Raritan Valley Community College
Course Outline

CSIT 254 - Data Structures

I. Basic Course Information

A. Course number and Title: **CSIT 254- Data Structures**

B. New or Modified Course: **Modified**

C. Date of Proposal: Semester **Fall** Year: **2018**

D. Effective Term **Fall 2019**

E. Sponsoring Department **Mathematics and Computer Science**

F. Semester Credit Hours: **4**

G. Weekly Contact Hours: **5**
Lecture: __**3**__
Lab: __2__
Out of class student work per week: **__7__**

H. Prerequisites: **CSIT 105 - Foundations of Computer Science or**

GDEV 242 – Object Oriented Programming

I. Laboratory Fees: **Yes**

J. Name and E-Mail Address of Department Chair and Divisional Dean at time of approval:

Lori Austin –lori.austin@raritanval.edu (Chair);

Sarah Imbriglio –sarah.imbriglio@raritanval.edu (Divisional Dean)

II. Catalog Description

(Prerequisite/s: CSIT 105 - Foundations of Computer Science or GDEV 242 – Object Oriented Programming. This course introduces students to the fundamental data structures used in Computer Science. The data structures covered include linked lists, doubly linked lists, stacks, queues, trees, and graphs. Algorithms that manipulate these data structures are discussed and used in laboratory work. Students are introduced to the run-time analysis of algorithms and basic algorithms for searching and sorting. The Honors Option is available for this course.

III. Statement of Course Need

A. This course introduces some of the most fundamental concepts in Computer Science - how data is represented and manipulated in software.

This course addresses a need for currently employed software developers who wish to upgrade their skills.

The organization of Data Structures follows the recommendations made by the Association for Computing Machinery.

B. This course has a weekly lab component. The lab is essential for providing students hands on programming to write Java programs to implement and use Data Structures

C. This course generally transfers as a Computer Science Requirement

IV. Place of Course in College Curriculum

A. Free Elective
B. This course serves a Programming Elective on the Computer and Programming Electives List
C. This course serves a Computer Elective on the Computer and Programming Electives List
D. This course meets a Requirement in
    1. Computer Science A.S. Degree
    2. Engineering Science, Electrical Track A.S. Degree
    3. Information Systems & Technology A.S. Degree
    4. Information Systems & Technology A.A.S. Degree
    5. Game Development A.A.S Degree
E. To see course transferability: a) for New Jersey schools, go to the NJ Transfer website, www.njtransfer.org; b) for all other colleges and universities, go to the individual websites.

V. Outline of Course Content

This course explores the following topics:

1. The Software Development Life Cycle
2. Run-Time Analysis (Big-O( ) Analysis)
3. Data Abstraction
4. Linked Lists and Doubly Linked Lists
5. Stacks
6. Queues
7. Recursion / Recursion on Data Structures
8. Trees / Binary Search Trees
9. Searching (Hash Tables)
10. Sorting (Recursive Merge Sort)

VI. General Educational and Course Learning Outcomes

A. General Education Goals

After completion of this course, students will be able to:

1. design and develop data structures that efficiently address program requirements (GE-NJ 2, GE-NJ 4)

2. analyze the data structures used in computer applications and the issues surrounding their implementation (GE-NJ 2, GE-NJ 4)

3. apply quantitative reasoning to analyze the performance of data structure algorithms in order to efficiently solve problems (GE-NJ 2)

B. Course learning outcomes:

At the conclusion of the course, students will be able to:

1. compare and contrast the basic data structures used in Computer Science: lists, stacks, queues, trees and graphs
2. identify and implement the basic operations for manipulating each type of data structure
3. create data structures using Java
4. analyze the run-time analysis of algorithms and express them using O ( ) notation
5. apply recursion to data structure operations
6. identify the appropriate data structure for a given problem
7. analyze algorithms to search or sort the data in various data structures (arrays, queues, stacks, etc.) and interpret their run-time performance
8. create and execute test plans which include the testing of boundary conditions

C. Assessment Instruments

A. Homework Assignments
B. Programming Labs
C. Programming Projects
D. Exams

VII. Grade Determinants

A. Homework Assignments
B. Programming Labs
C. Programming Projects
D. Midterm Exam

E. Final Exam

Primary formats, modes, and methods for teaching and learning that may be used in the course:

A. Lecture/Discussion -- New concepts are introduced in interactive lectures.
B. Laboratory -- Students apply new concepts in the laboratory. This also allows for one-on-one instruction and assistance.

VIII. Text and Materials

Suggested Textbook—Main, Michael. Data Structures & Other Objects Using JAVA, 4th Edition, Pearson, 2012.

(Please note: The course outline is intended only as a guide to course content and resources. Do not purchase textbooks based on this outline. The RVCC Bookstore is the sole resource for the most up-to-date information about textbooks.)

IX. Resources

A. Computer Lab for classroom instruction and exercises utilizing Java SDK Version 7 or higher and an IDE such as Notepad++, TextPad, SciTe, NetBeans, or Eclipse)
B. Access to Internet for Java Documentation at www.oracle.com

X. Honors Option

Definition: Prerequisites: Minimum GPA of 3.5 or permission of the instructor. Students pursuing the Honors Option will be required to demonstrate a higher level of knowledge and skill in each of their course programming projects. They will be required to take the concepts introduced and generalize them for broader application. Students will also do independent work researching application programming interfaces.

A. Educational Goals and Learning Outcomes:

General Education Goal - After completion of this course, the student will be able to:

Independently research application programming interfaces (G.E. 3)

Student Learning Outcomes – At the conclusion of this course, students will be able to:

a. Implement methods for manipulating complex, compound data structures

b. Apply data structure concepts to new contexts

B. Honors Option Content

In each of the programming assignments, Honors Option students will be given additional requirements to complete. These requirements will require the students to take the new concepts and apply them in ways that demonstrate a deeper understanding of the material. In addition, some projects will require students to work independently to find and use application programming interfaces that are not introduced in class lecture.

For each of the major programming assignments, here are possible honors option additions for each project. Each student may do one or more of the following:

- Run Time Analysis Project – create a new program that produces the same output with run time O(1)
- Finite State Machine Project – create a program so that the combination lock boundary conditions are variable
- Linked List Project – extend linked list data structures to have the ability to have multiple, simultaneous lists
- Stacks and Queues Project – extend system to allow for a variable number of stacks or queues
- Recursion Project – define and create their own fractal
- Binary Tree Program – extend program to allow for multiple trees. Create common, general methods to handle similar user actions. Add ability to merge trees
- Graphs – implement a sparse graph using linked lists as opposed to an adjacency matrix
- Graph Program – add additional data structures to the graph. Extend the user interface to include graphics or audio

C. Assessment Instruments for Honors Option Work

Honors Option students will be assessed for their ability to deliver the additional requirements in programming projects and performance on quizzes and exams. In addition, students will be assessed on their participation in interactive programming discussions and their contributions to un-graded group projects.

D. Grade Determinants for Honors Option Work

The final grade will be based upon students delivering the required additional programming functionality, grades in quizzes, exams, and programming assignments, as well as participation in classroom discussion and group work.