ICCS313 Algorithm Analysis

Term I/2019-20

Dr.Chaya Hiruncharoenvate

Who am I?

Officer @ Data Management & Analytics Department,
Securities and Exchange Commission

เจ้าหน้าที่บริหาร, ฝ่ายจัดการและวิเคราะห์ข้อมูลตลาดทุน
สำนักงานคณะกรรมการกำกับหลักทรัพย์และตลาดหลักทรัพย์ (กลต.)

PhD in Computer Science

MS in Very Large Information Systems

BS in Computer Science

What about you?

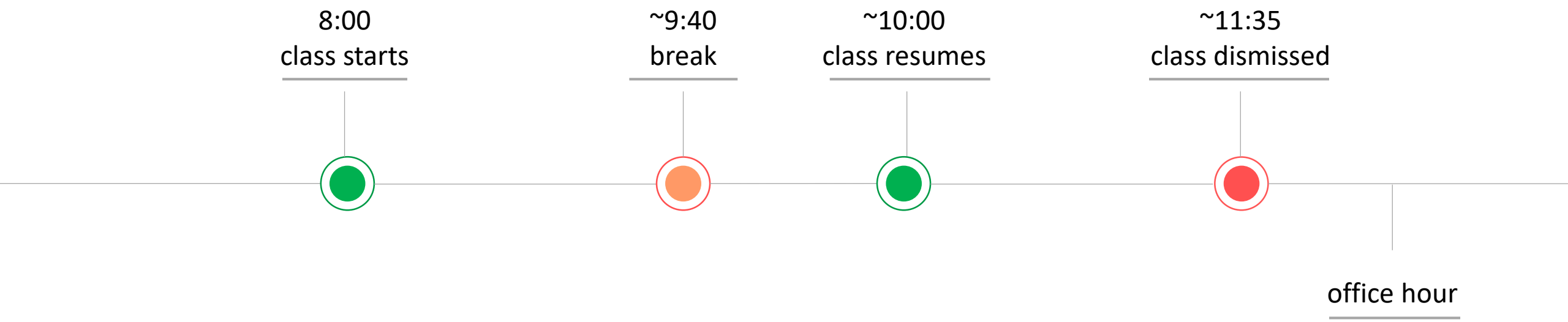
Syllabus

Course Website

https://canvas.instructure.com/enroll/MC3DEM

Time & Location

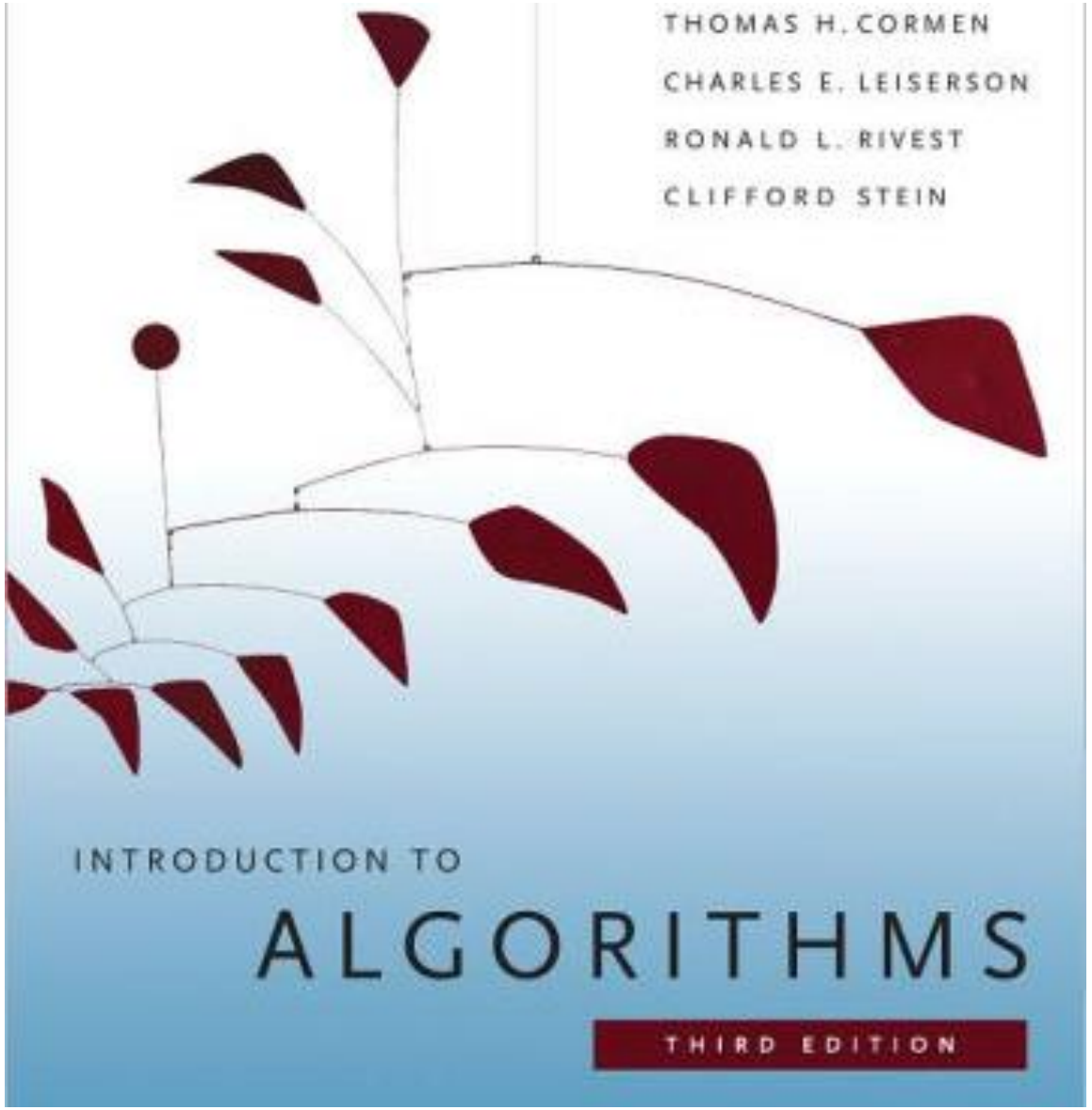
Saturdays 8:00-11:50 @ A322



Required Textbooks

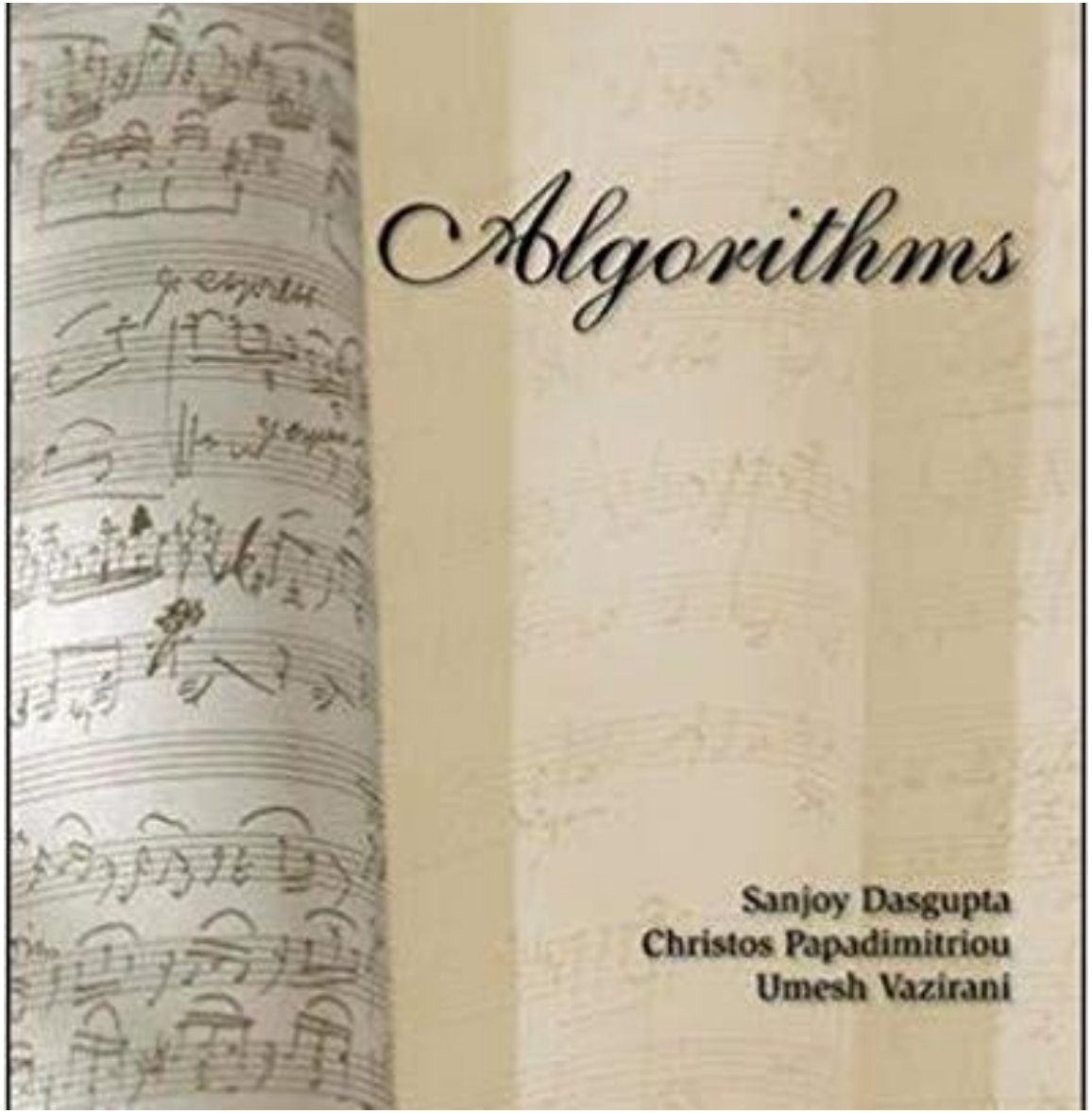
CLRS:

Cormen, T., Leiserson, C., Rivest R., and Stein, C. Introduction to Algorithms. 3rd Edition.



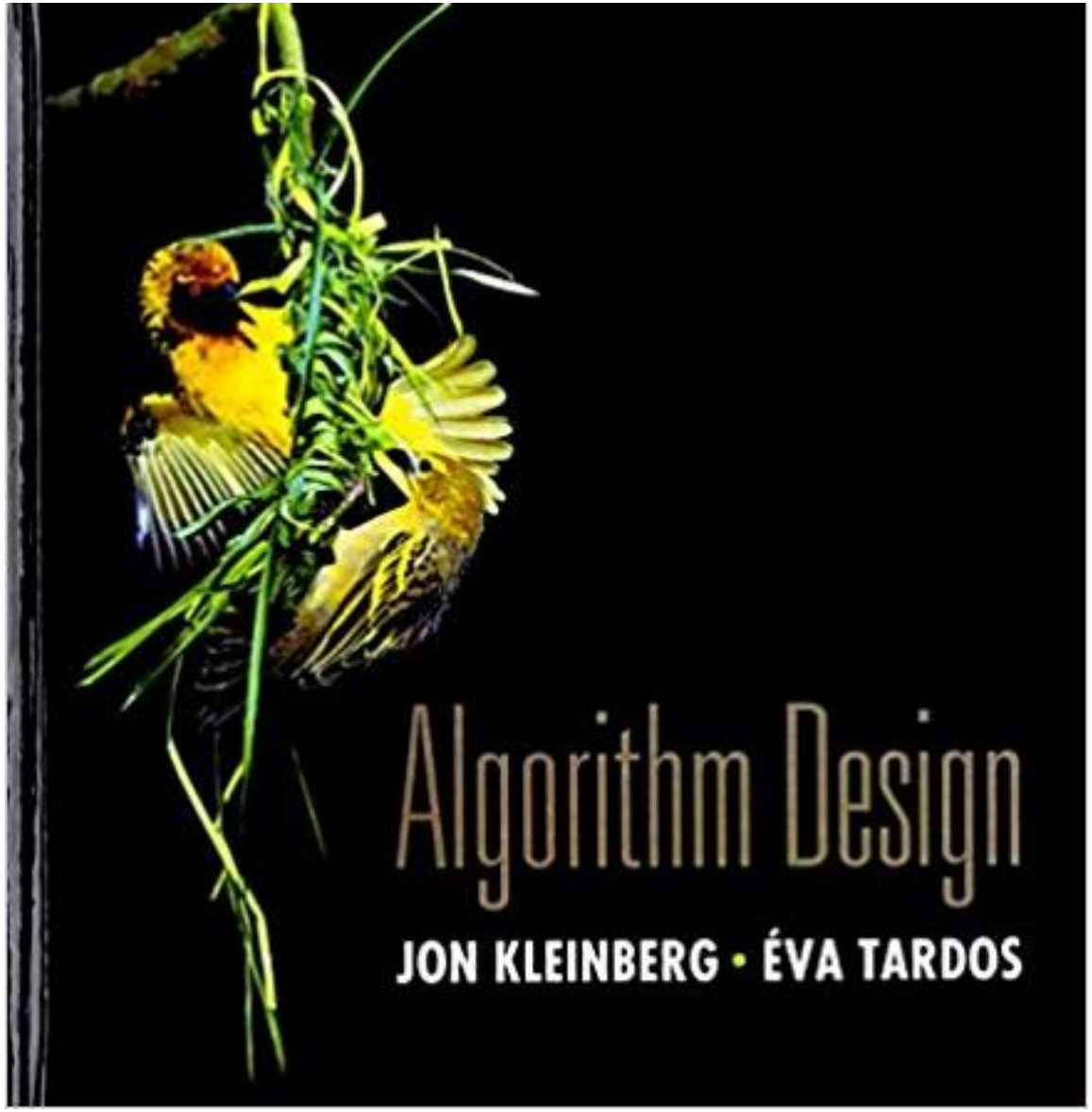
Optional Textbooks

Vazirani, U., Papadimitriou, C. and Dasgupta, S. Algorithms. *Online Version*.



Optional Textbooks

Kleinberg, J. and Tardos E.
Algorithm Design. 1st Ed.



Grade Distribution

Item	Weight
Assignments	30%
In-class Quizzes	30%
Participation	10%
Final Exam	30%

Grade Distribution

Item	Weight
Assignments	**30%**
In-class Quizzes	30%
Participation	10%
Final Exam	30%

Assigned weekly/biweekly (depending on class pace)
A combination of written and programming problems.
Written portions must be typeset.
One 24-hour late day is granted. Other late submissions = -50% of full mark.
No submission accepted after 24 hour of deadline.

Grade Distribution

Item	Weight
Assignments	30%
In-class Quizzes	**30%**
Participation	10%
Final Exam	30%

3 in-class quizzes: week 4, 7, and 10, covering only new materials
~1 hour each
One 2-sided A4 cheat sheet allowed

Grade Distribution

Item	Weight
Assignments	30%
In-class Quizzes	30%
Participation	**10%**
Final Exam	30%

I will try my best to remember all of your names in a few weeks!

Grade Distribution

Item	Weight
Assignments	30%
In-class Quizzes	30%
Participation	10%
Final Exam	**30%**

Administered during the final exam week.

Honor Code

- Collaborations on assignments are welcomed! The work submitted must be your own.
- Plagiarism and Cheating will not be tolerated.

Detailed syllabus is available on the course website.

Any questions?

Algorithm

Class Activity 1

1

Get into groups of 3-4 students

2

Discuss

“What is an Algorithm?”

3

Create a mindmap of your definition of “Algorithm”

Algorithm

- Well-defined computational procedure
- Takes input, produces output
- A tool for solving a well-specified computational problem

Things to Consider

- Correctness
  - Does the algorithm solve all instances of problems?
- Efficiency
  - Speed
  - **What other things to consider?**

Card Sorting

Class Activity 2

Get into your group, develop an algorithm to sort the card sorting problem

You have a hand of 13 cards. Sort the card from lowest (A) to highest (K). If you have two cards of the save number, the order of these two cards doesn’t matter.

Consider the following…

- Correctness
  - Does your algorithm work for any 13-card hand?
  - Does your algorithm work for any hand (any number of cards)?
- Efficiency
  - How many comparisons your algorithm make?
  - How many swaps your algorithm make?
- Implementation
  - How would you implement your algorithm in any language of your choice?
  - What data structure would you use?

Insertion Sort

INSERTION-SORT(A)

```
1   for j = 2 to A.length
2       key = A[j]
3       // Insert A[j] into the sorted sequence A[1 .. j − 1].
4       i = j − 1
5       while i > 0 and A[i] > key
6           A[i + 1] = A[i]
7           i = i − 1
8       A[i + 1] = key
```

Loop invariant

- Statement/Expression that is true in all iterations of the loop

Loop invariant?

```
INSERTION-SORT(A)
1  for j = 2 to A.length
2      key = A[j]
3      // Insert A[j] into the sorted sequence A[1 .. j − 1].
4      i = j − 1
5      while i > 0 and A[i] > key
6          A[i + 1] = A[i]
7          i = i − 1
8      A[i + 1] = key
```

Loop invariant

```
INSERTION-SORT(A)
1  for j = 2 to A.length
2      key = A[j]
3      // Insert A[j] into the sorted sequence A[1 .. j − 1].
4      i = j − 1
5      while i > 0 and A[i] > key
6          A[i + 1] = A[i]
7          i = i − 1
8      A[i + 1] = key
```

At the start of each iteration of the **for** loop (line 1-8), the subarray *A[1...j-1]* consists of elements originally in *A[1...j-1]* but in sorted order.

Proof of correctness

Show 3 things about our loop invariant

- Initialization
- Maintenance
- Termination

Initialization

The statement is true prior to the first iteration of the loop.

Maintenance

If the statement is true before an iteration of the loop, it remains true before the next iteration.

Termination

When the loop terminates, the invariant gives us a useful property that helps show that the algorithm is correct.

Proof of correctness

Show 3 things about our loop invariant

- Initialization
- Maintenance
- Termination

(board)

Induction

Time complexity

- ~ how much time the algorithm will use for some input
- 3 analysis
  - Worst-case
  - Best-case
  - Average-case

Worst-case Analysis

- Imagine the worst possible input and analyze the algo based on that.
- What's the worst case for our insertion sort?

Best-case Analysis

- Imagine the worst possible input and analyze the algo based on that.
- What’s the best case for our insertion sort?

Average-case Analysis

- Suppose our hand can be any hand with equal chance, on average how long (how many swaps) does the algorithm take?

Consider the searching problem

Input: A sequence of n numbers $A =< a_1, a_2, \ldots, a_n >$ and a value v.

Output: An index i such that $v = A[i]$ or the special value NIL if v does not appear in A.

Write pseudocode for **linear search**, which scans through the sequence, looking for v. Using a loop invariant, prove that your algorithm is correct. Make sure that your loop invariant fulfills the three necessary properties.