Data Structures and Algorithms

Lecture 13

Aniket Basu Roy

2026-02-14 Sat

Contents

1 Agenda

1.1 Data Structures

2 Data Structures

- What and Why
  - a way to store data/information and
  - a way to rerieve
  - a way to delete
  - a way to create relationship among data, e.g., precedence, successor, etc.
- Linear: Arrays, Stacks, Queues, Deques, Linked Lists

- Non-Linear: Heaps, Binary Search Trees, Graphs
- Hashing: The magic of $O(1)$ lookup

2.1 Example: A computer game that stores a deck of playing cards.

A few queries I want to answer:

1. I want to add a card into the deck.
2. Which card is at the top of the deck?
3. Is the deck empty?
4. Given a card are there any higher rank cards of the same suit in the deck.

2.2 Stacks

- Last-In-First-Out

2.3 Queues

- First-In-First-Out

2.4 Linked Lists

- A node is a continuous block of memory.
- A sequence of nodes are connected using pointers

3 Stacks and Permutations

Given numbers $1, 2, \ldots, n$ and a stack, you are allowed to push elements in the increasing order, i.e., $push(1), \ldots, push(2), \ldots, push(n), \ldots$. You are free to $pop()$ the stack whenever you want; once you pop an element that gets printed. What permutations of $1, 2, \ldots, n$ can you print by choosing the above rules? Can you generate all $n!$ permutations?

- Example: For $n = 3$, the permutation 3 1 2 cannot be printed. Why?
- What can be said about general values of n?

- Observe that the sequences of push and pop have a 1-1 mapping with the sequences of balanced parantheses, i.e., For every push you write a "(", and for every pop you write a ")" .
- Example: $n = 3$
  - 123 ()()()
  - 132 ()(())
  - 213 (())()
  - 231 (()())
  - 312 Not possible
  - 321 ((()))
- For general n, the number of ways you can have balanced parantheses with n ('s and n)'s (hereafter referred as the string of order n) is expressed as the Catalan numbers, where the n^{th} Catalan number C_n is $\frac{1}{n+1}\binom{2n}{n}$. Why?
- Let S be a string of balanced parantheses of order n. Observe S can be expressed as $A(B)$, where A and B are strings of varying orders. More precisely, $C_n = C_0C_{n-1} + \ldots + C_{n-1}C_0$. Use generating functions to solve this recurrence relation.