Session 8
Fundamentals

Lecture Summary

Derivation of generic term is low value long term, best dwell on the recurrence and aspects of its computation and performance

Fibonacci Numbers—Example Analysis of Recursive Algorithm

1. Fibonacci numbers occur naturally! (Check links in lecture slides for interesting stuff.)
2. Specifying the sequence via a recurrence
3. Solving the recurrence
4. Computation algorithms
5. Efficiency of computation algorithms

Session Exercise

P9. Code the recursive (call it *rFib*) and iterative (*iFib*) algorithms for computing *F*(n) Count the number of times the basic addition operation is performed for each. Compare count for *F*(8) with the value we obtained in class for *A*(8). **Hint**: read 8 in 🕮 Exercise 2.5.

P10. Use debugger timing to compare performance of *rFib* and *iFib* for large *n* (use *sortcomp.js* as example). Report results in discussion group.

🕮 **Exercise 2.5** • 5, 8 (see hint at end of Section 2.5)

> 👁 *This exercise illustrates that coding a solution efficiently is often not trivial even for simple problems*

Reading List

🕮 2.5

Keywords