UNIVERSITY OF TECHNOLOGY, JAMAICA
SYLLABUS OUTLINE

COLLEGE/FACULTY:	**Faculty of Engineering & Computing**
SCHOOL/DEPT:	**School of Computing & Information Technology**
COURSE OF STUDY:	**Bachelor of Science in Computing**
LEVEL:	**Two (2)**
MODULE TITLE:	**Data Structures**
MODULE CODE:	**CMP2006**
DURATION (Credit Hours):	**Ninety (90)**
CREDIT VALUE:	**Four (4)**
PREREQUISITES:	**Object Oriented Programming (CIT2004)**

1.0 MODULE DESCRIPTION

This module addresses aspects of the development and use of data structures and algorithms in computer applications. It emphasizes the specification of data structures as abstract data types using pseudo-code. It introduces data structures such as linked lists, stacks, queues and trees, while examining the complexity and efficiency of commonly used algorithms

2.0 MODULE OBJECTIVES

Upon Completion of this module, the student should:

i. understand the use of Abstract Data Types (ADTs)
ii. understand the purpose of various data structures
iii. apply iteration and recursion methodologies
iv. assess the efficiency of sorting & searching algorithms
v. evaluate efficiency of algorithms and code usingcomplexity analysis
vi. implement appropriate data structures to solve computer problems

3.0 MODULE CONTENT

UNIT 1 Introduction to Data Types and Data Structures (2hrs Lecture + 1hr Tutorial + 3hrs Practical)

Expected Learning Outcomes:
Upon completion of Unit 1, the student should be able to:

1.1 explain the need for structured data
1.2 differentiate between data types
1.3 describe the concept of an ADT
1.4 convert between the main number systems used in computing
1.5 prepare a semiformal specification of an ADT
1.6 implement a simple ADT in a supporting programming language

Content
- Data structures
- Data Types
  - Atomic and Composite data types
- Abstract Data Type
  - Definition
  - Use
- Number System Conversion

UNIT 2 Code Repetition – Iteration & Recursion (2hrs Lecture + 1hr Tutorial + 3hrs Practical)

Expected Learning Outcomes:
Upon completion of Unit 2, the student should be able to:

2.1 discuss the need to repeat code
2.2 discuss the different types of iteration loops
2.3 differentiate between iteration and recursion
2.4 identify the parts of an iterative and recursive block of code
2.5 explain the benefits and drawbacks of iteration and recursion
2.6 design iterative and recursive solutions for computer problems
2.7 implement iterative and recursive algorithms

Content
- Iteration
  - Types of Loops

    - Parts of an Iterative function
- Recursion
    - Parts of a Recursive function
    - Recursive algorithms
        - Factorial
        - Fibonacci Numbers
        - The Towers of Hanoi

UNIT 3 Introduction to Complexity Analysis
(2hrs Lecture + 1hr Tutorial + 3hrs Practical)

Expected Learning Outcomes:
Upon completion of Unit 3, the student should be able to:

3.1 explain the need for complexity analysis
3.2 discuss the various methods in measuring efficiency
3.3 identify challenges in measuring efficiency
3.4 explain the purpose of Asymptotic Analysis (Upper Bound using Big-O Notation)
3.5 distinguish amongst Big-O Notations
3.6 perform Big-O efficiency analysis ongiven algorithms and code snippets
3.7 evaluate mathematical expressions to derive the Asymptotic Upper Bound
3.8 compute the time taken to run code snippets

Content
- Measuring Efficiency
    - Time
    - Space
- Complexity Analysis
- Asymptotic Analysis
    - Big-O Analysis
    - Big-O Notations
- Timers

UNIT 4 Searching and Sorting
(4hrs Lecture + 2hrs Tutorial + 6hrs Practical)

Expected Learning Outcomes:
Upon completion of Unit 4, the student should be able to:

4.1 describe how popular search algorithms work

4.2 determine the most appropriate searching algorithmbased on the type of list to be searched
4.3 explain how popular sorting algorithms work
4.4 evaluate the efficiency of searching and sorting algorithms using Big-O analysis
4.5 implement searching and sorting algorithms with an array

Content

- Searching Algorithms
  - Linear/Sequential
  - Binary Search
  - Hashed Search
- Sorting Algorithms
  - Insertion Sort
  - Selection Sort
  - Bubble Sort
  - Quick Sort

UNIT 5 The Linear List as an ADT - Arrays & Linked Lists (4hrs Lecture + 2hrs Tutorial + 9hrs Practical)

Expected Learning Outcomes:
Upon completion of Unit 5, the student should be able to:

5.1 describe the linear list as an ADT
5.2 describe the two (2) categories of linear lists
5.3 explain the various operations of linear lists
5.4 describe how data is stored in an array
5.5 distinguish between row and column major order
5.6 discuss the drawbacks of an array
5.7 describe advantages of using a linked list over an array
5.8 compare the various types of linked lists
5.9 identify the applications of linked lists
5.10 calculate the size of an array
5.11 compute the memory address and position of each element in an array
5.12 evaluate the efficiency of the various operations of arrays and linked lists
5.13 draw a pictorial representation of an array and linked list
5.14 implement the various operations of the linear lists

Content

- Linear Lists
  - ADT specification
  - Types of Linear Lists
    - Arrays

        - Linked Lists
- Linear List Categories
    - General
    - Restricted
- Linear List operations
- Arrays
    - Operations
    - Storage
        - Memory Address
        - Memory Position
        - Row & Column Major Order
- Linked List
    - Operations
    - Types
        - Singly
        - Doubly
        - Circular

UNIT 6 The Stack as an ADT
(2hrs Lecture + 1hr Tutorial + 3hrs Practical)

Expected Learning Outcomes:
Upon completion of Unit 6, the student should be able to:

6.1 explain the concept of the stack ADT
6.2 describe the various operations of astack
6.3 discuss the various applications of a stack
6.4 assess the efficiency of the stack operations
6.5 draw a pictorial representation of a stack
6.6 implement stacks to solve an appropriate computer problem

Content
- Stacks
    - ADT specification
- Applications of a Stack
- Operations of a Stack

UNIT 7 The Queue as an ADT
(2hrs Lecture + 1hr Tutorial + 3hrs Practical)

Expected Learning Outcomes:

Upon completion of Unit 7, the student should be able to:

7.1 explain the concept of the queue ADT
7.2 explain the various operations of a queue
7.3 discuss the various applications of a queue
7.4 describe queuing theory
7.5 assess the efficiency of the queue operations
7.6 draw pictorial representations of a queue
7.7 implement queues to solve an appropriate computer problem

Content

- Queues
  - ADT specification
- Applications of a Queue
- Introduction to Queuing Theory
- Operations of a Queue

UNIT 8 Trees as an ADT
(4hrs Lecture + 2hrs Tutorial + 6hrs Practical)

Expected Learning Outcomes:
Upon completion of Unit 8, the student should be able to:

8.1 identify the parts of a tree
8.2 describebasic terminologies associated with trees
8.3 differentiate between general and binary trees
8.4 explain the purpose of expression trees
8.5 describe the operations that may be performed on a tree
8.6 demonstrate the various methods of tree traversal
8.7 assess the efficiency of tree operations
8.8 sketch various types of trees
8.9 construct expression trees from a mathematical expression
8.10 implement trees to solve computing problems

Content

- Trees
  - Parts of tree (Root, Node, Vertices)
  - Key terms (Level, Height, Child, Parent, Balance, Complete)
- Types of trees
  - Binary
  - General

    - Expression
- Tree Operations
- Expression Trees
    - Purpose
    - Traversal (Prefix, Infix & Postfix)
- Tree traversal
    - Preorder
    - Inorder
    - Postorder

UNIT 9 Search Trees
(4hrs Lecture + 2hrs Tutorial + 3hrs Practical)

Expected Learning Outcomes:
Upon completion of Unit 9, the student should be able to:

9.1 describe the different types of search trees
9.2 discuss the structure of search trees
9.3 explain the various operations of search trees
9.4 discuss the various applications of search trees
9.5 determine the efficiency of search tree operations
9.6 implement appropriate search trees to solve computer problems

Content
- Types of Search Trees
    - Binary Search Tree (BST)
    - Adelson-Velskii and Landis (AVL)
    - Heaps
- Search Tree Operations
- Applications of search trees

4.0 LEARNING AND TEACHING APPROCHES
- Lectures to provide coverage of the concepts and techniques to generate understanding.
- Tutorials to provide opportunities for a high level of interaction and discussion of the concepts.

- Practical sessions to facilitate demonstration, practice and application of techniques, to develop students' competence in the use of the supporting programming language tools and to broaden understanding of the concepts covered in the module.
- Student participation in labs and tutorials to create an environment in which all students have the opportunity to learn, develop insight into the material and provide useful critique and feedback on the issues explored, ultimately providing a rich overall learning experience.
- Case discussion and analyses to provide real-world understanding, critical thinking and discussion of issues related to data structures.
- Supplemental lecture, tutorial and practical notes will be given to all students.

5.0 ASSESSMENT PROCEDURES

Coursework **70%**

1. *Programming Project* 20%

This is a group project ranging from 3-4 members which aims to assess the student's ability to design a solution to a problem using the techniques taught and implement the solution using the supporting programming language.

2. *Lecture Tests 1* 10%

This theory tests provide the opportunity to assess the knowledge and competencies expressed in the learning outcomes for units 1-5.

3. *Lecture Tests 2* 10%

This theory tests provide the opportunity to assess the knowledge and competencies expressed in the learning outcomes for units 6-9.

4. *Lab Tests 1* 10%

This lab test covers units 1-5 and assesses the student's ability to implement solutions using appropriate data structures.

5. *Lab Test 2* 10%

This lab test covers units 6-9 and assesses the student's ability to implement solutions using appropriate data structures.

6. Participation 10%

This is a continuous assessment that evaluates the students' participation in weekly lab discussions and activities.

Final Exam **30%**

This exam is designed to assess students' mastery of the learning outcomes covered in all units.

6.0 BREAKDOWN OF HOURS

• Classroom Lecture	***26 hours***
• Classroom Tutorial	***13 hours***
• Classroom Practical	***39 hours (13hrs Supervised)***
• Assessments	***12 hours***
• **TOTAL**	***90 hours***

7.0 TEXTBOOKS AND REFERENCES

Required Text

- *Weiss, M. (n.d.). Data Structures and Algorithm Analysis in Java*. Pearson Education, Inc.

Recommended Text:

- *Deitel, P., &Dietel, H. (n.d.). C++ How to Program,* Latest Edition.Prentice Hall.
- *Gilberg, G., Forouzan, B. Data Structures: A Pseudocode Approach with C/C++*, Latest Edition. PWS Publishing Company.
- *Weiss, M. Data Structures and Problem Solving Using C++, 2nd Edition*. Pearson Education.

8.0 NAME/S OF SYLLABUS WRITER/S OR DEVELOPER/S

LalithaJonnalagadda

8.1 DATE OF PRESENTATION **March 2018**

8.2 NAME/S OF SYLLABUS REVIEWER/S

LalithaJonnalagadda and Christopher Slowley

David White

Tyrone Edwards **June 2011**

Philip Smith **March 2018**

8.3 **DATE/S OF REVISION** ....

9.0 APPROVAL

9.1 **PROGRAMME DIRECTOR (PD)**

....

9.2 **SIGNATURE OF PD** ....

9.3 **COLLEGE/FACULTY CURRICULUM COMMITTEE**

....

9.4 **SIGNATURE OF COMMITTEE CHAIR** ....

10.0 ACCEPTANCE BY OFFICE OF CURRICULUM DEVELOPMENT & EVALUATION (OCDE)

....

10.1 **DATE OF ACCEPTANCE BY OCDE** ....