Revised 8/22

NOVA COLLEGE-WIDE COURSE CONTENT SUMMARY
CSC 223 – DATA STRUCTURES AND ANALYSIS OF ALGORITHMS (4 CR.)

Course Description

Explores and contrasts data structures, algorithms for manipulating data structures, and their use and appropriateness in writing efficient real-world programming applications. Investigates implementations of different data structures for efficient searching, sorting, and other transformer operations. Third course in a three-course sequence. (CSC 221-222-223). Lecture 4 hours per week.

General Course Purpose

CSC 221, CSC 222, and CSC 223 comprise the standard sequence of minimal programming content for Computer Science majors. The course sequence will teach the students to use high-level languages and their applications to problem solve by using algorithms within procedural and object-oriented languages, while ensuring data adheres to a structured model. JAVA or C++ is the preferred language for this course, institutions may offer using a different language to align with primary 4-year partner requirements.

Course Prerequisites/Corequisites

Corequisite: CSC 208 or equivalent
Prerequisite: CSC 222 or by departmental consent

Course Objectives

Upon completing the course, the student will be able to:

Civic Engagement
- Engage and build technology that responds to human needs and helps people navigate institutional systems

Critical Thinking
- Assess why certain solutions might not work and to save time in coming up with a more efficient approach

Professional Readiness
- Work well with others and display situationally and culturally appropriate demeanor and behavior

Quantitative Literacy
- Perform accurate calculations, interpret quantitative information, apply and analyze relevant numerical data, and use results to support conclusions

Scientific Literacy
- Represent real-world objects and processes virtually by identifying properties, behavior, and operations relevant to solving problems on a computer.

Written Communication
- Develop, convey, and exchange ideas in writing, as appropriate to a given context and audience

Review of Object- Oriented Principles
- Compare and contrast procedural versus object-oriented programming
- Design class hierarchies using inheritance and interfaces
- Implement in code OOP constructs including encapsulation, inheritance, and polymorphism

- Review the design, implementation, and efficiency of recursive algorithms
- Review of arrays and exception handling

Analysis of Algorithms
- Discuss the differences between iterative vs. recursive algorithms
- Demonstrate worst-case complexity function
- Define other complexity functions such as best case, average case, and amortized.

Data Structures
- Describe and explain abstract data types including stacks, queues, singly and doubly linked list, sets, maps and graphs
- Compare and contrast contiguous and linked structures
- Explain the purpose and use of iterators
- Implement in code the various data structures using both contiguous and linked applications where applicable
- Analyze the time and space efficiency of data structures and algorithms and apply this analysis to select the best tools for solving problems.
- Explain how generics and parameterized types implement dynamic binding with polymorphism.

Searching and Sorting Algorithms
- Analyze a variety of algorithms for searching and sorting
- Classify the various sorting algorithms in terms of their Big-O analysis
- Implement both recursive and non-recursive algorithms for searches

Additional Data Structures
- Demonstrate the appropriate use of trees, graphs, sets, heaps, hash tables, and maps to computational problems
- Describe techniques to generate keys for hashed structures
- Discuss collision handling for hashing analysis
- Demonstrate the use of binary search trees
- Identify other types of tree data structures and their applications

Real-World Applications
- Create a solution to real-world computing problems by applying appropriate data structures.
- Employ best practices to design, document and implement the solution to a real-world application
- Make efficient use of formal testing and debugging.
- Apply the use of a version control system or a sandbox environment in team or multiple revision scenarios.
- Demonstrate proficiency in the use of programming languages to solve complex problems in a secure and robust manner.
- Discuss ethical aspects of programming and data handling.

Major Topics to be Included

- Review of Object- Oriented Principles
- Analysis of Algorithms
- Data Structures
- Searching and Sorting Algorithms
- Additional Data Structures
- Real-World Applications