Computer Programming II

Ming-Feng Tsai (Victor Tsai)

Dept. of Computer Science
National Chengchi University

Course Introduction

Course Information (1)

- Instructor
  - Name: Ming-Feng Tsai (蔡銘峰)
  - Email: mftsai@cs.nccu.edu.tw
  - Office: 大仁樓 413 室
    - Office Hours: Wed. 13-14 or email by arrangement
- Time
  - Lecture: Wed. 2-4 sessions (09:00-12:00)
  - Lab: Wed. C session (12:00-13:00)
- Place
  - Lecture & Lab: 大仁 200301

Course Information (2)

- Teaching Assistants (TAs)
  - Name: 吳武峰、簡傑、洪睿宸、陳奕玄
  - Office Hours: TBA
- Optional Textbooks
  - The C Programming Language, K&R. (全華代理)
  - Practical C Programming, Steve Oualline. (O'Reilly Media)
  - Master Algorithms with C, Kyle Loudon. (O'Reilly Media)
  - Advanced C, Peter D. Hipson.
  - UNIX Unbounded a beginning approach, Amir Afzal.

Course Information (3)

- Enrollment: Freshmen of CS department
- Prerequisites: Basic Programming Skill
- Grading
  - Midterm: 30%
  - Final Exam: 35%
  - Labs & Assignments: 35%
  - Bonus (participation): < 5%
- Assignments: 6 to 8 programming assignments
- Late Policy
  - Original points - n * 20 points (n = the number of delay days)
  - For example, 80 - 3 * 20 = 20 points with 3 days delay
  - Basically, no points after 5 days

Course Information (4)

- Grade Appeal
  - Student has two weeks (from the date handed back) to request a re-grade or appeal the grade recorded in the GradeBook. A re-grade will be performed on the entire Lab/Assignment/Exam and can lower the score!!
  - In any appeal procedure, it's the student's responsibility to keep possession of his/her Lab/Assignment/Exam. In the process of a re-grade, a student has to arrange for a TA to modify the grade in the presence of the student. A student should not hand over any material to the TA for keeping.
  - A lost or missing Lab/Assignment/Exam is no reason for a modification of a grade.

Course Information (5)

- Account: UNIX account on ghost.cs.nccu.edu.tw
- Website: http://wm5.nccu.edu.tw/
- Assignment Submission
  - Electronic Submission
  - Online judgement system

Course Information (6)

- SEVER PENALTIES ARE APPLIED FOR THE FOLLOWINGS
  - Actively sharing (or copying) all or parts of someone else's code/answers on Assignment/Exam
  - Passively allowing the sharing (or copying) of your own codes/answers on Assignment/Exam
- What is Cheating?
  - On Assignments: allowing others to view your code or reading some else's source code constitutes cheating. Students should protect their own work so that anther student cannot copy any part of their code. Therefore, if copying has been detected, it will be assumed that cheating has occurred by all parties involved.
  - On Exams: allowing others to view your answers or reading someone else's answer constitutes cheating.

Course Information (7)

- Cheating
  - Don’t do it!!
  - We’ll catch you, and we’ll punish you!!
- Penalties
  - A zero score for the assignment and a deduction of 20 points from the student’s course point total.
  - A second occurrence of cheating will mean an automatic grade of **ZERO** in the course and the notification of a Dean in student’s college.

Course Outline (1)

- Part I – Advanced Features of C
  - File Input/Output
  - Pointer and Arrays
  - Advanced Pointers
  - C Preprocessor
  - Bitwise Operations
  - Modular Programming
  - Portability Problem
  - Debugging and Optimization

Course Outline (2)

Course Outline (2)

- Part II – Advanced Unix Tools
  - Unix Basics: essential commands, pipes, redirections
  - Effective Programming Tools: emacs/vi, svn/git, make, gdb
  - Useful Tools: grep, sed, diff, xargs, find, …
  - Scripting: Bash, Perl/Python

Course Outline (2)

- Part II – Advanced Unix Tools
  - Unix Basics: essential commands, pipes, redirections
  - Effective Programming Tools: emacs/vi, svn/git, make, gdb
  - Useful Tools: grep, sed, diff, xargs, find, …
  - Scripting: Bash, Perl/Python
- Part III: Basic Data Structure and Algorithms
  - Queue, Stack, Linked List
  - Searching, Sorting

Tentative Schedule (1)

Tentative Schedule (1)

Tentative Schedule (1)

Week	Topics

Tentative Schedule (1)

Week	Topics
1	Course Introduction

Tentative Schedule (1)

Week	Topics
1	Course Introduction
2	Unix Basics (1)

Tentative Schedule (1)

Week	Topics
1	Course Introduction
2	Unix Basics (1)
3	Unix Basics (2)

Tentative Schedule (1)

Week	Topics
1	Course Introduction
2	Unix Basics (1)
3	Unix Basics (2)
4	C Revisited

Tentative Schedule (1)

Week	Topics
1	Course Introduction
2	Unix Basics (1)
3	Unix Basics (2)
4	C Revisited
5	Advanced Pointers (1)

Tentative Schedule (1)

Week	Topics
1	Course Introduction
2	Unix Basics (1)
3	Unix Basics (2)
4	C Revisited
5	Advanced Pointers (1)
6	Advanced Pointers (2)

Tentative Schedule (1)

Week	Topics
1	Course Introduction
2	Unix Basics (1)
3	Unix Basics (2)
4	C Revisited
5	Advanced Pointers (1)
6	Advanced Pointers (2)
7	Module Programming

Tentative Schedule (1)

Week	Topics
1	Course Introduction
2	Unix Basics (1)
3	Unix Basics (2)
4	C Revisited
5	Advanced Pointers (1)
6	Advanced Pointers (2)
7	Module Programming
8	Basic Data Structure (1): Linked List

Tentative Schedule (1)

Week	Topics
1	Course Introduction
2	Unix Basics (1)
3	Unix Basics (2)
4	C Revisited
5	Advanced Pointers (1)
6	Advanced Pointers (2)
7	Module Programming
8	Basic Data Structure (1): Linked List
9	Midterm

Tentative Schedule (2)

Tentative Schedule (2)

Tentative Schedule (2)

Week	Topics

Tentative Schedule (2)

Week	Topics
10	Basic Data Structure (2): Stack and Queue

Tentative Schedule (2)

Week	Topics
10	Basic Data Structure (2): Stack and Queue
11	Basic Data Structure (3): Tree and Graph

Tentative Schedule (2)

Week	Topics
10	Basic Data Structure (2): Stack and Queue
11	Basic Data Structure (3): Tree and Graph
12	Basic Algorithm (1)

Tentative Schedule (2)

Week	Topics
10	Basic Data Structure (2): Stack and Queue
11	Basic Data Structure (3): Tree and Graph
12	Basic Algorithm (1)
13	Basic Algorithm (2)

Tentative Schedule (2)

Week	Topics
10	Basic Data Structure (2): Stack and Queue
11	Basic Data Structure (3): Tree and Graph
12	Basic Algorithm (1)
13	Basic Algorithm (2)
14	Problem Solving (1)

Tentative Schedule (2)

Week	Topics
10	Basic Data Structure (2): Stack and Queue
11	Basic Data Structure (3): Tree and Graph
12	Basic Algorithm (1)
13	Basic Algorithm (2)
14	Problem Solving (1)
15	Problem Solving (2)

Tentative Schedule (2)

Week	Topics
10	Basic Data Structure (2): Stack and Queue
11	Basic Data Structure (3): Tree and Graph
12	Basic Algorithm (1)
13	Basic Algorithm (2)
14	Problem Solving (1)
15	Problem Solving (2)
16	Introduction to Shell Scripting (1)

Tentative Schedule (2)

Week	Topics
10	Basic Data Structure (2): Stack and Queue
11	Basic Data Structure (3): Tree and Graph
12	Basic Algorithm (1)
13	Basic Algorithm (2)
14	Problem Solving (1)
15	Problem Solving (2)
16	Introduction to Shell Scripting (1)
17	Introduction to Shell Scripting (2)

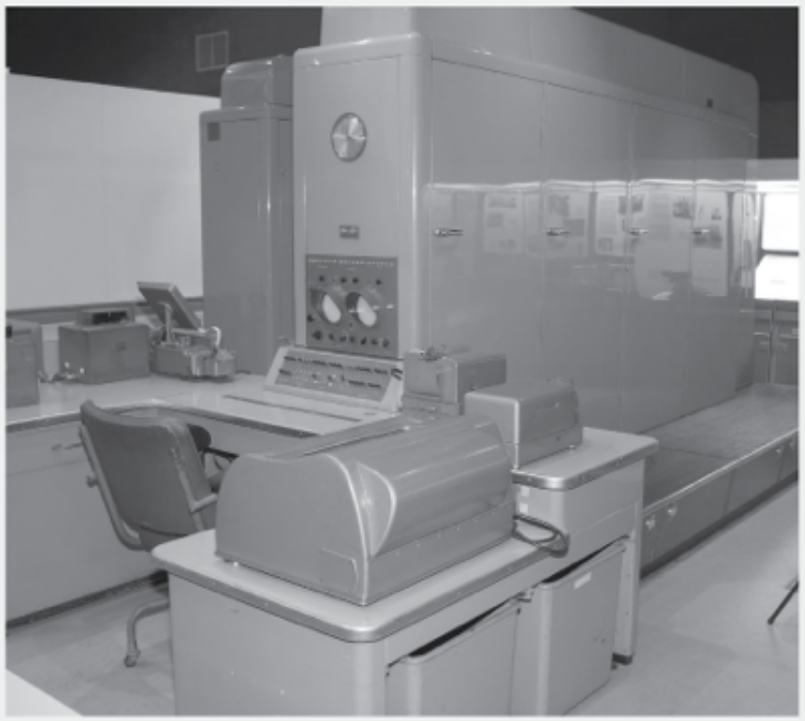
Tentative Schedule (2)

Week	Topics
10	Basic Data Structure (2): Stack and Queue
11	Basic Data Structure (3): Tree and Graph
12	Basic Algorithm (1)
13	Basic Algorithm (2)
14	Problem Solving (1)
15	Problem Solving (2)
16	Introduction to Shell Scripting (1)
17	Introduction to Shell Scripting (2)
18	Final Exams

Questions?

Evolution of Computer Languages

YEAR: 1957
LANGUAGE: FORTRAN



Tim Jenner / Shutterstock.com

FORmula**TRAN**slation, is the oldest language still in use. Created by John Backus, the language was developed to perform high-level scientific, mathematical, statistical computations.

The language is still used in aerospace, automotive industries, government, and research institutions.

» Used by
NATIONAL WEATHER SERVICE

A LOOK AT THE CODE:

```
*
C Hello World in Fortran 77
C (lines must be 6 characters
indented)
*
PROGRAM REELECT
WRITE(UNIT=*, FMT=*)
'I like Ike'
END
```

Created in 1874 to increase typing speed, the QWERTY keyboard was responsible for the majority of computer languages ever created.

The arrangement was set based on an analysis of keys likely to cause jams and designing the QWERTY board to separate likely offenders, like T and H.

YEAR: 1959
LANGUAGE: COBOL

COmmon **B**usiness **O**riented **L**anguage is behind the majority of business transaction systems running credit card processing, ATMs, telephone and cell calls, hospital systems, government, automotive systems, and traffic signal systems. The COBOL development team, lead by Dr. Grace Murray Hopper, set out to create a uniform, user-friendly language for business transactions.

UNITED STATES POSTAL SERVICE » Used by UNITED STATES POSTAL SERVICE

A LOOK AT THE CODE:

```
IDENTIFICATION DIVISION.
PROGRAM-ID.
StandardAlert.
AUTHOR. Fabritius.
ENVIRONMENT DIVISION.
CONFIGURATION SECTION.
INPUT-OUTPUT SECTION.
DATA DIVISION.
FILE SECTION.
WORKING-STORAGE SECTION.
LINKAGE SECTION.
PROCEDURE DIVISION.
DISPLAY "DANGER! DESTROY!".
STOP RUN.
```

In 1937, Binary Code was Claude Shannon's thesis on translating text into mathematical code was the foundation for the first fully operational electromechanical computer, Zuse's 1941 Z3.

Computers still speak binary code, but most modern programmers never touch the 0s and 1s.

YEAR: 1964
LANGUAGE: BASIC

Developed by students at Dartmouth College, Beginners All Purpose Symbolic Instruction Code was designed to be a simplified language for those without a strong technical or mathematical background. A modified version, written by Bill Gates and Paul Allen became Microsoft's first product. It was sold to M.I.T.S. for the Altair.

» INTEGER BASIC RAN THE ORIGINAL APPLE II IN 1977

A LOOK AT THE CODE:

```
100 BEGIN
101 GOTO 102
102 PRINT "HOW ABOUT
A NICE GAME OF CHESS?"
103 END
```

Basic has over 2 million lines of code in use, in 1975 there were only 4,000.

YEAR: 1969
LANGUAGE: C

C was developed between 1969 and 1973 by Dennis Ritchie at the Bell Telephone Laboratories for use with the Unix operating system. It was named "C" because its features were derived from an earlier language called "B."

C had become powerful enough that most of the Unix kernel was rewritten in C - one of the first operating system kernels implemented in a language other than assembly.

» LINUX TODAY IS BASED ON C

A LOOK AT THE CODE:

```
#include <stdio.h>

main()
{
  puts ("your first C
program");
}
```

YEAR: 1970
LANGUAGE: PASCAL

The language was named for Blaise Pascal, credited for inventing the first adding machine in 1641. Niklaus Wirth created Pascal as a teaching tool and it grew to into widespread commercial use.

» Used by SKYPE (OBJECT PASCAL)

A LOOK AT THE CODE:

```
PROGRAM SageAdvice (OUTPUT);
BEGIN
WRITELN('If you spent all those
hours');
WRITELN('Learning C instead of
Pascal');
WRITELN('you might have a job
now.');
END.
```

The first version of Word had 27,000 lines of code. Today, the current version of Office has over 30 million.

YEAR: 1983
LANGUAGE: C++

From Bell Labs, Bjarne Stroustrup modified the C language to C++ and created what many consider the most popular programming language ever. It's been listed in the top ten programming languages since 1986 and achieved Hall of Fame status in 2003.

» Used by MS OFFICE; ADOBE PDF READER; FIREFOX

A LOOK AT THE CODE:

```
#include<iostream>
using namespace std;
int main()
{
cout<< "C++ is the grade
you get when you're very,
very slightly above
average." <<endl;
return 0;
}
```

YEAR: 1987
LANGUAGE: PERL

Larry Wall, a UNIX programmer, created Perl after attempting to extract data for a report and finding UNIX couldn't perform the operations he needed. Practical Extraction Report Language was described by its inventor as a language for "getting your job done."

» Used by CRAIGSLIST

A LOOK AT THE CODE:

```
#!/usr/bin/perl
# Hello World in Perl
print "I don't always
write convoluted
scripts, but when I do,
I write them in Perl.
\n";
```

YEAR: 1991
LANGUAGE: PYTHON

The Mac OS/X uses about 90 million lines of code.

Monty Python served as the inspiration for the name of this language. Guido Van Rossum developed Python to fix problems in the ABC language and continues to serve as its lead designer.

YouTube » Used by GOOGLE SEARCH, YOUTUBE, NASA

A LOOK AT THE CODE:

```
# Hello World in Python
print"The airspeed velocity of
an unladen European Swallow is
approximately 11 meters per
second."
```

YEAR: 1993
LANGUAGE: RUBY

Yukihiro “matz” Matsumoto named Ruby for July’s birthstone. HE developed the language by blending parts of his favorite languages, Perl, Smalltalk, Eiffel, Ada, and Lisp.

» Used by BASECAMP

A LOOK AT THE CODE:

```
"I'd rather be
writing this in
Java.\n".display
```

YEAR: 1995
LANGUAGE: PHP

Rasmus Lerdorf developed PHP to replace a set Perl scripts used to maintain his personal home page. Today, PHP has grown in to an integral part of web architecture running on over 20 million websites.

» Used by FACEBOOK

A LOOK AT THE CODE:

```
<?php
echo"Fun Fact:
PHP used to
stand for
'Personal Home
Page'\n";
?>
```

A mere 15 million lines of code ran Windows 95. Windows 7 uses more than 50 million lines of code.

YEAR: 1995
LANGUAGE: JAVA

A team of Sun Microsystems developers lead by James Gosling created Java to run set top boxes for interactive television. Java now runs on over 1.1 billion PCs worldwide and many websites can't function with out it.

» Used by 2004 MARS ROVERS

A LOOK AT THE CODE:

```
public class HelloWorld {
public static void
main(String[] args) {
System.out.println("I
will now interrupt you
with update notifications
every other day for the
rest of your life.");
}
}
```

YEAR: 1995
LANGUAGE: JAVASCRIPT

Java and Javacript are unrelated and have very different semantics.

JavaScript was originally developed by Brendan Eich of Netscape under the name Mocha. JavaScript uses syntax influenced by that of C.

Although meant to run on the client (browser) it is now finding use on the server as node.js. Also, AJAX is dependent on Javascript.

rackspace HOSTING » Used by RACKSPACE (CLIENT SIDE)

A LOOK AT THE CODE:

```
<html>
<body>

<script type="text/javascript">
document.write("<h1>This is a
heading</h1>");
document.write("<p>This is a
paragraph.</p>");
document.write("<p>This is
another paragraph.</p>");
</script>

</body>
</html>
```

YEAR: 2005

LANGUAGE: RUBY ON RAILS

(Framework for programming language, Ruby)

Ruby on Rails was extracted by David Heinemeier Hansson from his work on Basecamp, a project management tool by 37signals. Hansson first released Ruby on Rails as open source in July 2004, but did not share commit rights to the project until February 2005. It is now on version 3.0.7 and has more than 1,800 contributors.

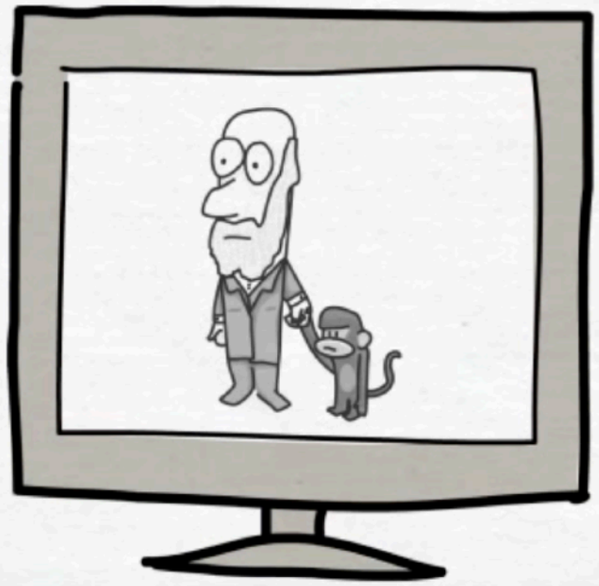
A single game application for the iPhone uses around 2 million lines of code.

A LOOK AT THE CODE:

```
def
section_link(name,options)
if options[:action] ==
@current_action and
options[:controller] ==
@current_controller
link_to(name, options,
:class => 'on') else
link_to(name,options)
end end
```

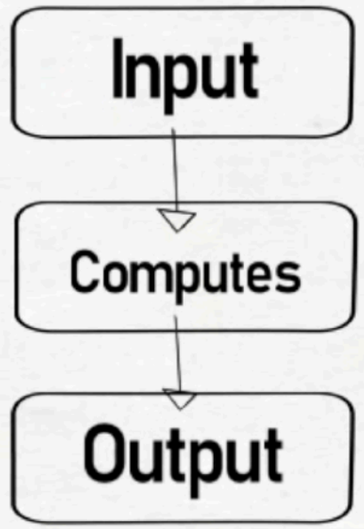
Feb 2023	Feb 2022	Change	Programming Language	Ratings	Change
1	1		Python	15.49%	+0.16%
2	2		C	15.39%	+1.31%
3	4	↑	C++	13.94%	+5.93%
4	3	↓	Java	13.21%	+1.07%
5	5		C#	6.38%	+1.01%
6	6		Visual Basic	4.14%	-1.09%
7	7		JavaScript	2.52%	+0.70%
8	10	↑	SQL	2.12%	+0.58%
9	9		Assembly language	1.38%	-0.21%
10	8	↓	PHP	1.29%	-0.49%
11	11		Go	1.11%	-0.12%
12	13	↑	R	1.08%	-0.04%
13	14	↑	MATLAB	0.99%	-0.04%
14	15	↑	Delphi/Object Pascal	0.95%	+0.05%
15	12	↓	Swift	0.93%	-0.25%
16	16		Ruby	0.83%	-0.06%
17	19	↑	Perl	0.79%	-0.01%
18	22	⇈	Scratch	0.76%	+0.13%
19	17	↓	Classic Visual Basic	0.74%	-0.09%
20	24	⇈	Rust	0.70%	+0.16%

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



History and Evolution
of
Computers
and Computer Programming
Language

"Computer"



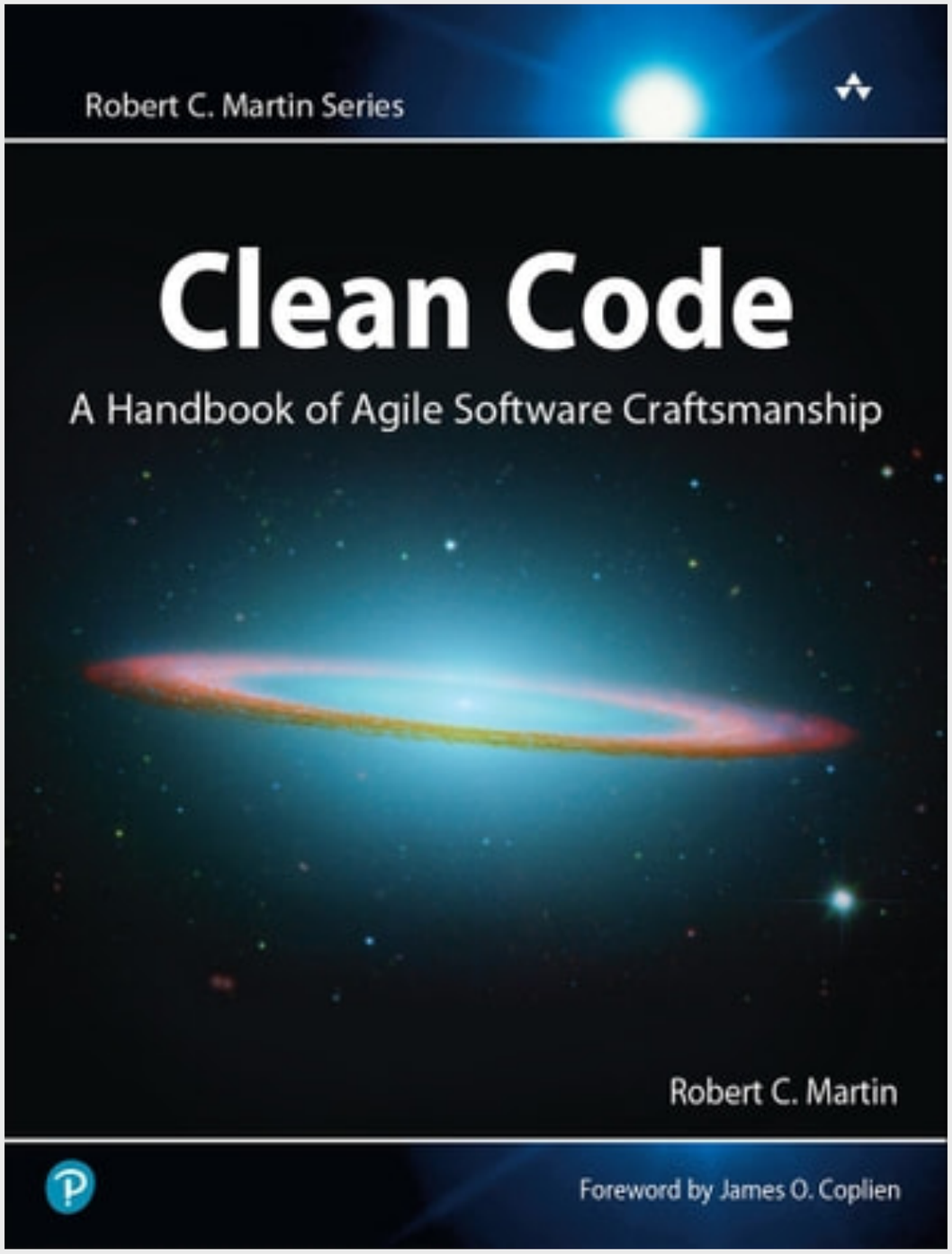
0:21 / 7:18

https://www.youtube.com/watch?v=Pn5znSOGHcs

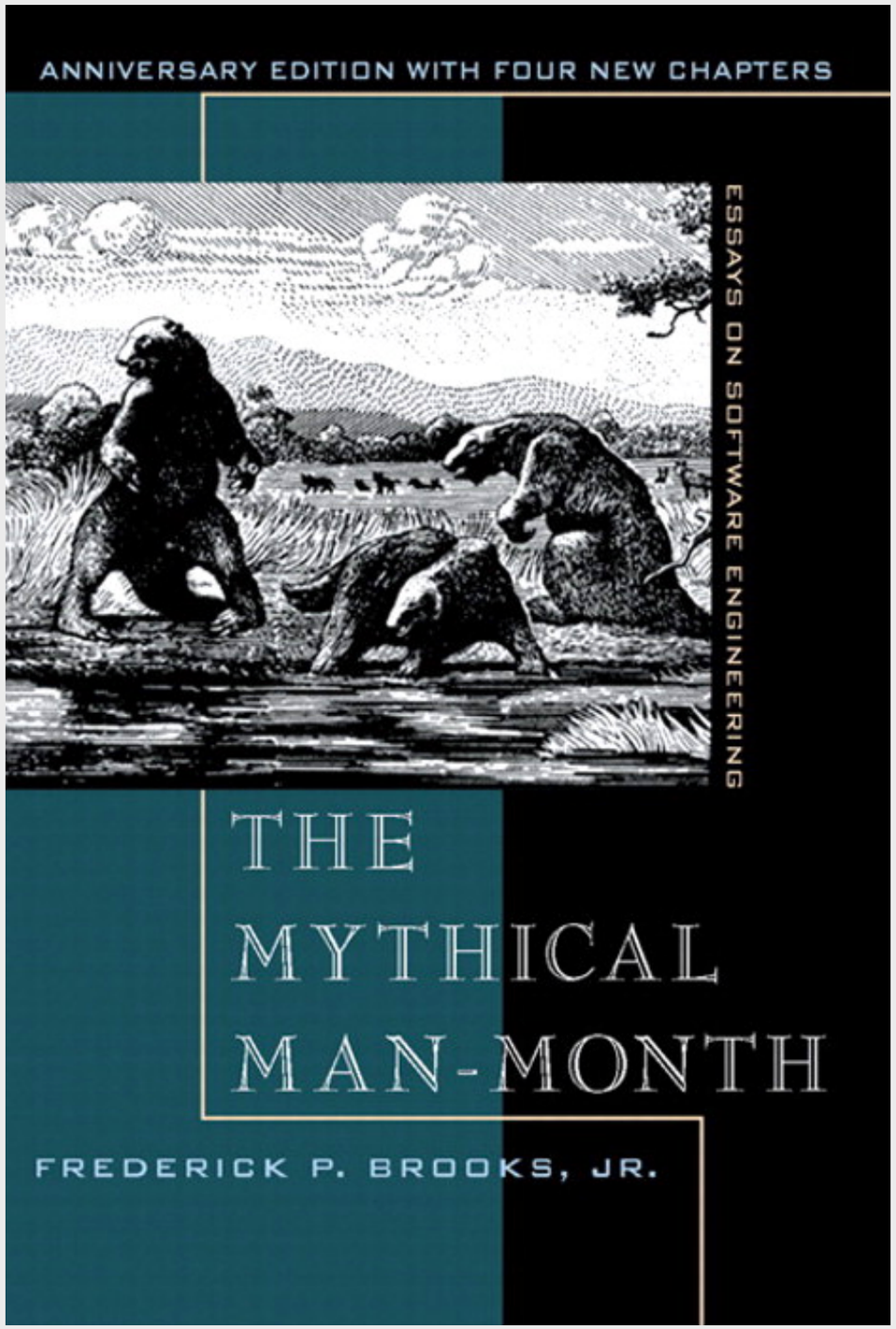
10 Programming Books That Every Programmer Must Read Once

https://www.geeksforgeeks.org/best-programming-books/

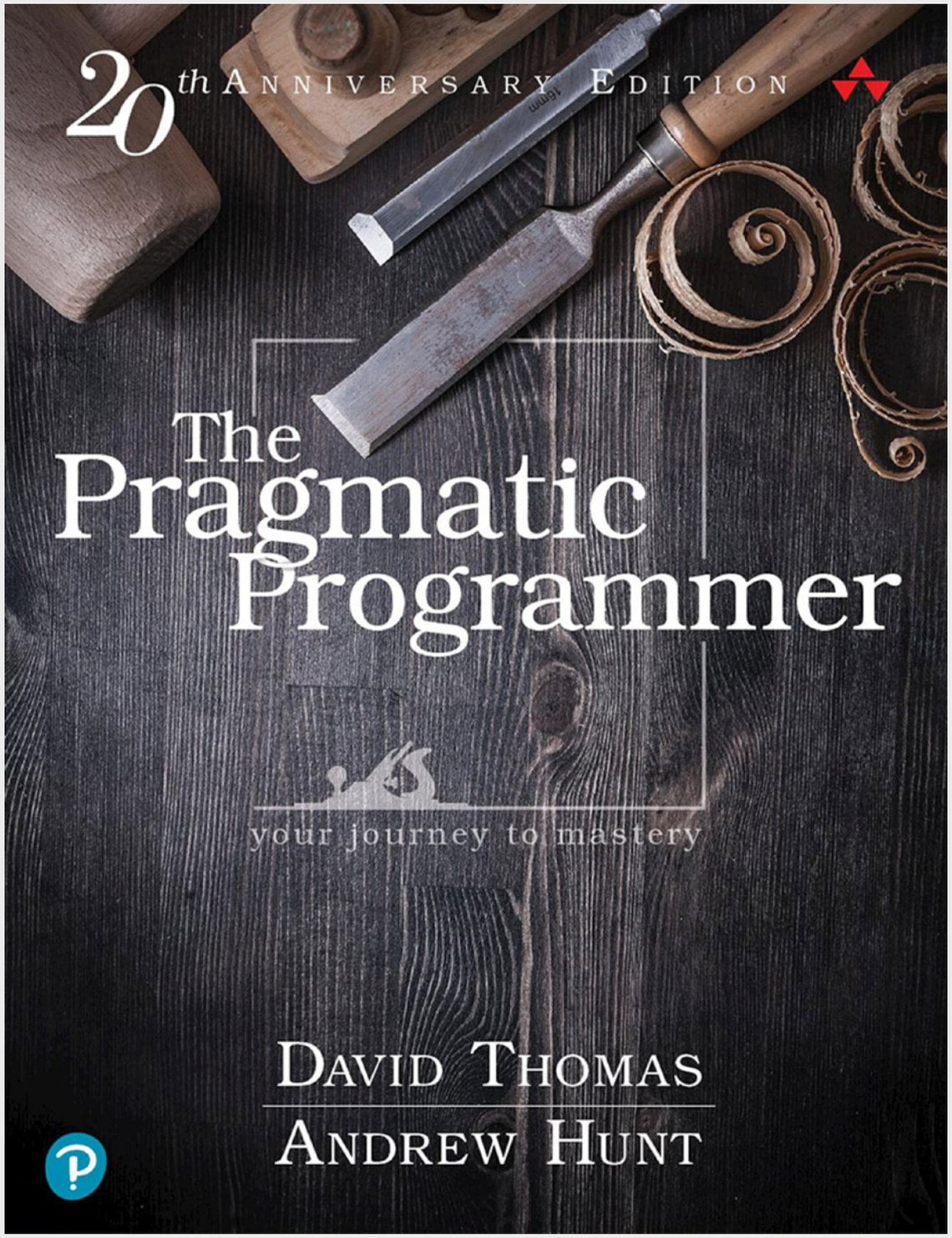
1. Clean Code


Robert C. Martin Series
Clean Code
A Handbook of Agile Software Craftsmanship
Robert C. Martin
Foreword by James O. Coplien

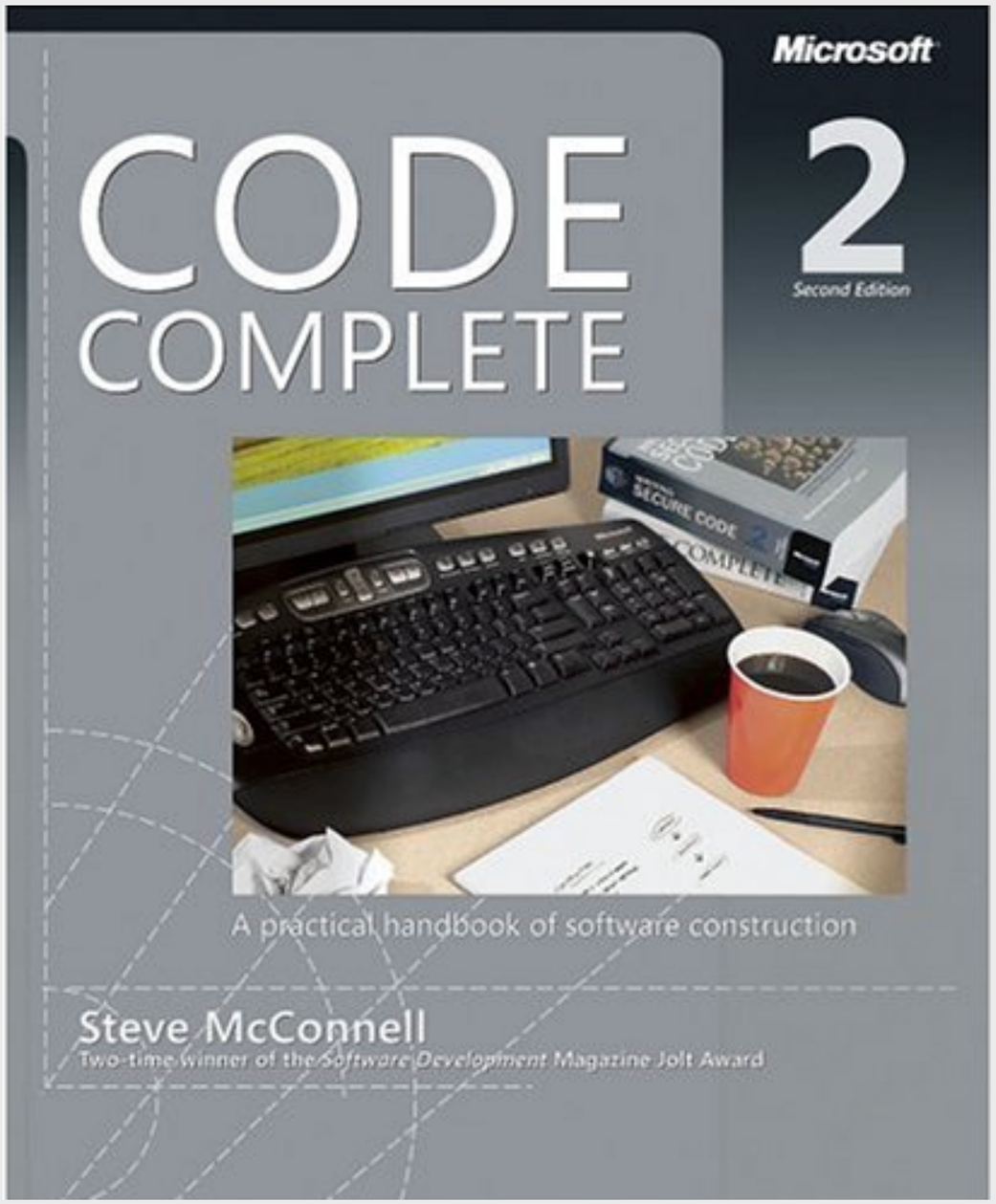
2. The Mythical Man-Month: Essays on Software Engineering



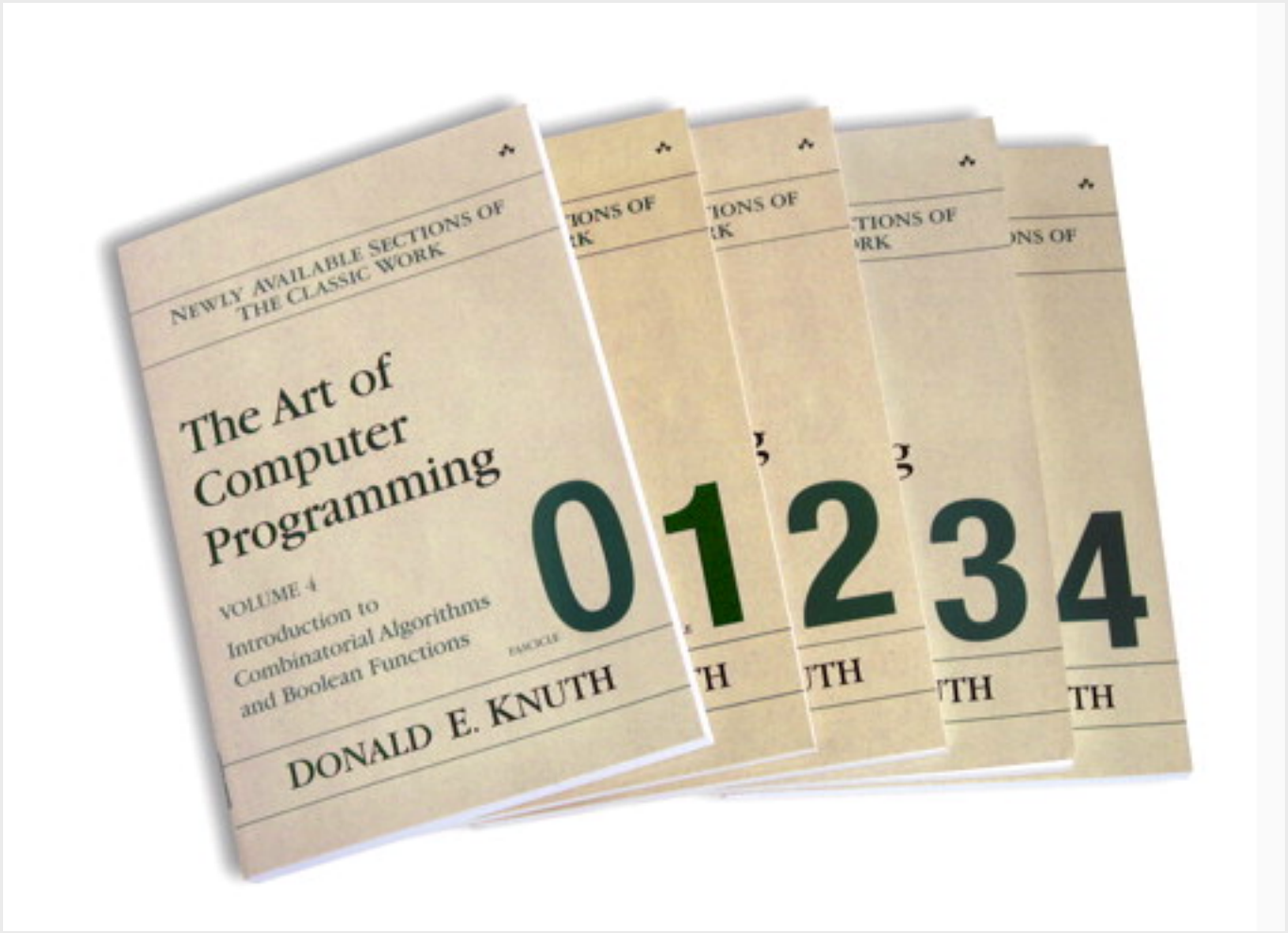
3. The Pragmatic Programmer: Your Journey to Mastery



4. Code Complete: A Practical Handbook of Software Construction

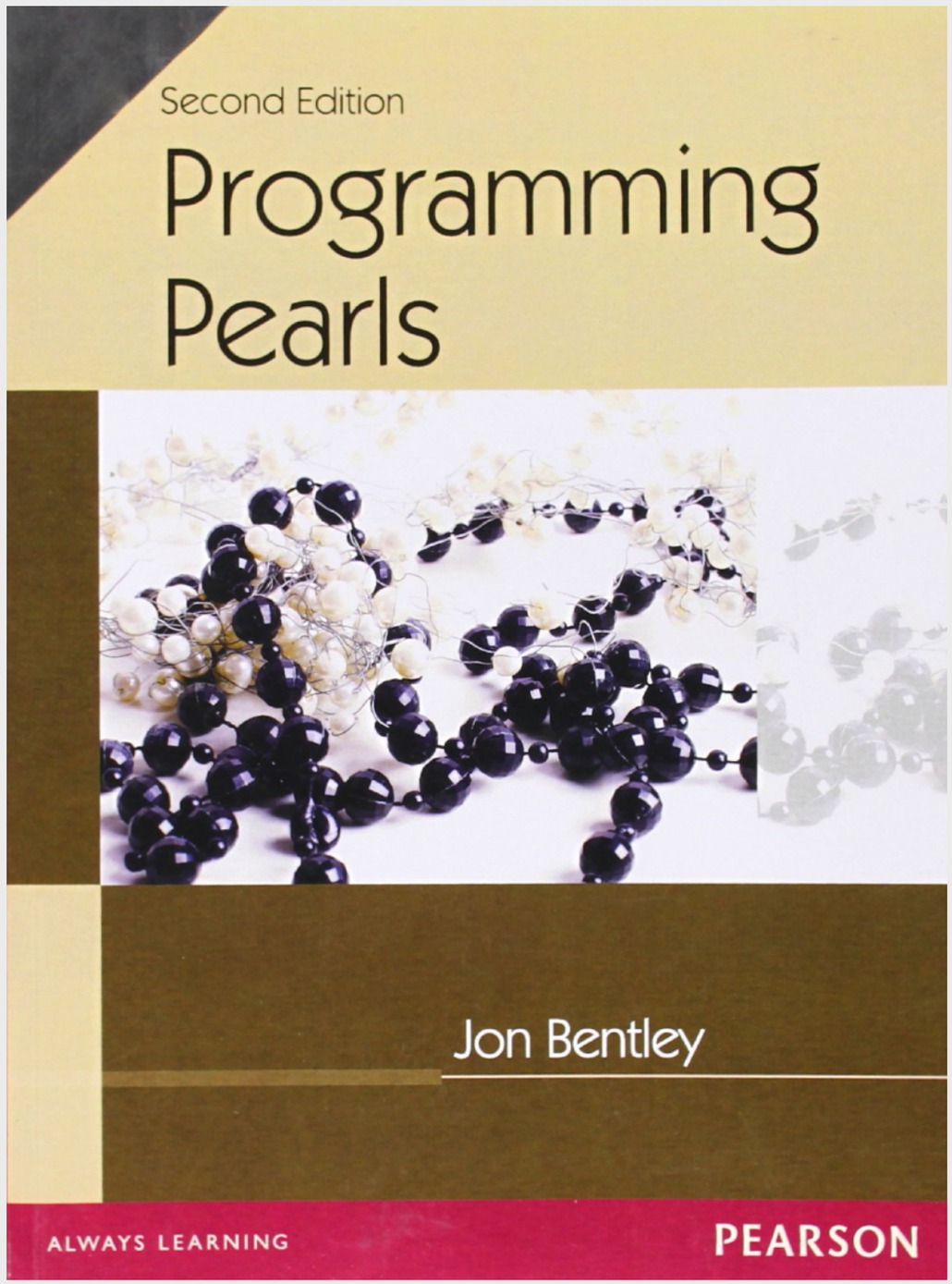


5. The Art of Computer Programming

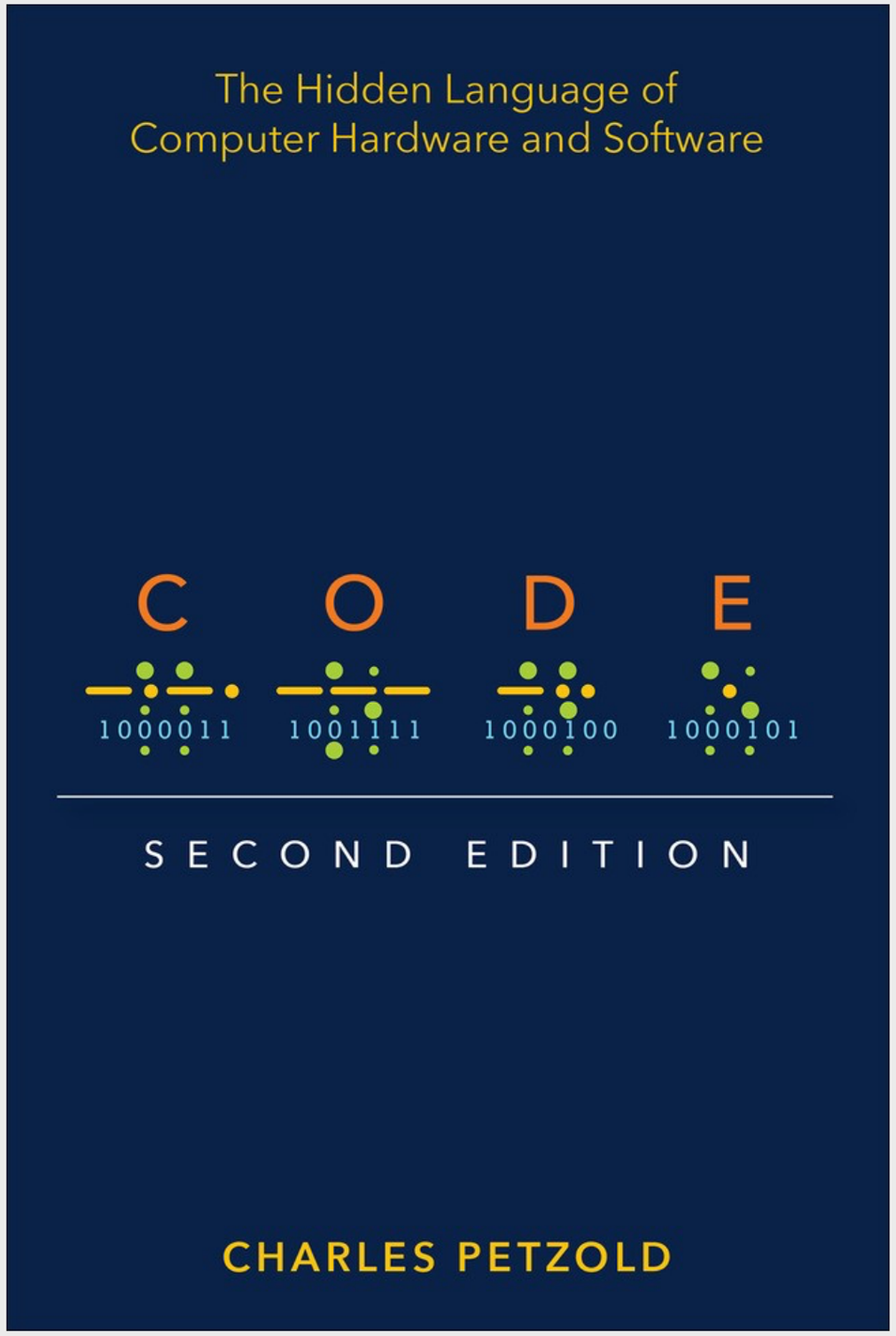


https://en.wikipedia.org/wiki/The_Art_of_Computer_Programming

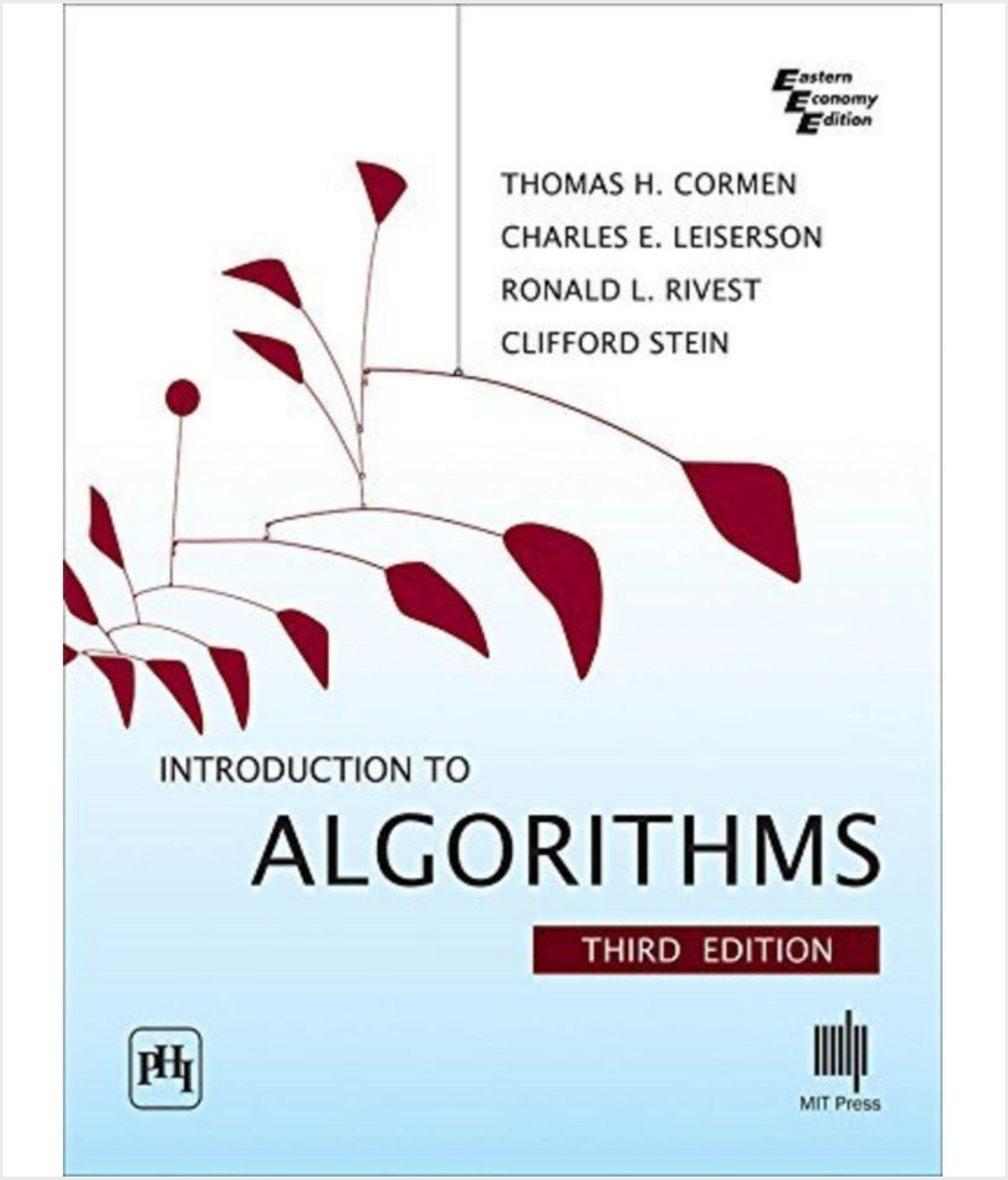
6. Programming Pearls



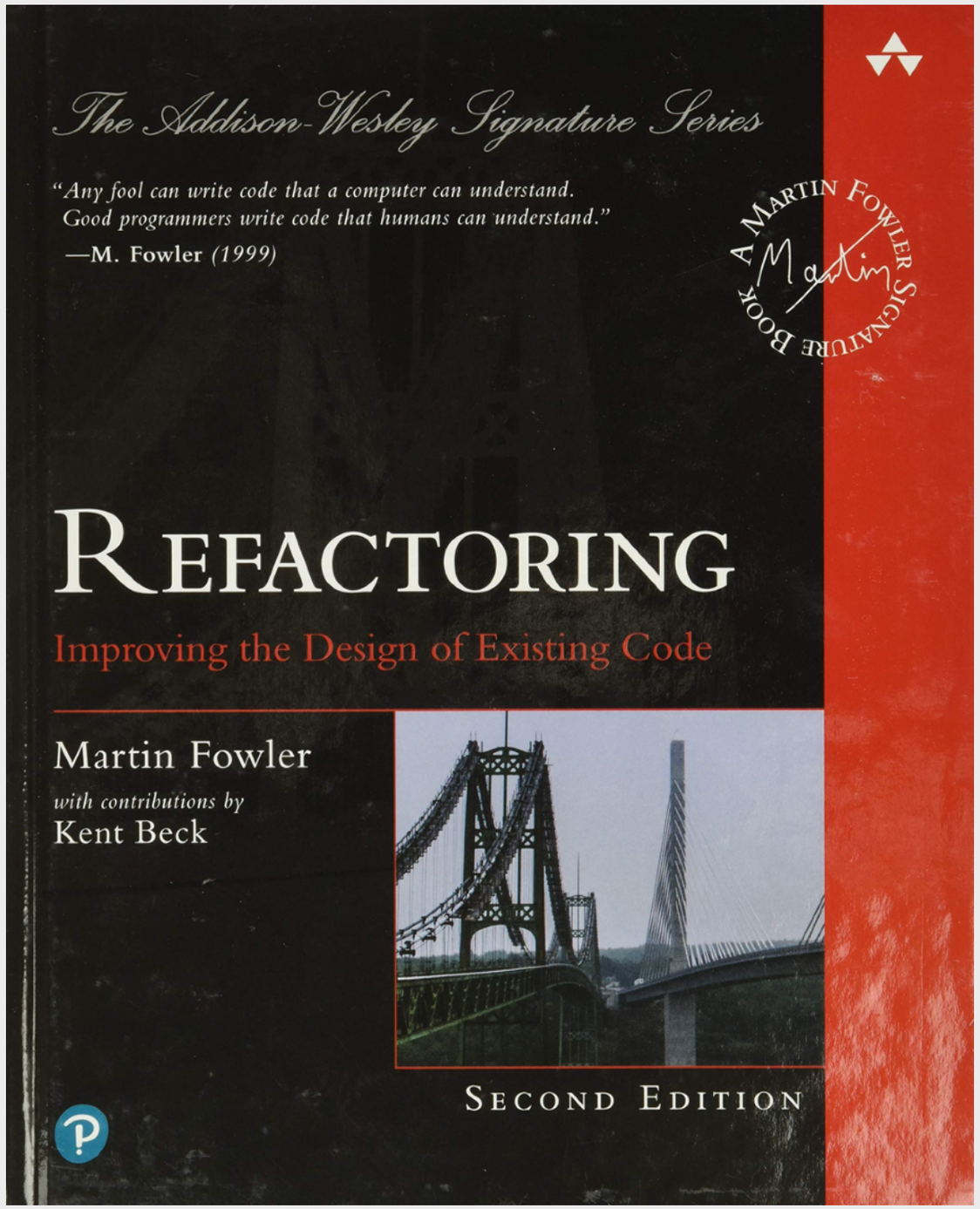
7. Code: The Hidden Language of Computer Hardware and Software



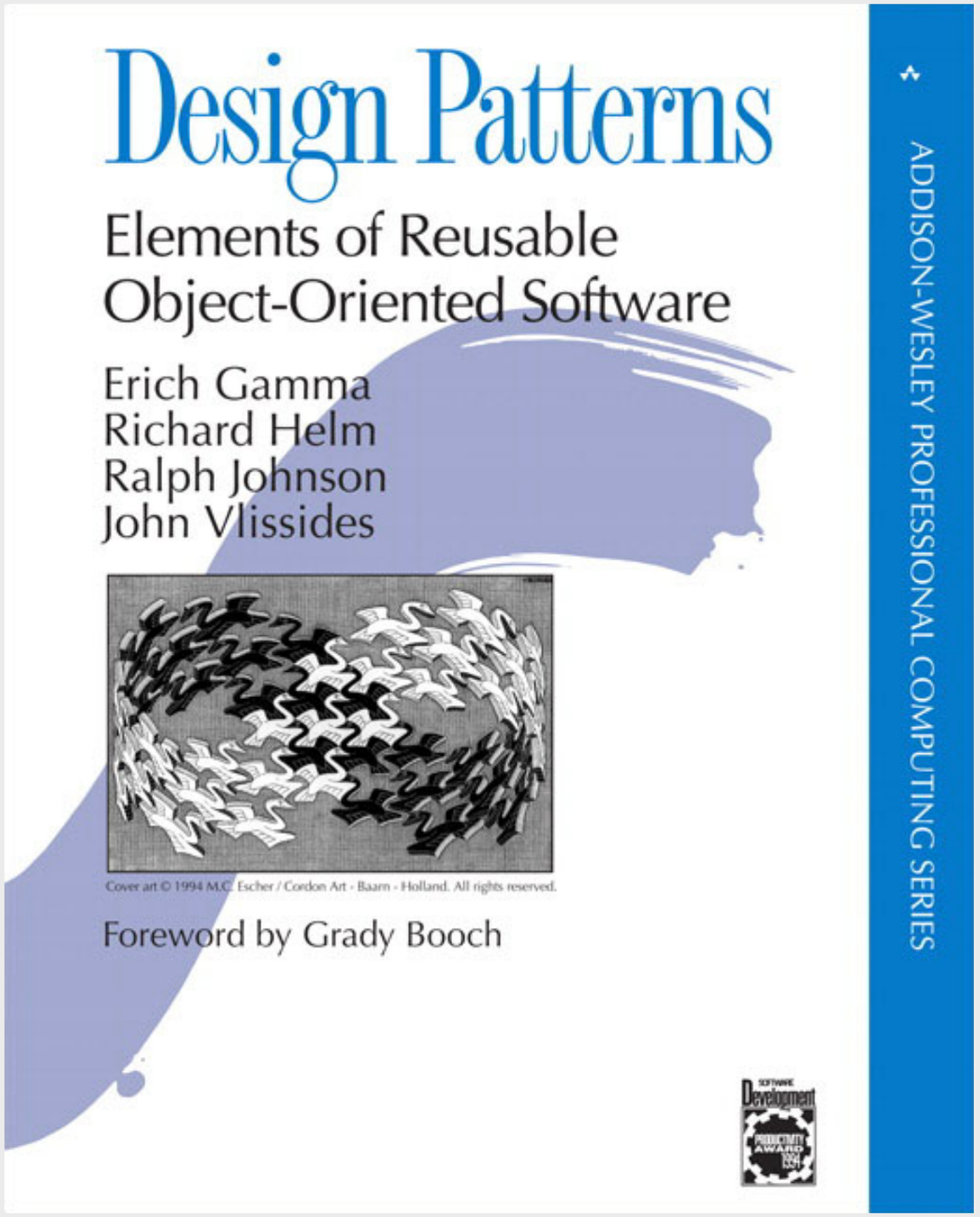
8. Introduction to Algorithms



9. Refactoring: Improving the Design of Existing Code



10. Design Patterns: Elements of Reusable Object-Oriented Software



加簽順序

- 本課程為**資訊系大一基礎必修課**，加簽順序為:
  - 1.本系本班、轉系
  - 2.重修
  - 3.雙修
  - 4.輔系
  - 5.其他（依加簽序號遞補）
- **待選課加退選開始後，請於將正課加簽單列印出來帶至課堂上手動加簽，依上述順序進行加簽至雙輔生結束為止！（如沒有加簽到，歡迎旁聽）**