High-Level Learning Outcomes

- Emphasis on computational thinking and problem-solving. (Difficulty of problems will be increased, compared to CMSC 131.)
- Employ Systematic Program Design to go from a problem/concept to a well-designed, correct implementation.
- Test and debug small (2000 lines or less) programs.
- Master more advanced Java features.
  - Checked vs. unchecked exceptions
  - Enumerations
  - Inner and nested classes
  - Initialization blocks
  - Comparators, Comparable
  - Cloneable, clone
  - Iterators, Iterable
- Master the basics of algorithmic complexity, including big-O notation, and analyzing complexity of (non-recursive) code fragments. This topic will be emphasized throughout the course as new algorithms are learned and data structures are compared.
- Learn how inheritance is used in Object-Oriented programming and master the underlying intricacies.
- Develop a working knowledge of common Abstract Data Types and Data Structures, including an understanding of the advantages/disadvantages of each one, and the ability to choose an appropriate data structure for a particular problem.
- Practice creating concrete implementations of standard data structures. Typically, this will require implementations employing parameterized types.
- Master common algorithms associated with particular data structures. This will include both recursive and iterative solutions, as appropriate.
- Develop a familiarity with the Java Collections Framework, including basic use of generics. Learn how to choose an appropriate collection for a specific problem.
- Learn numerous sorting algorithms and develop an understanding for how/when to apply them.
- Learn basics of concurrent programming, including data races, deadlock, and synchronization.
- Learn to recognize basic algorithm strategies and to identify problems in which they could be employed.