HOUSATONIC COMMUNITY COLLEGE

Course Name: Data Structures

Course Number: CSC* E240

Credits: 3

Catalog description: The course focuses on fundamental data structures, which allow one to store collections of data with fast updates and queries. The course covers analysis and design of fundamental data structures and use of data structures as tools to algorithmically design efficient computer programs. Topics covered include managing complexity, abstraction, analysis, vectors, lists, queues, trees, sets, heaps, hash tables and maps, graphs, and recursion.

The course requires substantial hands-on use of computers in a computerized classroom environment.

Prerequisite: CSC* E224 ***or*** CSC* E218 ***and*** MAT* E 210

Corequisite or Parallel:

General Education Competencies Satisfied:

HCC General Education Requirement Designated Competency Attribute Code(s):

None

Additional CSCU General Education Requirements for CSCU Transfer Degree Programs:

None

Embedded Competency(ies):

None

Discipline-Specific Attribute Code(s):

☒ COMP Computer Science Elective

Course objectives:

General Education Goals and Outcomes:

None

Course Specific Objectives:

- Describe the properties, interfaces, and behaviors of basic abstract data types, such as collections, stacks, and queues.
- Analyze the asymptotic time complexity of the fundamental operations associated with a variety of data structures, such as vectors, linked lists, trees, and heaps.
- Analyze the space utilization of common data structures in terms of the long-term storage needed to maintain the structure, as well as the short-term memory requirements of fundamental operations, such as sorting.
- Design and implement general-purpose, reusable data structures that implement one or more abstractions.
- Compare and contrast the operation of common data structures (such as linear structures, priority queues, tree structures, hash tables, maps, and graphs) in terms of time complexity, space utilization, and the abstract data types they implement.
- Choose appropriate and efficient data structures and algorithms to solve a problem.
- Compare data structures and algorithms for efficiency using algorithm analysis and experiments.
- Apply algorithm analysis and knowledge of discrete mathematics to evaluate algorithms and data structures.
- Implement and use linear data structures, including stacks, queues, lists.
- Implement and use search structures and algorithms including binary search, search trees, and hash tables.
- Use and implement search data structures, including search trees and hash tables.
- Use and implement priority queues.
- Implement sorting algorithms and compare their performance analytically and empirically.
- Understand graphs and their representations; ability to implement graph search using BFS, DFS, and Dijkstra's Algorithm.
- Solve problems using references and dynamic memory allocation.
- Write recursive methods and understand when recursion is appropriate to a problem.

Course Content:

1. Self-referential classes and dynamic memory allocation
2. Recursion
3. Abstract data types: Set, List, Dictionary/Map
4. Linked lists, ADTs, classes, objects
5. Runtime analysis and big-O notation
6. Amortized runtime
7. Inheritance and polymorphism
8. Array lists and vectors
9. Stacks and queues
10. Iterators
11. Sorting algorithms
12. Search algorithms, sorted lists
13. Graphs and their uses
14. Graph search algorithms
15. BFS, DFS, and PageRank
16. Trees and their implementations
17. Tree traversals and search
18. Priority queues
19. Heaps
20. Balanced search trees
21. Hash tables and their analysis
22. Hash functions
23. Other graph algorithms
24. Dijkstra
25. Balanced binary search trees
26. AVL trees
27. Log-structured merge trees
28. Splay trees
29. Bloom filters
30. Tries
31. Suffix trees
32. Design patterns

Date Course Created:

Date of Last Revision: 01/01/2018