<table>
<tr><td>1.</td><td>Name of Course</td><td colspan="6">Data Structure & Algorithm</td></tr>
<tr><td>2.</td><td>Course Code</td><td colspan="6">CCPS2553</td></tr>
<tr><td>3.</td><td>Name(s) of academic staff</td><td colspan="6"></td></tr>
<tr><td>4.</td><td>Rationale for the inclusion of the course/module in the programme</td><td colspan="6">Faculty
Data structures and algorithms lie at the heart of Computer Science as they are the basis for the efficient solution of programming tasks. Students of computing are required to be aware of the large number of ways in which data is represented and manipulated by the machine. Previous study of programming have developed the concept of data type - primitive types, integers and floating point numbers, characters - as well as simple, static, structures such as arrays, strings and records. This module will extend this understanding. It will examine the process of creating, and exploiting more flexible data memory structures such as, linked lists, stacks, queues and trees.</td></tr>
<tr><td>5.</td><td>Semester and Year offered</td><td colspan="6">1/2</td></tr>
<tr><td rowspan="3">6.</td><td rowspan="2">Total Student Learning Time (SLT)</td><td colspan="4">Face to Face</td><td colspan="2">Total Guided and Independent Learning</td></tr>
<tr><td>L = Lecture
T = Tutorial
P = Practical
O= Others</td><td>L</td><td>T</td><td>P</td><td>O</td><td colspan="2">Independent = 84
Total =126</td></tr>
<tr><td>28</td><td></td><td>28</td><td></td><td colspan="2"></td></tr>
<tr><td>7.</td><td>Credit Value</td><td colspan="6">3</td></tr>
<tr><td>8.</td><td>Prerequisite (if any)</td><td colspan="6">CCP0100 Object Oriented Programming</td></tr>
<tr><td>9.</td><td colspan="7">Objectives:
• To promote a good understanding of different forms of data structure representation.
• Students, during this course, are trained to deal with Abstract Data Types (ADT) and to implement different data structures using different methods.</td></tr>
<tr><td>10.</td><td colspan="7">Learning outcomes:
Upon successful completion of this course, students will have the skills to deal with ADT and implement data structures using different methods. Students :
• Should be able to know basic terms associated with data structures, and the main principles that each deal by such as Stacks, Queues, Lists and Strings.
• Should be able to choose suitable data structure that use for any applications at hand.
• Should be able to deal with Searching and Sorting Algorithms.
• Should be able to use and create data structure by C++ language.</td></tr>
<tr><td>11.</td><td colspan="7">Transferable Skills:
Know terms associated with data structures and choose suitable data structure that use for different applications.</td></tr>
</table>

12. **Teaching-learning and assessment strategy**

A variety of teaching and learning strategies are used throughout the course, including:

- Classroom lessons. Lectures and Power Point presentations
- Laboratory sessions: Practice exercises
- brainstorming;
- student-Lecturer discussion
- collaborative and co-operative learning;
- Independent study.

Assessment strategies include the following:

- Ongoing quizzes
- Midterm tests
- Performance Assessment (mini-project, Assigned exercises, Presentation and final project)
- Lecturer Observation

13. **Synopsis:**

Students of computing are required to be aware of the large number of ways in which data is represented and manipulated by the machine. The aim of this course is to provide students with the theoretical background and practical experience relating to the design and implementation of several types of data structures.

14. **Mode of Delivery:**

Class Lectures and Laboratory sessions

15. **Assessment Methods and Types:**

The assessment for this course will be based on the following:

Coursework **50%**

- Quizzes 10%
- Project and assignments 20%
- Mid-Semester Exam 20%

Final Examination **50%**

100%

16. **Mapping of the course/module to the Programme Aims**

A1	A2	A3	A4	A5	A6	A7	A8	A9
4	0	0	0	3	1	1	2	0

17. **Mapping of the course/module to the Programme Learning Outcomes**

LO1	LO2	LO3	LO4	LO5	LO6	LO7	LO8	LO9	LO10	LO11	LO12
5	0	0	0	0	0	1	2	4	0	1	0

18. **Content outline of the course/module and the SLT per topic**

	Details	SLT			
		L	P	Indep.	Total
Topic 1	• Course Introduction • C++ Revision (Classes, inheritance, overloading) • Introduction Data Structure and types	2	2	6	10
Topic 2	Introduction to Stack o Advantages o Disadvantages o Algorithm o Implementation o Examples	2	2	6	10

	Topic 3	Introduction to Queue and Simulation o advantages o disadvantages o Algorithm o implementation o Examples	4	4	12	20
	Topic 4	Introduction to Linked Stack o advantages o disadvantages o Algorithm o implementation o Examples Introduction to Linked Queue. o advantages o disadvantages o Algorithm o implementation o Examples	4	4	12	20
	Topic 5	Introduction to Recursion. o advantages o disadvantages o Algorithm o implementation o Examples	2	2	6	10
	Topic 6	Introduction to Lists o advantages o disadvantages o Algorithm o implementation o Examples	2	2	6	10
	Topic 7	Introduction to Strings o advantages o disadvantages o Algorithm o implementation o Examples	2	2	6	10
	Topic 8	Introduction to Trees. o advantages o disadvantages o Algorithm o implementation o Examples	2	2	6	10

	Topic 9	Searching and Sorting Algorithms ○ Sequential search ○ Binary search ○ Insertion sort ○ selection sort ○ shell sort ○ quick sort ○ merge sort ○ Radix sort ○ Hashing Note: For each Searching and Sorting Algorithms explain ○ advantages ○ disadvantages ○ Algorithm ○ Implementation ○ Examples	8	8	24	40
		Total	**28**	**28**	**84**	**140**
	Laboratory	Laboratory Details Exercises based on topics covered in each lecture. Experiment work must include the following • C++ Revision (Classes, inheritance, overloading) • Data Structure types • Stack Implementation • Queue Implementation • Linked Stack Implementation • Linked Queue Implementation • Recursion Implementation • Lists Implementation • Strings Implementation • Trees Implementation • Sorting Algorithms Implementation • Min-projects				
19.	**Main references supporting the course:** 1. Augenstein & Tenenbaum Langsam, **Data Structure Using C & C++,** 2nd Ed (Paperback - 2007) 2. Ellis Horowitz, Sartaj Sahni, and Susan Anderson-Freed, **Fundamentals of Data Structures in C** (2007)					
	Additional references supporting the course: 1. Larry Nyhoff, ADTs, **Data Structures and Problem Solving with C++**, 2nd Ed. , Printice Hall, 2005. 2. Malik, **Data Structures using C++**, Thomson , 2003. 3. Clifford Shaffer, **A practical Introduction to Data Structures and Algorithm Analysis**, 2nd Ed. , Prentice Hall, 2001. 4. Frank M. Carrano, **Data Abstraction & Problem Solving with C++** (5th Edition) (2006) 5. Frank M. Carrano, **Data Abstraction and Problem Solving with C++: Walls and Mirrors** (4th Edition) by (2004)					
20.	**Other additional information** All materials will be available to the students online.					