• Java prerequisites
  Topics:
  – Classes, objects, and references
  – Access modifiers
  – Arguments and parameters
  – Garbage collection

  Self-test questions: Appendix B

• Designing classes
  Topics:
  – Composition and inheritance
  – Access modifiers
  – static keyword
  – Overriding methods
  – Dynamic binding and method polymorphism
  – Reference type vs. object type
  – Interfaces
  – Typecasting
  – Generic interfaces, classes, and methods
  – Generic type declarations, including bounded types, type wildcards, and bounded wildcards
  – Assertions

  Self-test questions: Appendix C; Prelude; Java Interludes JI1, JI5

Exercises:
  – Prelude: 1, 2
• Bag

Note: Since this was our first data structure, we covered several topics in this section even though they weren’t specific only to bags.

Topics:

– ADTs, collections, data structures and their relation to interfaces and classes
– Client vs. implementer
– Considering corner cases
– Test methods
– Bag ADT and interface
– Bag array vs. linked implementations
– Resizing arrays
– Inner classes, static and non-static

Self-test questions: Chapters 1–3

Exercises:

– Chapter 1: 1, 5
– Chapter 2: 1, 5, 6, 11
– Chapter 3: 1, 5, 12, 14

• Algorithm analysis

Topics:

– Asymptotic analysis
– Big-O notation
– Growth rates
– Amortized analysis
– Sum of the first n integers
– Analysis of bag implementations

Self-test questions: Chapter 4

Exercises:

– Chapter 4: 1–6, 10–12, 17
• Stack
  Topics:
  – Stack interface
  – Using stacks to match brackets
  – Using stacks to evaluate postfix
  – Using stacks to convert infix to postfix
  – Array vs. linked implementations of stack (runtime, memory usage)
  – Program stack / run-time stack

Self-test questions: Chapters 5, 6

Exercises:
  – Chapter 5: 1, 3, 6–8
  – Chapter 6: 1, 3, 5, 8, 9

• Recursion
  Topics:
  – Breaking problems into subproblems
  – Requirements for recursion to work
  – Activation records
  – Divide & conquer vs. general recursion
  – Tail recursion
  – Easy vs. hard recursive algorithms to make iterative
  – Overheads of recursion
  – Recursive backtracking
    * General goals
      * The specific structure we used (next, extend, isFullSoln, reject)
  – Analyzing recursive methods with recursion trees
  – Processing arrays recursively by specifying bounds of subarray

Self-test questions: Chapters 9, 19

Exercises:
  – Chapter 9: 1, 2, 5, 8, 12
  – Chapter 19: 4, 6, 7
• Sorting
  Topics:
  - Simple sorts
    * Selection sort
    * Bubble sort
    * Insertion sort
  - Shell sort
  - Divide & conquer for sorting
    * Merge sort
    * Quicksort (including effect of pivot, pivot selection techniques)
    * Base cases
  - Sorting in-place vs. memory overhead
  - Stable sorting
  - Runtime analysis of sort methods

Self-test questions: Chapters 15, 16

Exercises:
  - Chapter 15: 1–3, 11, 13
  - Chapter 16: 1–4, 9

• List
  Topics:
  - List ADT and uses
  - Array vs. linked implementations of List (runtime, memory usage)
  - Circular linked list
  - Doubly-linked list

Self-test questions: Chapters 10–12

Exercises:
  - Chapter 10: 1, 2, 4, 5, 7, 8
  - Chapter 11: 1, 8, 9, 11
  - Chapter 12: 1, 2, 4, 15
• Iterator
  Topics:
  – Reasons for iterators
  – Iterator<T>, which classes should implement it, and why
  – Iterable<T>

  Self-test questions: Java Interlude 4, Chapter 13

  Exercises:
  – Chapter 13: 1–3, 10

• Dictionary and Hashing
  Topics:
  – Searching with sorted/unsorted array/linked chain
  – Dictionary ADT and uses
  – Simple Dictionary implementations based on simple searching
  – Analysis of simple Dictionary implementations
  – Hashing, main ideas
  – Requirements for a good hash function
  – Open addressing, linear probing and double hashing
  – Closed addressing
  – Using hashing to implement Dictionary
  – Comparative analysis of Dictionary implementations, including analysis with multiple variables

  Self-test questions: Chapters 19–23

  Exercises:
  – Chapter 19: 2, 5
  – Chapter 20: 1–3, 6
  – Chapter 21: 1, 2, 6
  – Chapter 22: 1, 10, 12
  – Chapter 23: 1, 4
• Queue
  Topics:
  – Queue ADT
  – Linked Queue implementation
  – Two-part circular linked Queue implementation
  – Logically-circular array Queue implementation
  – Analysis of Queue implementations

Self-test questions: Chapters 7, 8 (excluding priority queues)

Exercises:
  – Chapter 7: 1, 3
  – Chapter 8: 3–6, 10

• String Pattern Matching

  Note: Readings are provided as an excerpt from the CS 1501 textbook, *Algorithms* by Robert Sedgewick and Kevin Wayne, not our course textbook. You can find this excerpt on Canvas.

  Topics:
  – String pattern matching, main idea
  – Brute-force approach
  – Knuth-Morris-Pratt approach, including DFA simulation
  – Boyer-Moore approach, particularly mismatched character heuristic
  – Rabin-Karp approach using hashing

Exercises:
  – 5.3.3, 5.3.6, 5.3.11, 5.3.12