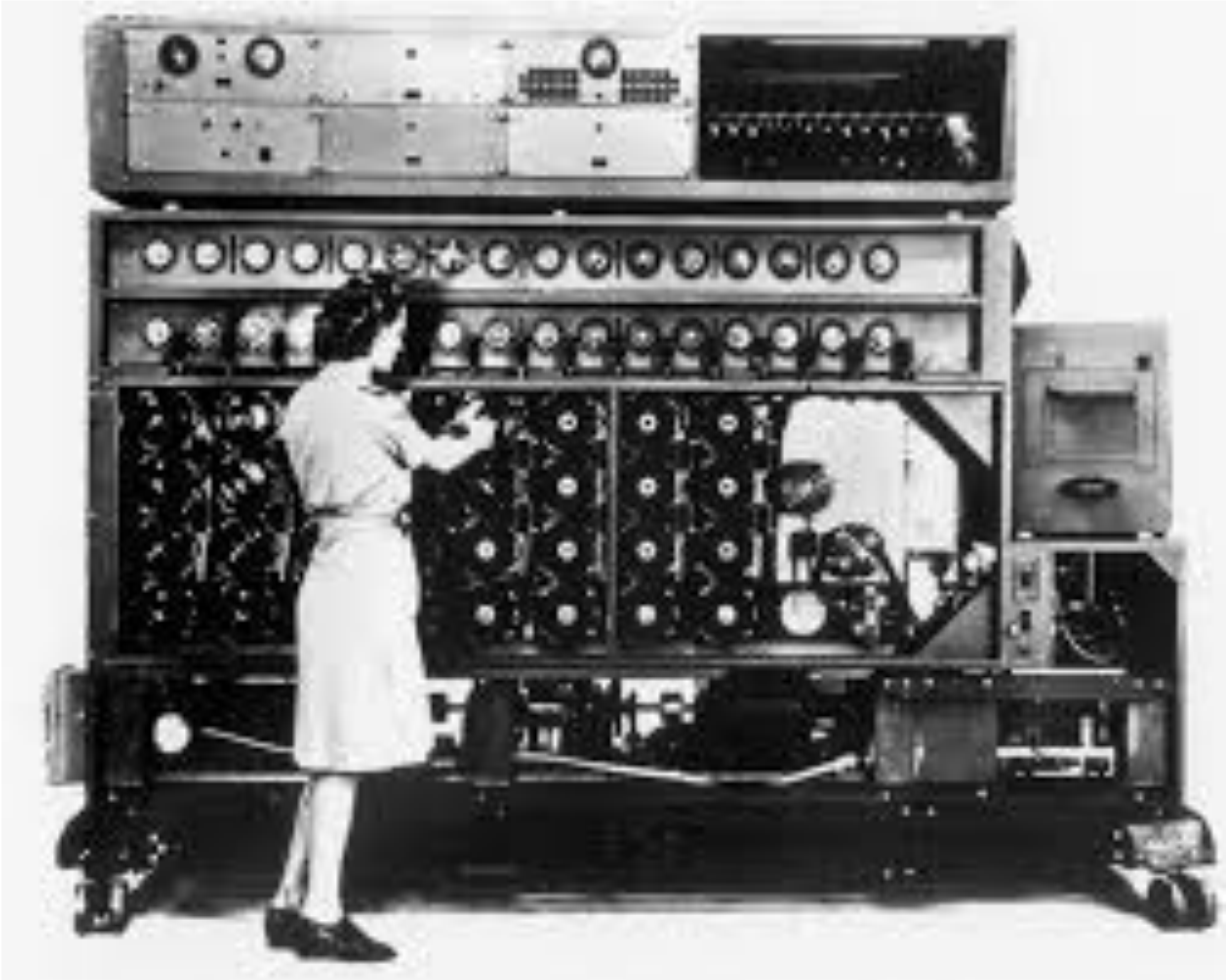
$Turing\ \ Machine$

1936

Church-Turing Thesis:
A function on the natural numbers can be calculated by an effective method if and only if it is computable by a Turing machine



COLOSSUS



BOMBE

1942 - 1945
BLETCHLEY PARK - LONDON, UK

ENIGMA
II. WORLD WAR
CRYPTO MACHINE

EARLY YEARS 1950 - 1960

UNIVAC 1103 (1953)

Computers appeared as new commercial devices, and ignited the “computing business”, (have a look to https://en.wikipedia.org/wiki/Timeline_of_computing for more details…)

Computer Programming:

Machine Codes → Assembly Language → Modern programming languages FORTRAN’54, ALGOL’60, *(later PASCAL’70, C’73, C++’82, JAVA’91, PYTHON, …)*

TOWARDS A NEW SCIENTIFIC DISCIPLINE (1960 - 1970)

ON THE COMPUTATIONAL COMPLEXITY OF ALGORITHMS

BY

J. HARTMANIS AND R. E. STEARNS

I. **Introduction.** In his celebrated paper [1], A. M. Turing investigated the computability of sequences (functions) by mechanical procedures and showed that the set of sequences can be partitioned into computable and noncomputable sequences. One finds, however, that some computable sequences are very easy to compute whereas other computable sequences seem to have an inherent complexity that makes them difficult to compute. In this paper, we investigate a scheme of classifying sequences according to how hard they are to compute. This scheme puts a rich structure on the computable sequences and a variety of theorems are established. Furthermore, this scheme can be generalized to classify numbers, functions, or recognition problems according to their computational complexity.

Hartmanis & Stearn'1965: Time and space complexity analysis of algorithms

PATHS, TREES, AND FLOWERS

JACK EDMONDS

2. Digression. An explanation is due on the use of the words "efficient algorithm." First, what I present is a conceptual description of an algorithm and not a particular formalized algorithm or "code."

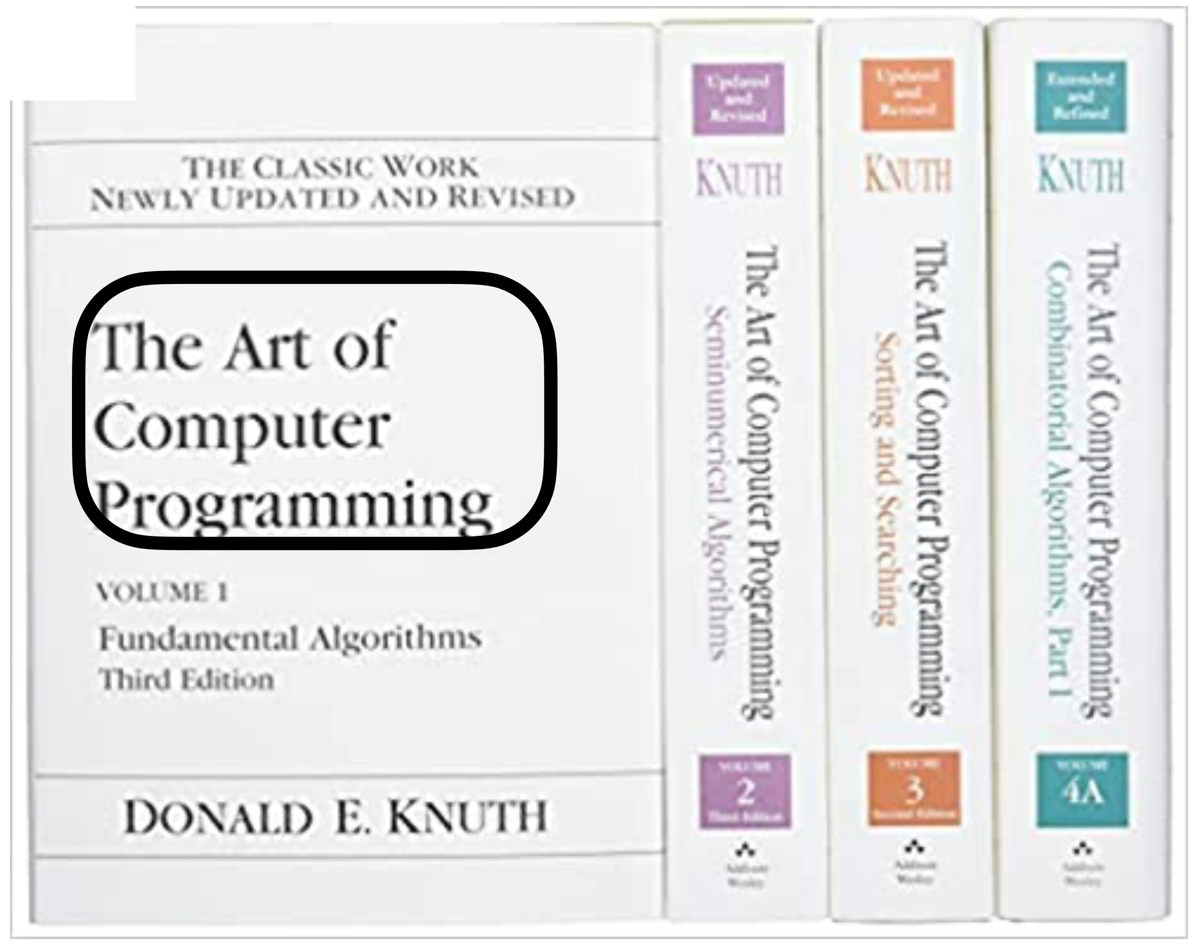
For practical purposes computational details are vital. However, my purpose is only to show as attractively as I can that there is an efficient algorithm. According to the dictionary, "efficient" means "adequate in operation or performance." This is roughly the meaning I want—in the sense that it is conceivable for maximum matching to have no efficient algorithm. Perhaps a better word is "good."

I am claiming, as a mathematical result, the existence of a *good* algorithm for finding a maximum cardinality matching in a graph.

There is an obvious finite algorithm, but that algorithm increases in difficulty exponentially with the size of the graph. It is by no means obvious whether *or not* there exists an algorithm whose difficulty increases only algebraically with the size of the graph.

The mathematical significance of this paper rests largely on the assumption that the two preceding sentences have mathematical meaning. I am not prepared to set up the machinery necessary to give them formal meaning, nor

Edmunds'65: Definition of Efficient Algorithm



1962(start)-1971

How can we measure the performance of an algorithm?

When does an algorithm is assumed to be "efficient" ?

Elegance in computing ?

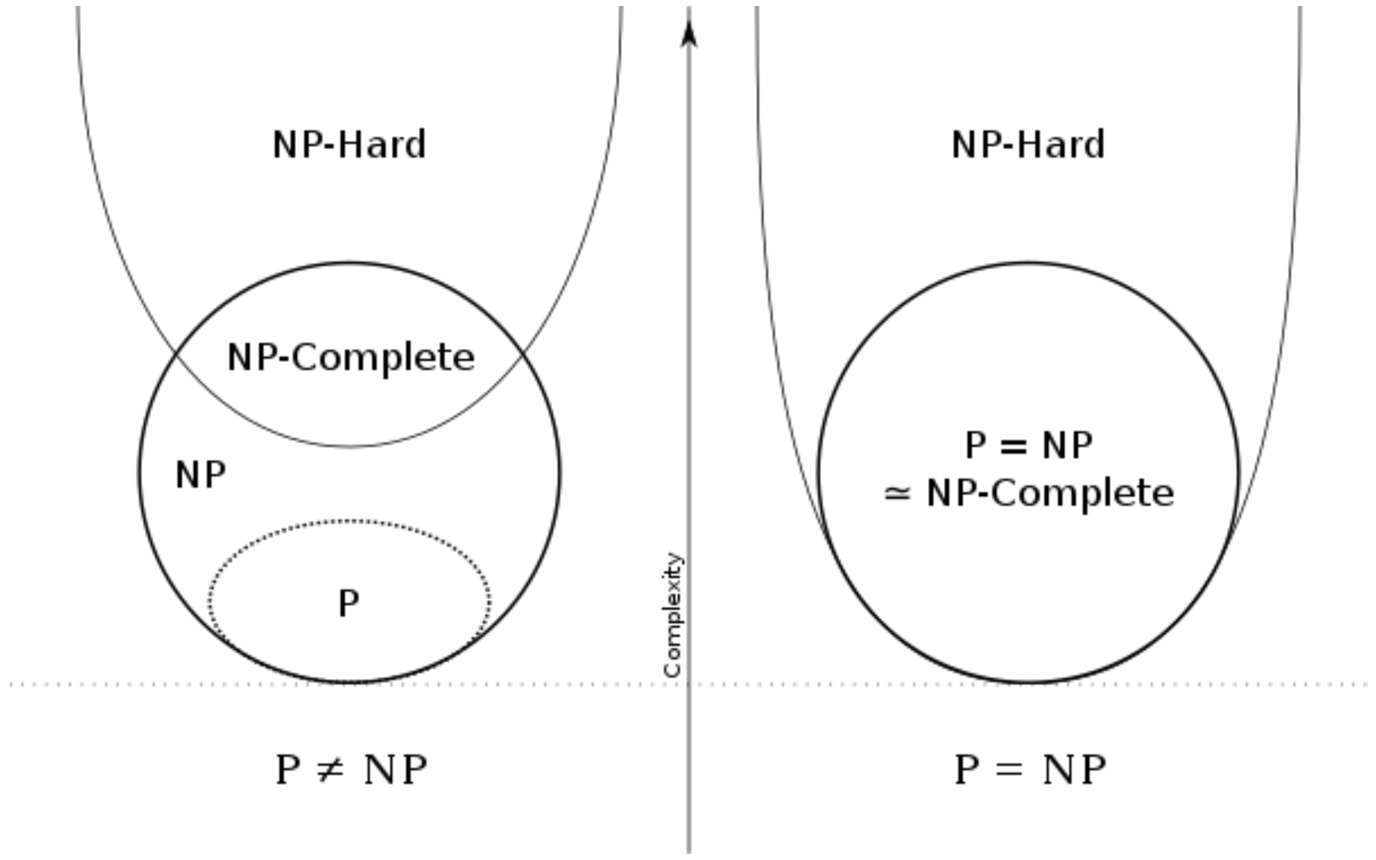
***Elegance** is beauty that shows unusual **effectiveness** and **simplicity**….*

PEN & PAPER ERA IN ALGORITHMS & THEORY (up to late 80s)

Stephan Cook
1971 (3-SAT)

Leonid Levin
1971

Richard Karp
1972
21 new hard problems



Some computational problems are really hard, with amazing relations between them

Developments in many sub-divisions of computer science: **parallel algorithms**, **external memory**, **randomization**, **approximation**, etc….

GAP BETWEEN THEORY AND PRACTICE (up to late 90s)

! Personal computers !
Everyone can purchase.

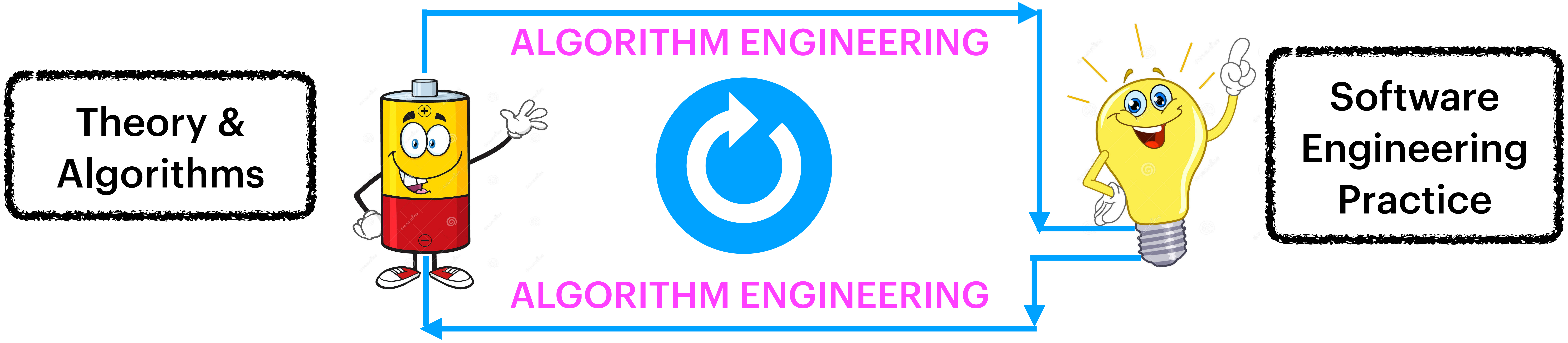
Software industry developing very fast ! More and more digitization in daily life…

- Theory and practice were flowing in different rivers.
- Practice was not in big need for theory, and theory was not much interested in practice.
- However, the gap between theory and practice became apparent.

ALGORITHM ENGINEERING PARADIGM (2000 - ...)

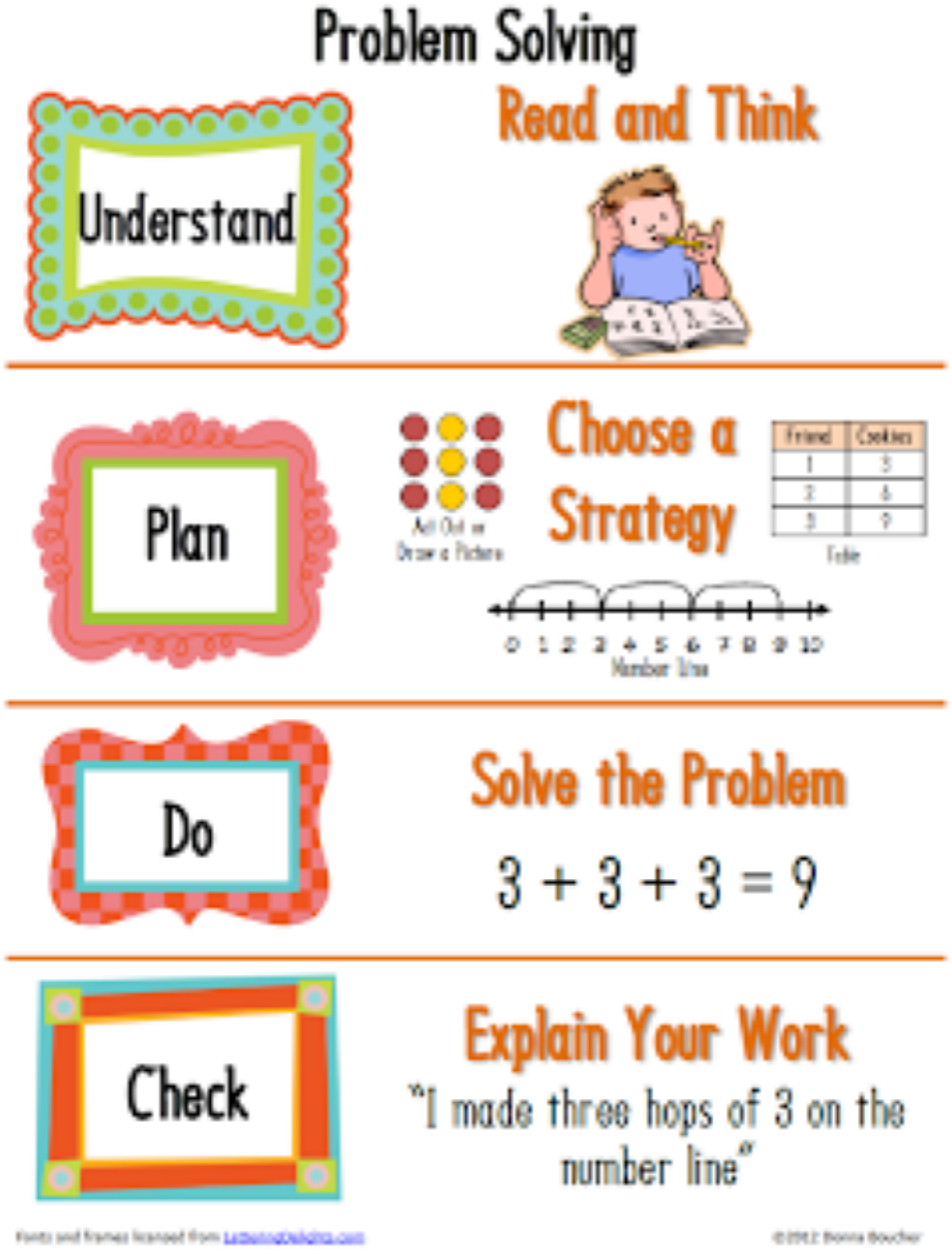
Previous hypotheses , which were assumed to be unrealistic, are turning into realities due to :

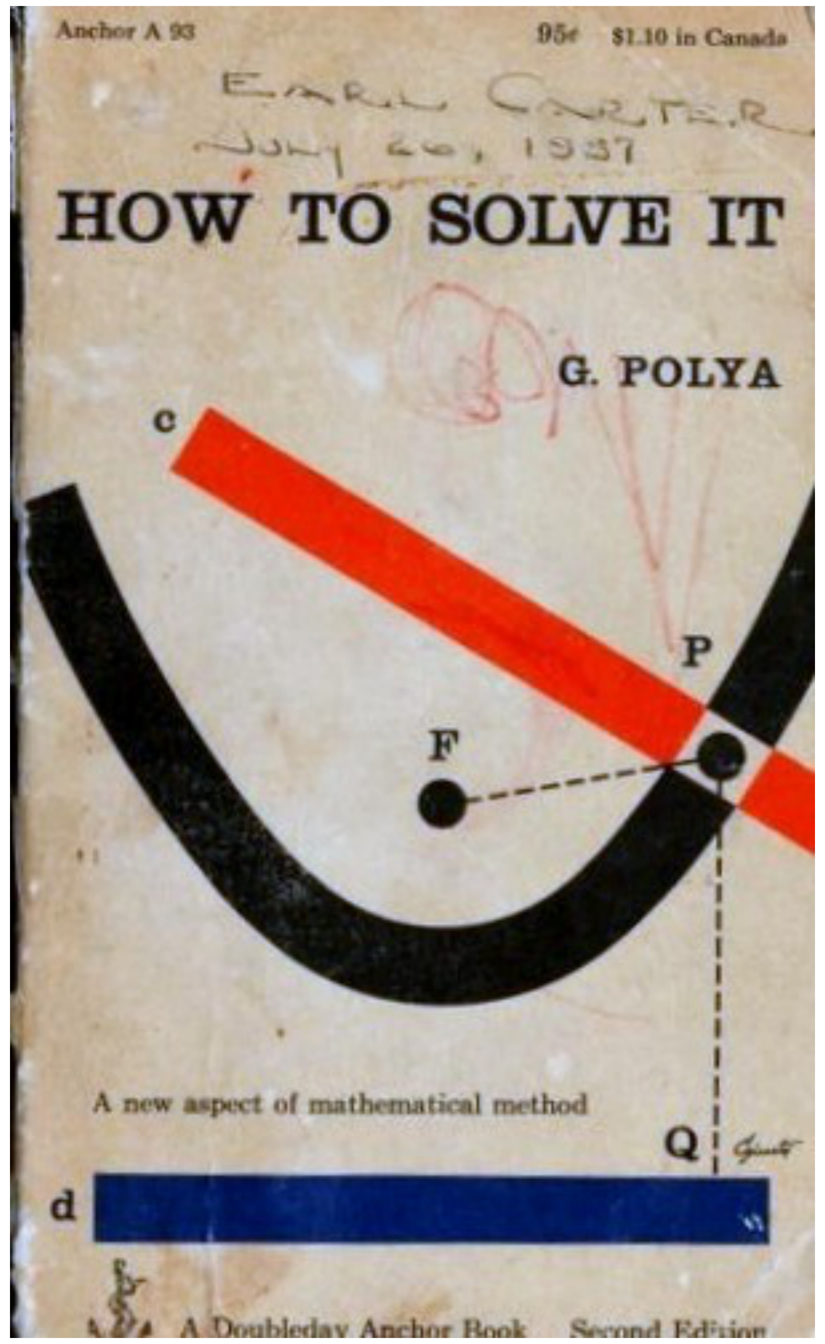
- Data deluge and ever increasing digitization
- Advances on computing platforms and processor technology
- Increasing importance of theoretical results in practical applications



Sound understanding of algorithms and data structures is essential for devising efficient solutions of computational challenges.

Is it sufficient ? (Another story)





The Aim of This Course

To improve our algorithm design and analysis skills and increase our awareness on integrating them into applications.

IMPORTANT NOTICE: We assume

- **you know the basic data structures and algorithms at the undergraduate level**
- **you know how to program in Python on Jupyter Notebook environment**
- **you have the basic probability and discrete math knowledge at the undergraduate level**

This course does not aim to teach any programming.

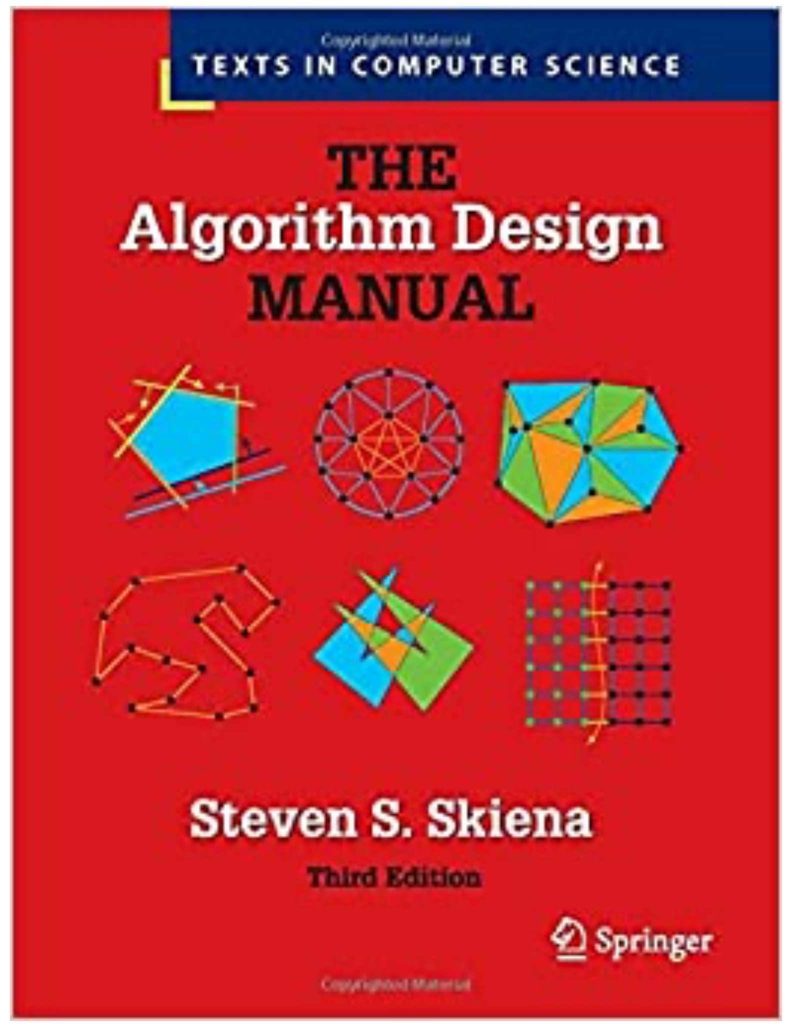
We will be doing programming a lot during this course,

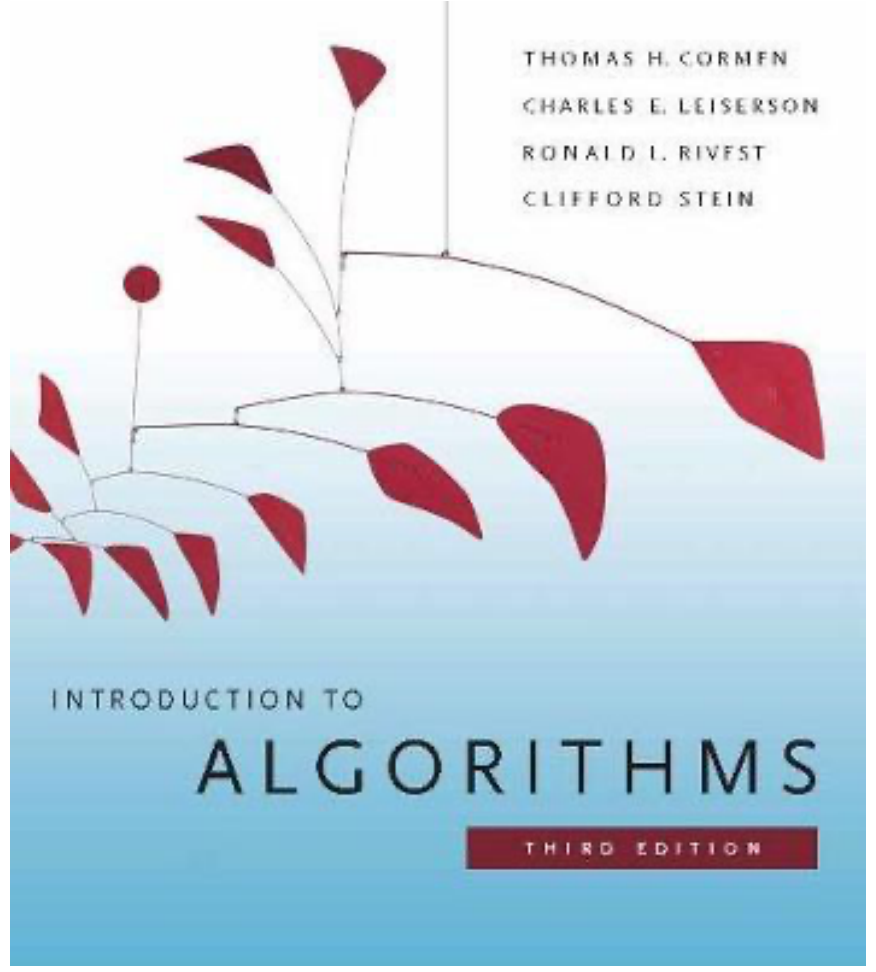
but do not have a special purpose to teach it.

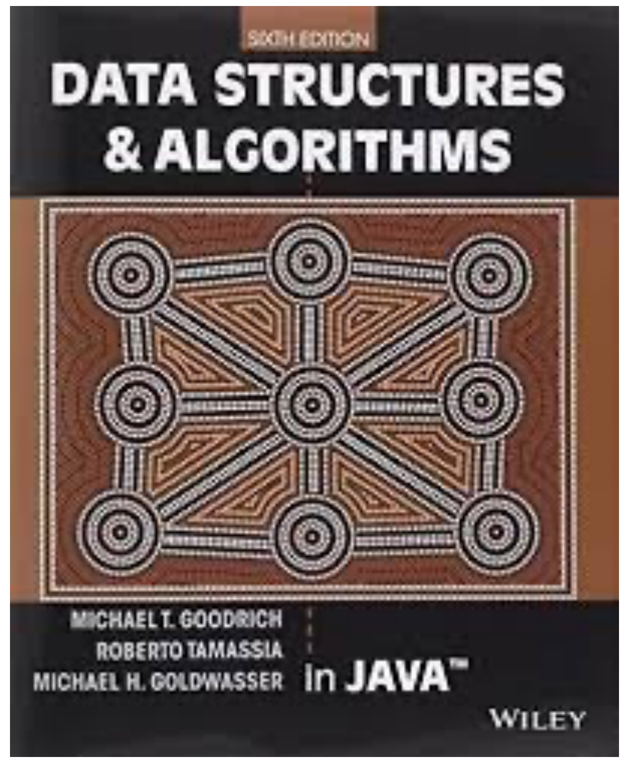
Textbooks & Learning Materials

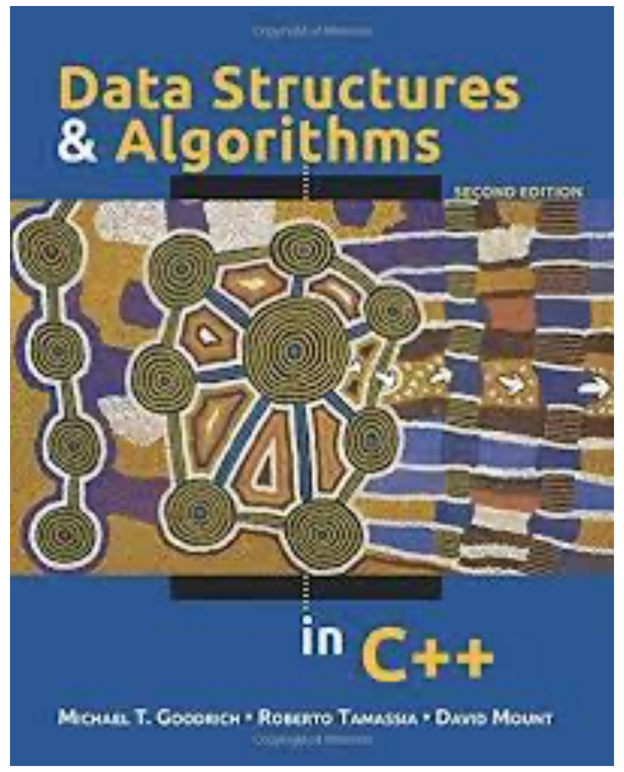
This course does not aim to teach any programming,

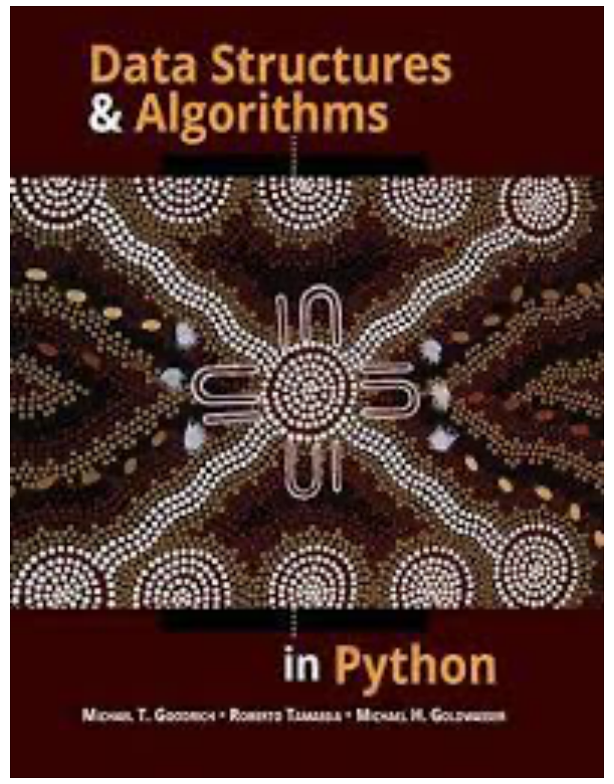
but to improve your skills in algorithms, particularly on real-life scenarios !











There are many great algorithms textbooks you can refer to, but particularly I would like to mention the following

- **The Algorithm Design Manual, 3rd Edition by Steven Skiena**
- **Introduction to Algorithms, Cormen et al.**
- **Data Structuctures and Algorithms in Java / Python / C/C++, Goodrich et al.**
- **Papers and other reading materials will be provided during the course**

Course Organization

Principal Instructor:

M. Oğuzhan Külekci
Email: okulekci@iu.edu
Office: Luddy - 2030
Office Hours:
Monday, 11:00am-1:00pm
Wednesday, 11:00am-1:00pm

Head of Associate Instructurs

Ravichandra Pothamsetty
E-mail: rapotham@iu.edu

Any questions/concerns about the course should be addressed to Ravi first.

Course Organization

- **PLEASE FOLLOW YOUR REGISTERED SECTION.**
    - **We have 11 LAB sections. BUT, each student is registered to a specific LAB session and you can only attend to your section, NOT OTHERS!**
    - **There are two sections of Applied Algorithms. Please join your section to follow the lectures, NOT the other one !**
    - **During all evaluations, you will be attending the section which you are registered to, NOT the other section. NEVER !**
- **This is a huge course with 300+ students. Please consider this fact on all correspondences we will have during the term.**

Course Organization

- **OFFICE Hours:**

  Monday 1pm - 5pm
  Tuesday 9am - 1pm
  Wednesday 1pm - 5pm
  Thursday 9am - 1pm
  Friday 11am - 1pm

Office hours will be held at Luddy 2014.

This is the discussion area at the second floor of Luddy.
Please go and check this place.
The AIs will be waiting you to help on your questions an the mentioned times.

Some Notes on Course Execution

- Almost <u>every week</u> there will be a homework assignment, which will require pen&paper work or programming or **<u>mostly BOTH !</u>**
- Each week, homework assignments will be announced at 5:00 pm., Friday and the submissions will be due to the following week 4:59 pm, Friday.
- You will be doing programming exercises during the LAB sessions, that aims to improve your understanding via examples. Attend the LAB, put your effort to accomplish the task.
- You can visit the AIs on the specified office hours without any prior appointment.
- During all communications, please remember we have around 300+ students !
- We will be using a question/answering platform for better communication. You will receive the updates.
- No bargaining on the grades !

Grading

- Homeworks %50
    - Your average will be computed by **excluding your worst score**
    - Late submissions **will not be accepted** regardless of the causes
    - **Python** will be used for the programming exercises
- Midterm %20
- Final %20
- LAB performance %10

- PLAGIARISM IS STRICTLY FORBIDDEN AND WILL BE FOLLOWED WITH AUTOMATED TOOLS.
- PAY ATTENTION TO DELIVER YOUR DUTIES INDIVIDUALLY.
- UNIQUENESS IN THE SUBMITTED HWs WILL BE CONSIDERED AS CRITERIA IN THE GRADING.

Tentative Schedule : Part-1

	Week	Lecture 1	Lecture 2	HW ID	LABS
22-Aug	1	Introduction	Asymptotic Notation -1	X	LAB-0
29-Aug	2	Asymptotic Notation -2	Review of Basics - 1	HW-1	LAB-1
5-Sep	3	Review of Basics - 2	Review of Basics - 3	HW-2	LAB-2

- LAB-0 will make introduction to Python and exempt from any evaluation
- 5-Sep Monday is Labor Day, no classes

Tentative Schedule: Part-2

	Week	Lecture 1	Lecture 2	HW ID	LABS
12-Sep	4	Amortized Analysis -1	Amortized Analysis -2	HW-3	LAB-3
19-Sep	5	Recursions	Divide&Conquer	HW-4	LAB-4
26-Sep	6	Dynamic Prog. -1	Dynamic Prog. -2	HW-5	LAB-5
3-Oct	7	Heap	Huffman Codes	X	LAB-6
10-Oct	8	Review	Midterm	HW-6	X

- 12 Oct. Wednesday - MIDTERM
- Fall-Break between 14-17 Oct , so No Labs on week 8

Tentative Schedule: Part-3

	Week	Lecture 1	Lecture 2	HW ID	LABS
17-Oct	9	Sorting&Selection -1	Sorting&Selection -2	HW-7	LAB-7
24-Oct	10	Greedy Algorithms	Graph-1	HW-8	LAB-8
31-Oct	11	Graph-2	Graph-3	HW-9	LAB-9
7-Nov	12	Randomized Algo 1	Randomized Algo 2	HW-10	LAB-10
14-Nov	13	Hashing-1	Hashing-2	X	X
21-Nov	14	X	X	X	X
28-Nov	15	Streaming Alg. - 1	Streaming Alg. - 2	HW-11	LAB-11
5-Dec	16	Miscellaneous topic	Review of course	X	X

- No LABS and HW on week 13 due to Thanks Giving
- No classes at all on 14th week again due to Thanks Giving

Tentative Schedule of Homework Assignments

HW ID	Announcement Date	Due Date	Week of The Course
1	September 2	September 9	2
2	September 9	September 16	3
3	September 16	September 23	4
4	September 23	September 30	5
5	September 30	October 7	6
6	October 14	October 21	8
7	October 21	October 28	9
8	October 28	November 4	10
9	November 4	November 11	11
10	November 11	November 28	12
11	November 28	December 5	15

Topics We Plan to Cover

TENTATIVE, SUBJECT TO CHANGES !

Week 1-3	Algorithm Analysis - Review	2 Lectures	Measuring the complexity, asymptotic notation, growth of functions..
	Fundamental Data Structures - Review	3 Lectures	Array, linked list, stack, queue, tree via visiting some interesting problems
Week 4	Amortized Analysis	2 Lectures	Analysis of algorithms that are hard to achieve with the classical approach
Week 5	Recursions and Divide-And-Conquer	2 Lectures	Recursions, master theorem, D&C type algorithms based on recursions
Week 6	Dynamic Programming	2 Lectures	Efficient handling of recursions with overlaps
Week 7	Heaps and Huffman Codes	2 Lectures	Priority queues, Huffman coding

Week 8 → MIDTERM

Topics We Plan to Cover

TENTATIVE, SUBJECT TO CHANGE !

Week 9	Selected topics in sorting, selection, and order	2 Lectures	Dictionaries, k-th smallest element, rank/select queries, wavelet trees
Week 10	Greedy Alg. & Intro. Graph Algorithms	2 Lectures	Scheduling, introduction to graphs
Week 11	Graph Algorithms	2 lectures	Representing graphs, traversal, shortest path, minimum spanning tree,
Week 12	Randomized Algorithms	2	Randomization with applications
Week 13	Hashing	2 Lectures	Basics of hashing, Bloom filters, min-hash, randomization, sample
Week 15	Streaming Algorithms	2 Lectures	Resorvour sampling, Misra-Gries and Count-Min Sketch
Week 16	Miscellaneous	2 Lectures	?

→ FINAL EXAM

Questions, Comments ?