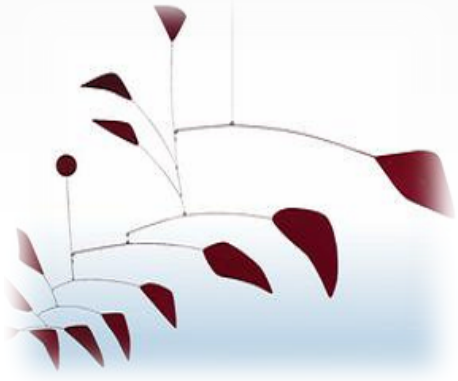


EECS 4101-5101
Advanced Data Structures

Shahin Kamali

Topic 1a - Formalities

York University

Picture is from the cover of the textbook CLRS.

Logistics

- Lecture: Mondays and Wednesdays, 17:30-19:00 at Keele CC 108 (Jan. 9, 2023 - Apr. 10, 2023)

Logistics

- Lecture: Mondays and Wednesdays, 17:30-19:00 at Keele CC 108 (Jan. 9, 2023 - Apr. 10, 2023)
- Webpage: https://www.eecs.yorku.ca/ kamalis/Teaching/Winter23/EECS4101.html

Logistics

- Lecture: Mondays and Wednesdays, 17:30-19:00 at Keele CC 108 (Jan. 9, 2023 - Apr. 10, 2023)
- Webpage: https://www.eecs.yorku.ca/ kamalis/Teaching/Winter23/EECS4101.html

Logistics

- Lecture: Mondays and Wednesdays, 17:30-19:00 at Keele CC 108 (Jan. 9, 2023 - Apr. 10, 2023)
- Webpage: https://www.eecs.yorku.ca/ kamalis/Teaching/Winter23/EECS4101.html
- Piazza: `https://piazza.com/yorku.ca/winter2023/advanceddatastructures`

Logistics

- Lecture: Mondays and Wednesdays, 17:30-19:00 at Keele CC 108 (Jan. 9, 2023 - Apr. 10, 2023)
- Webpage: https://www.eecs.yorku.ca/ kamalis/Teaching/Winter23/EECS4101.html
- Piazza: `https://piazza.com/yorku.ca/winter2023/advanceddatastructures`
- Tuesday 14:00 - 15:00 at LAS-3052B, Tuesday 15:00 - 16:00 on Zoom (or by appointment) `https://yorku.zoom.us/j/3531400655`
  - You can post your questions (if you prefer anonymously) on Piazza so that all your classmates see the solution

Textbook

- All materials from the class will be posted in the course web-page. The slides will be the main source for assignments and exams. The following book is the main reference for the materials covered in the class.
  - Introduction to Algorithms, third edition, by Cormen, Leiserson, Rivest, and Stein, MIT Press.
- Optional textbooks:
  - Algorithms and Data Structures, by Mehlhorn and Sanders, Springer, 2008.
  - Pat Morin, Open data structures (`https://opendatastructures.org/`).
  - Advanced Data Structures, by Brass, Cambridge, 2008.

Grading

- There will be:
  - Five assignments (25 percent)
  - Two quizzes (10 percent)
  - A midterm exam (20 percent)
  - A final exam (45 percent)

Grading

- There will be:
  - Five assignments (25 percent)
  - Two quizzes (10 percent)
  - A midterm exam (20 percent)
  - A final exam (45 percent)

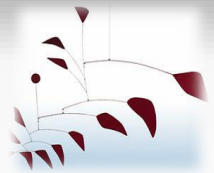
> ***Theorem***
>
> *The focus of this course is on learning, practicing, and discovering intuitive and practical data structures.*

Grading

- There will be:
  - Five assignments (25 percent)
  - Two quizzes (10 percent)
  - A midterm exam (20 percent)
  - A final exam (45 percent)

Theorem

The focus of this course is on learning, practicing, and discovering intuitive and practical data structures.

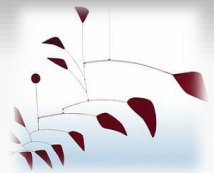


Grading (cntd.)

- Five assignments:
  - submit only pdf files (preferably use LaTeX) on Crowdmark (`https://www.crowdmark.com/`).

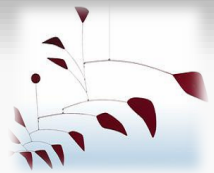
Grading (cntd.)

- Five assignments:
  - submit only pdf files (preferably use LaTeX) on Crowdmark (`https://www.crowdmark.com/`).
- Quizzes, Midterm & Final exams:
  - all are in-person and closed-book.
  - sample exams will be provided for practice for midterm and final.



Topics

- **Introduction:**
  - Amortization, Self-Adjustment, Competitiveness.



Topics

- **Introduction:**
  - Amortization, Self-Adjustment, Competitiveness.
- **Dictionaries:**
  - Move-to-Front Heuristic on Linear Lists.
  - Binary Search Trees review, Random BSTs, Red Black Trees.
  - B-trees, 2-3-4 Trees.
  - Splay Trees, Dynamic Optimality Conjecture.
  - Hash tables.
  - Augmenting Data Structures.

Topics

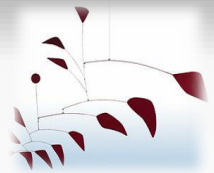
- **Introduction:**
  - Amortization, Self-Adjustment, Competitiveness.
- **Dictionaries:**
  - Move-to-Front Heuristic on Linear Lists.
  - Binary Search Trees review, Random BSTs, Red Black Trees.
  - B-trees, 2-3-4 Trees.
  - Splay Trees, Dynamic Optimality Conjecture.
  - Hash tables.
  - Augmenting Data Structures.
- **Priority Queues:**
  - Binomial Heaps and Fibonacci Heaps.

Topics

- **Introduction:**
  - Amortization, Self-Adjustment, Competitiveness.
- **Dictionaries:**
  - Move-to-Front Heuristic on Linear Lists.
  - Binary Search Trees review, Random BSTs, Red Black Trees.
  - B-trees, 2-3-4 Trees.
  - Splay Trees, Dynamic Optimality Conjecture.
  - Hash tables.
  - Augmenting Data Structures.
- **Priority Queues:**
  - Binomial Heaps and Fibonacci Heaps.
- **Disjoint Sets:**
  - Union-Find Data Structures.

Topics

- **Introduction:**
  - Amortization, Self-Adjustment, Competitiveness.
- **Dictionaries:**
  - Move-to-Front Heuristic on Linear Lists.
  - Binary Search Trees review, Random BSTs, Red Black Trees.
  - B-trees, 2-3-4 Trees.
  - Splay Trees, Dynamic Optimality Conjecture.
  - Hash tables.
  - Augmenting Data Structures.
- **Priority Queues:**
  - Binomial Heaps and Fibonacci Heaps.
- **Disjoint Sets:**
  - Union-Find Data Structures.
- **Compact and Succinct Data Structures:**
  - Rank/Select Data Structures, Tree Structures, Information Theoretic Lower Bounds.



Topics

- **Introduction:**
  - Amortization, Self-Adjustment, Competitiveness.
- **Dictionaries:**
  - Move-to-Front Heuristic on Linear Lists.
  - Binary Search Trees review, Random BSTs, Red Black Trees.
  - B-trees, 2-3-4 Trees.
  - Splay Trees, Dynamic Optimality Conjecture.
  - Hash tables.
  - Augmenting Data Structures.
- **Priority Queues:**
  - Binomial Heaps and Fibonacci Heaps.
- **Disjoint Sets:**
  - Union-Find Data Structures.
- **Compact and Succinct Data Structures:**
  - Rank/Select Data Structures, Tree Structures, Information Theoretic Lower Bounds.
- **A Taste of Algorithms**
  - Computation Geometry
  - Approximation and Online Algorithms

Summer Projects

- Consider applying for a summer project to become familiar with research in Algorithms and Data Structures:
  - **Compact Data Structures:** Data Structures that are space-efficient and allow fast queries.

Summer Projects

- Consider applying for a summer project to become familiar with research in Algorithms and Data Structures:
  - **Compact Data Structures:** Data Structures that are space-efficient and allow fast queries.
  - **Burning Graphs:** How fast does a meme/fake-news may distribute in a network?

Summer Projects

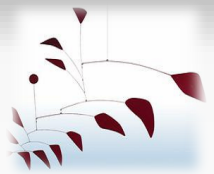
- Consider applying for a summer project to become familiar with research in Algorithms and Data Structures:
  - **Compact Data Structures:** Data Structures that are space-efficient and allow fast queries.
  - **Burning Graphs:** How fast does a meme/fake-news may distribute in a network?
  - **Fairness in algorithms:** How do we design algorithms that are “fair"? E.g., in the knapsack problem, you accept or reject items only depending on their value and size. What if the items are coming from two groups, and you want to accept a balanced number between grous?

Summer Projects

- Consider applying for a summer project to become familiar with research in Algorithms and Data Structures:
  - **Compact Data Structures:** Data Structures that are space-efficient and allow fast queries.
  - **Burning Graphs:** How fast does a meme/fake-news may distribute in a network?
  - **Fairness in algorithms:** How do we design algorithms that are "fair"? E.g., in the knapsack problem, you accept or reject items only depending on their value and size. What if the items are coming from two groups, and you want to accept a balanced number between grous?
  - **Online algorithms with prediction:** designing algorithms that receive inputs sequentially and also get erroneous predictions about the input.

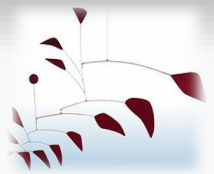
Important Dates (tentative)

January 9 first class
January 31 assignment 1 due
February 12, assignment 2 due
February 15, quiz 1
February 18-24 reading week
March 2 assignment 3 due
March 6 midterm
March 17 assignment 4 due
March 22 quiz 2
April 8 assignment 5 due
April 10 classes end
April 12-27 Winter examination



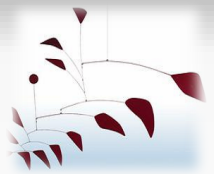
Prerequisites

- What I have learned from previous courses?
- Asymptotic notations, e.g., big O, Ω, o, ω, Θ.
- Basic algorithms and analysis techniques, e.g., binary search, sorting algorithms, loop analysis, solving recursions, etc.



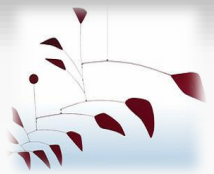
Prerequisites

- What I have learned from previous courses?
- Asymptotic notations, e.g., big O, Ω, o, ω, Θ.
- Basic algorithms and analysis techniques, e.g., binary search, sorting algorithms, loop analysis, solving recursions, etc.
- Algorithmic paradigms like Divide-and-Conquer, Greedy algorithms, and Dynamic Programming.



Prerequisites

- What I have learned from previous courses?
- Asymptotic notations, e.g., big O, Ω, o, ω, Θ.
- Basic algorithms and analysis techniques, e.g., binary search, sorting algorithms, loop analysis, solving recursions, etc.
- Algorithmic paradigms like Divide-and-Conquer, Greedy algorithms, and Dynamic Programming.
- Basic abstract data types (ADTs) and data structures
  - Stacks, queues, dictionaries, binary search trees, hash tables, heaps, graphs.

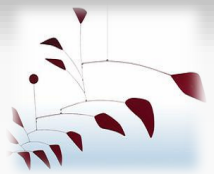


Abstract Data Type

- What is an Abstract Data Type (ADT)?

Definition

An abstract data type is formed by I) a set of values (data items) and II) a set of operations allowed on these items



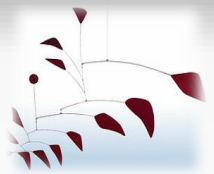
Abstract Data Type

- What is an Abstract Data Type (ADT)?

Definition

An abstract data type is formed by I) a set of values (data items) and II) a set of operations allowed on these items

- Stack is an ADT. Data items can be anything and operations are *push* and *pop*
- An ADT is abstract way of looking at data (no implementation is prescribed)
- An ADT is the way data 'looks' from the view point of user

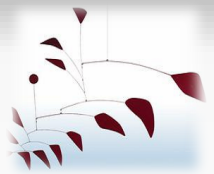


Data Structure

- What is a Data Structure?

Definition

A data structure is a concrete representation of data, including how data is organized, stored, and accessed on a computer



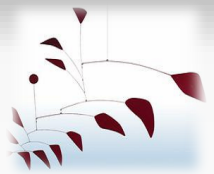
Data Structure

- What is a Data Structure?

Definition

A data structure is a concrete representation of data, including how data is organized, stored, and accessed on a computer

- A linked-list is a data structure
- Data structures are **implementations** of ADTs
- A data structure is the way data 'looks' from the view point of implementer



ADTs vs Data Structures

- ADTs: Stacks, queues, priority queues, dictionaries
- Data structures array, linked-list, binary-search-tree, binary-heap hash-table-using-probing, hash-table-using-chaining, adjacency list, adjacency matrix, etc.