Recursion

CSE 232 - Dr. Josh Nahum

Reading:

No Reading

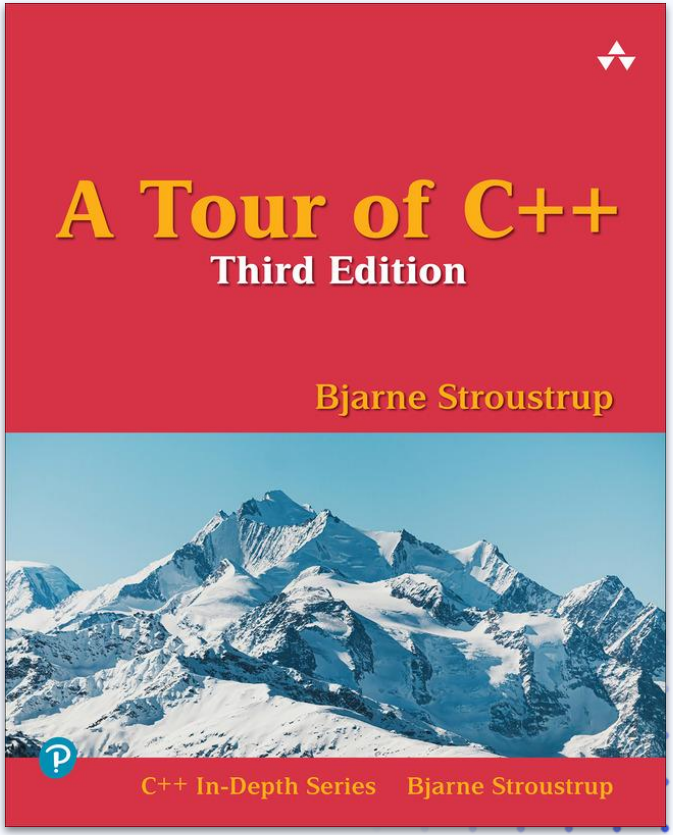


Table of contents

00 Motivation

CSE 331 - Algorithms and Data Structures

Efficiency

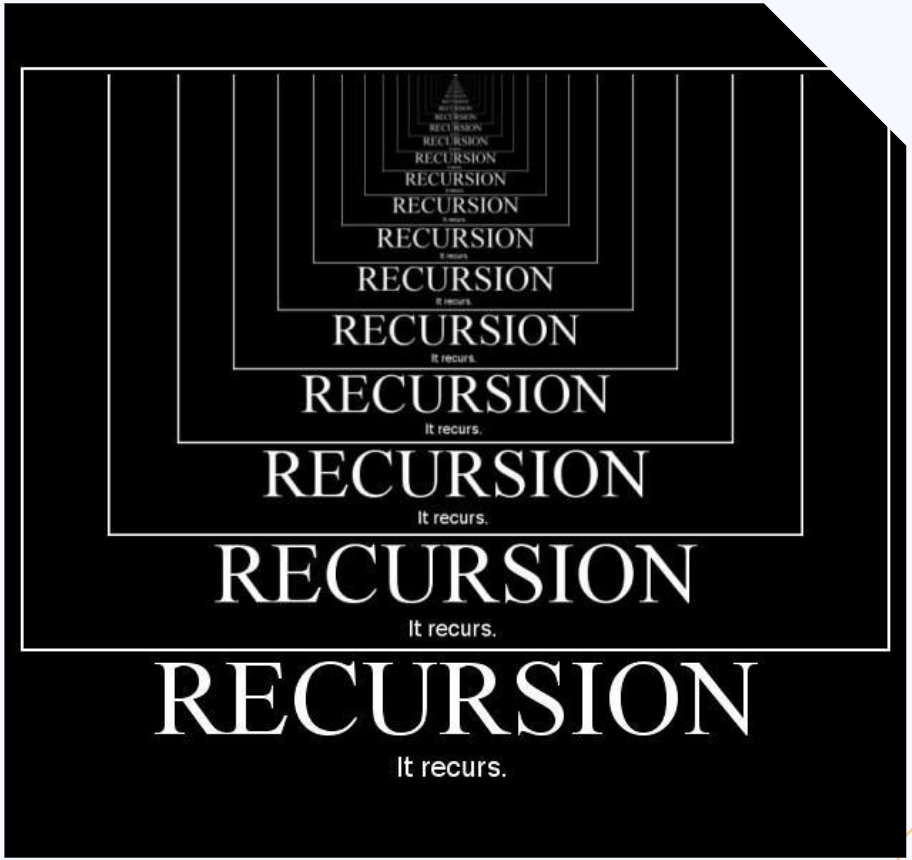
So far, correctness has been the only important criteria to evaluate code on. Soon, you'll need to care about performance (speed).

Theory and Design

The algorithms and data structures you use will be the primary determinant of whither your code runs in a reasonable amount of time and space.

Recursion

Recursion is probably the single more algorithmically important topic you will learn in CS.



01

Examples

Live Code Demonstation

02

Sort Algorithms

Select Sorting Survey

Bubble Sort

Easy to implement, very slow

Insertion Sort

Fast for small sequences, slow for long

Quicksort

Fast for large sequences, very common

Merge Sort

Also fast for large sequences, support "stable" sorting

Why so many?

There are often many ways to accomplish the same goal in computer science. It is important to understand the tradeoffs and constraints of the algorithms you choose.

03

Data Structures

Storing A Sequence

Arrays

std::vector (or std::array, or C-style arrays) give you efficient access to every element, but can be slow to grow in size due to needed to reallocate memory

Linked Lists

Linked lists are less space efficient and can only allow fast access to elements near the beginning and end, but they support much faster append operations

Other Data Structures

Graphs

Store nodes and links between them representing relationships between entities

Maps

Store associations between keys and values

Trees

Sort hierarchical and/or ordered data in a structure that support efficient search

Heaps

Structures that sort data in a manner that supports efficient access of the maximum value

Attribution

Please ask questions via Piazza

Dr. Joshua Nahum
www.nahum.us
EB 3504

CREDITS: This presentation template was created by **Slidesgo**, and includes icons by **Flaticon**, and infographics & images by **Freepik**