Data Structures and Algorithms I

Module Description

Module code	CS103-S27	**Credits**	7 ECTS
Programme	Foundation programme	**Level**	Semester 2
Semester	Spring 2027	**Prerequisites**	CS101-F26 Introduction to Informatics; MAT101-F26 Discrete Structures
Module leader	Dr. Ali Khudiyev	**Email**	ali.khudiyev@wosm.academy
Office hours	Monday–Friday, 10:00–17:00 (Baku time, UTC+4)		

1. Module Overview

Data Structures and Algorithms I is the first of two modules on the design and analysis of algorithms. It teaches you to reason precisely about two questions that every program raises: *is it correct?* and *how much time and memory does it need?* You will study the fundamental data structures and the classic algorithms for sorting and searching, and learn to choose between them by comparing the cost of their operations.

The module builds directly on Semester 1. Programming is done in C. C exposes how data structures are laid out in memory, which makes the cost of each operation visible; the first units introduce C to students who learnt to program in Java in Introduction to Informatics. Proofs by induction, asymptotic reasoning and basic combinatorics come from Discrete Structures. Data Structures and Algorithms II (Semester 3) continues with graph algorithms and general design techniques.

2. Aims

- Introduce rigorous methods for proving algorithms correct and for analysing their running time and memory use.
- Give you a working toolkit of standard data structures and algorithms, and the judgement to choose between them.
- Develop the habit of thinking about data structures as abstract data types with separately analysed implementations.
- Introduce programming in C, and strengthen your programming by implementing, testing and measuring the data structures you study, with explicit control over memory.

3. Learning Outcomes

After completing this course, you will be able to:

LO1. Analyse the running time and memory use of iterative and recursive algorithms using asymptotic notation, and solve recurrences by substitution, recursion trees and the master theorem.

LO2. Prove the correctness of algorithms using loop invariants and induction.

LO3. Implement and use the standard abstract data types (lists, stacks, queues, priority queues, sets and maps) in C with array-based and linked representations, managing memory explicitly, and analyse dynamic arrays with amortised analysis.

LO4. Explain, implement and compare comparison-based and linear-time sorting algorithms, and explain the $\Omega(n \log n)$ lower bound for comparison sorting.

LO5. Design, implement and analyse hash tables and balanced search trees, and choose an appropriate data structure for a given problem.

4. Syllabus

Unit	Topic
1	Introduction: algorithms, problems and the RAM model
2	C for Java programmers I: types, control flow, functions, pointers and arrays
3	C for Java programmers II: strings, structs, dynamic memory and the toolchain
4	Asymptotic notation and the growth of functions
5	Correctness: loop invariants and proofs by induction
6	Recursion and recurrences: substitution, recursion trees, master theorem
7	Arrays, dynamic arrays and amortised analysis
8	Linked lists, stacks and queues
9	Abstract data types in C: header files, opaque types and function pointers
10	Elementary sorting; merge sort
11	Quicksort and randomised analysis
12	Heaps, heapsort and priority queues
13	Sorting lower bound; counting sort, radix sort and bucket sort
14	Searching: binary search; selection and order statistics
15	Hashing: hash functions, chaining and open addressing
16	Binary search trees
17	Balanced search trees: AVL and red-black trees
18	Course Review & Exam

5. Teaching and Learning Methods

The module is self-paced. All materials are provided on Moodle. Each unit has a lecture, exercises on analysis and proofs, and a programming task in C in which you implement and test the data structure or algorithm of the unit.

Activity	Hours
Lectures	34
Labs / Tutorials	40
Independent study	126
Total	**200**

6. Assessment

Your knowledge will be assessed throughout the course with quizzes, programming assignments and the final exam. The quizzes and the exam check whether you can analyse algorithms, prove them correct and select suitable data structures. The programming assignments check whether you can implement data structures and algorithms correctly and efficiently.

Component	Weight	Due	LOs
Quizzes	20%	Throughout	LO1–LO5
Programming assignments	40%	Throughout	LO3–LO5
Final exam	40%	End of module	LO1–LO5
Total	**100%**		

7. Reading List

Essential

- T. H. Cormen, C. E. Leiserson, R. L. Rivest, C. Stein: *Introduction to Algorithms*, 4th ed., MIT Press, 2022.

Recommended

- R. Sedgewick: *Algorithms in C, Parts 1–4: Fundamentals, Data Structures, Sorting, Searching*, 3rd ed., Addison-Wesley, 1998.
- B. W. Kernighan, D. M. Ritchie: *The C Programming Language*, 2nd ed., Prentice Hall, 1988.
- K. N. King: *C Programming: A Modern Approach*, 2nd ed., W. W. Norton, 2008.
- S. S. Skiena: *The Algorithm Design Manual*, 3rd ed., Springer, 2020.

8. Materials and Licence

Course materials are published under the Creative Commons Attribution-ShareAlike 4.0 licence unless stated otherwise. See wosm.academy/licence for details.

9. Policies

- **Attendance:** Attendance is not mandatory. The module is self-paced, so you can study the materials when it suits you.
- **Late submissions:** Work submitted after the deadline is penalised.
- **Academic integrity:** All submitted work must be your own. See the Code of Conduct.