Course Specification
(Bachelor)

Course Title:	Data Structure and Algorithms with Java
Course Code:	CS2105
Program:	Bachelor of Computer Science BSc in Cybersecurity BSc in Software Engineering
Department:	Computer Science and Artificial Intelligence
College:	Computing
Institution:	Umm Al Qura University
Version:	1.0
Last Revision Date:	22/04/2025

Table of Contents

A. General information about the course: 3

Course Learning Outcomes (CLOs), Teaching Strategies and Assessment Methods (Bachelor of Cybersecurity) 5

C. Course Content 7

D. Students Assessment Activities 9

E. Learning Resources and Facilities 9

F. Assessment of Course Quality 10

G. Specification Approval 10

A. General information about the course:

1. Course Identification

1. Credit hours: (3)					
2. Course type					
A.	☐University	☐College	☒ Department	☐Track	☐Others
B.	☒ Required		☐Elective		
3. Level/year at which this course is offered: **BSc in Computer Science; BSc in Cybersecurity AND BSc in Software Engineering - (2nd year/ 3rd level)**					
4. Course General Description:					
This course provides theoretical and practical knowledge of fundamental data structures and algorithms using Java programming language. This course covers time and space complexity analysis which allows students to assess and select the appropriate algorithms and implementation of data structures to a certain problem. Topics covered include linked-list, stacks, queues, lists, trees, sets, maps hash tables and graphs. These data structures are explained using basic sorting and searching techniques with brief overview of recursion and memory management. The course also explores the implementation of a range of data structures in the Java programming language. The knowledge and practice of these structures are of utmost importance. It will make the students able to organize, represent and manipulate the data, which is central to computing.					
5. Pre-requirements for this course (if any)**:**					
BSc in Computer Science and BSc in Cybersecurity: CS1704 - Object oriented programming with java **BSc in Software Engineering:** SE1101 - Object-oriented Application Development					
6. Co-requisites for this course (if any)**:**					
NA					
7. Course Main Objective(s):					

On successful completion of this course students will be able to:

- Explain fundamental computing algorithms
- Explain the key sorting algoritems
- Explain the key concepts of data structures and how they operate including Linked Lists, stack, queues, list, sets, maps, trees, heaps, hash tables, graph.
- Choose the most appropriate algorithms and data structure for a given problem.
- Build and use these sorting and searching algorithms , and data structures in code.

2. Teaching mode (mark all that apply)

No	Mode of Instruction	Contact Hours	Percentage
1	Traditional classroom	60	100%
2	E-learning		
3	Hybrid • Traditional classroom • E-learning		
4	Distance learning		

3. Contact Hours (based on the academic semester)

No	Activity	Contact Hours
1.	**Lectures**	30
2.	**Laboratory/Studio**	30
3.	**Field**	
4.	**Tutorial**	
5.	**Others (specify)**	
Total		60

Course Learning Outcomes (CLOs), Teaching Strategies and Assessment Methods (Bachelor of Science in Computer Science)

Code	Course Learning Outcomes	Code of PLOs aligned with the program	Teaching Strategies	Assessment Methods
1.0	**Knowledge and understanding**			
1.1	Explain the fundamentals of algorithms analysis	K1	Lectures and discussions	Quizzes, lab exercises, assignments, and exams

Code	Course Learning Outcomes	Code of PLOs aligned with the program	Teaching Strategies	Assessment Methods
1.2	Acquire the key concepts of data structures	K1	Lectures and discussions	Quizzes, lab exercises, assignments, and exams
2.0	**Skills**			
2.1	Demonstrate fundamental knowledge of data structure and its implementation.	S2	Lectures and discussions	Lab exercises and assignments, and exams
2.2	Design and implement and choose the most appropriate data structure for a certain problem.	S2	Lectures	Lab exercises and assignments, and exams
2.3	Assess and select an algorithm and justify the selection, and then apply the algorithm to a certain problem.	S1	Lectures	Lab exercises and assignments, and exams
3.0	**Values, autonomy, and responsibility**			

Course Learning Outcomes (CLOs), Teaching Strategies and Assessment Methods (BSc in Cybersecurity)

Code	Course Learning Outcomes	Code of PLOs aligned with the program	Teaching Strategies	Assessment Methods
1.0	**Knowledge and understanding**			
1.1	Explain the fundamentals of algorithms analysis	K1	Lectures and discussions	Quizzes, lab exercises, assignments, and exams
1.2	Acquire the key concepts of data structures	K1	Lectures and discussions	Quizzes, lab exercises, assignments, and exams

2.0	**Skills**			
2.1	Demonstrate fundamental knowledge of data structure and its implementation.	S2	Lectures and discussions	Lab exercises and assignments, and exams
2.2	Design and implement and choose the most appropriate data structure for a certain problem.	S2	Lectures	Lab exercises and assignments, and exams
2.3	Assess and select an algorithm and justify the selection, and then apply the algorithm to a certain problem.	S1	Lectures	Lab exercises and assignments, and exams
3.0	**Values, autonomy, and responsibility**			

Course Learning Outcomes (CLOs), Teaching Strategies and Assessment Methods (BSc in Software Engineering)

Code	Course Learning Outcomes	Code of PLOs aligned with the program	Teaching Strategies	Assessment Methods
1.0	**Knowledge and understanding**			
1.1	Explain the fundamentals of algorithms analysis	K1	Lectures and discussions	Quizzes, lab exercises, assignments, and exams
1.2	Acquire the key concepts of data structures	K3	Lectures and discussions	Quizzes, lab exercises, assignments, and exams
2.0	**Skills**			
2.1	Demonstrate fundamental knowledge of data structure and its implementation.	S1	Lectures and discussions	Lab exercises and assignments, and exams
2.2	Design and implement and choose the most appropriate data	S2	Lectures	Lab exercises and assignments, and exams

	structure for a certain problem.			
2.3	Assess and select an algorithm and justify the selection, and then apply the algorithm to a certain problem.	S3	Lectures	Lab exercises and assignments, and exams
3.0	**Values, autonomy, and responsibility**			

C. Course Content

No	List of Topics	Contact Hours
1.	**Introduction to Data Structures:** – Overview of data structures and algorithms. – Python's built-in data structures (lists, tuples, dictionaries). – Review Arrays (static vs dynamic arrays). – Numeral algorithms: basic arithmetic and modular arithmetic. **Lab:** – Exercises in numeral algorithms (e.g., computing the average of a list of numbers, finding the minimum, maximum, and mode in a list, approximating the square root of a number, or finding the greatest common divisor GCD). Implement linear search in Python.	**4**
2.	**Runtime and Memory Efficiency:** – Introduction to time complexity and Big-O notation. – Revisiting runtime and memory efficiency for all covered algorithms. – Trade-offs between time and space in practical implementations. – Introduction to amortized analysis for data structures (dynamic arrays, Fibonacci tree). **Lab:** – Analyze the efficiency of various data structure implementations. – Optimize a given algorithm for both runtime and memory. Comparison of sorting algorithms (runtime, memory efficiency).	**8**
3.	**Linked Lists:** – Linked lists: singly, doubly, and circular linked lists. – Comparison of arrays and linked lists. – Operations: insertion, deletion, traversal. **Lab:** – Implement linked lists in Python. Solve problems like reversing a linked list and detecting cycles.	**4**

4.	**Stacks and Queues:** – Stack operations and applications: expression evaluation, function call stacks. – Queue types: linear, circular, and double-ended queues (deque). – Applications in scheduling and buffering. **Lab:** – Implement stacks and queues in Python. Solve real-world scenarios like balanced parentheses and task scheduling.	**4**
5.	**Hash Tables:** – Hash functions and their properties. – Collision resolution strategies: chaining, open addressing. – Efficiency of hash tables in searching and insertion. **Lab:** – Implement a hash table from scratch in Python. Solve problems like word frequency counting and anagrams detection.	**4**
6.	**Trees:** – Binary trees: structure and traversal techniques. – Binary Search Trees (BST): insertion, deletion, and search. – Balanced trees: AVL trees, and rotations. – Heaps: min-heap and max-heap, Fibonacci Trees. **Lab:** – Implement binary trees and BSTs in Python: (discuss result with sorted and unsorted data). – Implement heaps and perform heap sort in Python. Solve problems like finding the k largest/smallest elements.	**12**
7.	**Recursion and String-Matching Algorithms:** – **Recursion:** ▪ Basics of recursion. **Lab:** Solve problems using recursion (e.g., Fibonacci, factorial).	**4**
8.	**Sorting Algorithms:** – Basics of sorting algorithms: define internal/external sort and in-place sort. – Sorting Algorithms Examples: (quicksort, heapsort, merge sort, selection sort, insertion sort algorithms). **Lab:** – Implement basic sorting algorithms in Python. Solve real-world sorting problems.	**12**
9.	**Fundamental Graph Algorithms:** – Fundamental definitions and applications, bipartite graphs, strongly connected components, and topological sorting.	**8**

	– Graph representation: adjacency matrix and adjacency list. – Graph traversal: Breadth-First Search (BFS), and Depth-First Search (DFS). **Lab:** – Implement BFS and DFS in Python. Solve problems like detecting connected components and cycles.	
Total		**60**

D. Students Assessment Activities

No	Assessment Activities *	Assessment timing (in week no)	Percentage of Total Assessment Score
1.	**Assignments and Quizes**	1-15	30%
2.	**Midterm Exams**	7-13	30%
3.	**Theoretical Final Exam**	16-18	40%

*Assessment Activities (i.e., Written test, oral test, oral presentation, group project, essay, etc.).

E. Learning Resources and Facilities

1. References and Learning Resources

Essential References	M. Goodrich, R. Tamassia, M. Goldwasser, Data structures and algorithms in Java, 6th Edition, Goldwasser, 2014.
Supportive References	Book and Slides
Electronic Materials	Slides and related handouts
Other Learning Materials	

2. Required Facilities and equipment

Items	Resources
facilities (Classrooms, laboratories, exhibition rooms, simulation rooms, etc.)	Classroom & Laboratory
Technology equipment (projector, smart board, software)	Overhead projector and smart board. Integrated Development Environment (e.g., NetBeans, Eclipse, JBuilder). Java Development Kit (JDK)
Other equipment (depending on the nature of the specialty)	

F. Assessment of Course Quality

Assessment Areas/Issues	Assessor	Assessment Methods
Effectiveness of teaching	Students	Indirect: a student-feedback form is distributed at the end of the course.
Effectiveness of Students assessment	Quality Assurance committee	Direct: review and analyze assessment methods and materials.
Quality of learning resources	Coordinator and curriculum committee	Indirect: course materials are regularly reviewed in order to keep it updated.
The extent to which CLOs have been achieved	Coordinator and Quality Assurance committee	Direct: analyze the grades of students in each assessment method to find out which CLOs are below 60%.
Other		

Assessors (Students, Faculty, Program Leaders, Peer Reviewers, Others (specify)

Assessment Methods (Direct, Indirect)

G. Specification Approval

Council / Committee	**Computer Science and Artificial Intelligence Department Council**
Reference No.	**THE 16TH MEETING FOR THE ACADEMIC YEAR 1446H**
Date	**22/04/2025**

Council /Committee	**CYBERSECURITY DEPARTMENT COUNCIL**
Reference No.	**THE 16TH MEETING FOR THE ACADEMIC YEAR 1446H**
Date	**22/04/2025**

Council /Committee	**SOFTWARE ENGINEERING DEPARTMENT COUNCIL**
Reference No.	**THE 17TH MEETING FOR THE ACADEMIC YEAR 1446H**
Date	**22/04/2025**