Freie Universität Bozen
Libera Università di Bolzano
Università Liedia de Bulsan

Syllabus

Course Description

Course Title	Data Structures and Algorithms
Course Code	76243
Course Title Additional	
Scientific-Disciplinary Sector	INFO-01/A
Language	English
Degree Course	Bachelor in Computer Science
Other Degree Courses (Loaned)	
Lecturers	Prof. Werner Nutt, Werner.Nutt@unibz.it https://www.unibz.it/en/faculties/engineering/academic-staff/person/7380 dr. Tiziano Dalmonte, Tiziano.Dalmonte@unibz.it https://www.unibz.it/en/faculties/engineering/academic-staff/person/47069
Teaching Assistant	Dott. Anton Gnatenko Dott. Pavel Merinov
Semester	First semester
Course Year/s	2
CP	9
Teaching Hours	60
Lab Hours	30
Individual Study Hours	135
Planned Office Hours	18
Contents Summary	By the end of the course, students will be able to formulate algorithmic problems and identify such problems within applications. They will gain a solid understanding of standard data structures and the algorithmic techniques used to solve related problems. Students will also understand the interplay between algorithmic approaches and the choice of suitable data structures.

	In addition, they will learn how to verify the correctness of algorithms and analyse their time and space complexity. Finally, students will be able to compare alternative algorithms in terms of their suitability for a given application.
Course Topics	- Design Principles: Problem reduction via recursion - Searching and Sorting - Correctness: Loop invariants, termination - Complexity: Asymptotic analysis - Divide and Conquer - Pointers, dynamic data structures, linked lists - Abstract data types: stacks, queues, priority queues, maps - Binary trees, red-black trees - Graph algorithms
Keywords	Algorithms, Data Structures, Algorithmic Complexity, Correctness of Algorithms, Design of Algorithms
Recommended Prerequisites	The course requires introductory Java programming skills and basic knowledge of sets, functions, and calculus.
Propaedeutic Courses	
Teaching Format	The course combines lectures, lab group sessions led by teaching instructors, and biweekly coursework assignments. In the lectures, the instructor introduces new concepts and techniques through board presentations and short in-class exercises. The assignments give students the opportunity to consolidate these concepts by applying them to selected problems. Students also analyse the performance of their implementations and compare the results with theoretical predictions. In the lab groups, students discuss possible approaches to the assignment tasks with the instructors and compare alternative solutions. They also work on additional problems, independent of the assignments, to deepen their understanding of the material covered in the lectures.
Mandatory Attendance	Attendance is not compulsory but strongly recommended. Lectures include board presentations, exercises, and discussions designed to foster algorithmic thinking—a skill best developed through practice. All materials and assignments are available on the OLE page; however, slides and notes alone are not sufficient for mastering the course. Regular attendance, active participation in exercises,

	and timely submission of coursework are key to success.
Specific Educational Objectives and Learning Outcomes	Knowledge and Understanding - D1.3 Have a solid knowledge of the most important data structures and programming techniques. - D1.5 Know the concepts of complexity of algorithms and data structures - D1.6 Have a solid knowledge of the most important algorithms for searching and manipulating common data structures and of their complexity Applying knowledge and understanding - D2.3 Be able to solve problems using programming methodologies. - D2.6 Be able to analyze and measure size, complexity and critical aspects of algorithms and data structures Ability to make judgments - D3.1 Be able to collect and interpret useful data and to judge information systems and their applicability. - D3.2 Be able to work autonomously according to the own level of knowledge and understanding. - D3.6 Be able to collect useful data about the performance of algorithms and to judge which algorithm is most suitable for a given task Communication skills - D4.1 Be able to use one of the three languages English, Italian and German, and be able to use technical terms and communication appropriately. - D4.4 Be able to structure and write technical documentation. Learning skills - D5.1 Have developed learning capabilities to pursue further studies with a high degree of autonomy. - D5.3 Be able to follow the fast technological evolution and to learn cutting edge IT technologies and innovative aspects of last generation information systems.
Specific Educational Objectives and Learning	Algorithm design as a primary goal: Students will not only become familiar with existing algorithms but above all learn systematic

Outcomes (additional info.)	methods for designing new algorithms to solve previously unseen problems. Heuristic design principles: Students will learn to design algorithms incrementally by building and extending partial solutions while scanning input data, making use of auxiliary information about the already processed part of the input. Loop invariants as a design tool: Students will learn to formulate and use loop invariants systematically, both as a guiding principle for algorithm design and as a means of ensuring correctness.
Assessment	The assessment is based on a written final exam, a mock exam, and coursework assignments. 1. The written exam consists of several questions. Each question describes an algorithmic problem and provides examples of input and expected output. Students are expected to outline solution ideas, present an algorithm in pseudocode, and analyze its running time. The questions vary in difficulty. Passing the written exam is mandatory. 2. The mock exam has the same structure as the written exam but is shorter and limited to the content of the first half of the course. It allows students to familiarize themselves with the exam format and expectations. 3. The assignments are more extensive and challenging. They typically require both a solution idea and a Java implementation. The code is submitted via the Codebase platform, where it is automatically tested against unit tests. Assignments provide practice in applying course concepts to more demanding problems. Marks are valid for the three exam sessions following the teaching of the course. The coursework and the mock exam are optional. Weighting and calculation of the final mark are specified in Evaluation criteria.
Evaluation Criteria	The final grade is based on coursework assignments (45%), a mock exam (5%), and the written final exam (50%).

	To pass the course, students must pass the written exam. In this exam, they are required to apply techniques taught in the course to given settings and to design algorithms for new problems. The algorithms must be analyzed with respect to correctness and efficiency. Answers are marked according to their correctness, the suitability of the proposed algorithms, and the validity and clarity of the analysis. The mock exam follows the same evaluation criteria as the written exam. In the coursework assignments, students develop solutions to algorithmic problems and analyze them in terms of correctness and running time. These exercises are assessed on correctness, efficiency, and the validity of the analysis. In addition, experiments require students to implement variants of algorithms and to determine under which conditions each variant performs best. The experiments are assessed on the appropriateness of the experimental design, the quality of the measurements, and the validity of the conclusions drawn.
Required Readings	Thomas H. Cormen. Introduction to Algorithms. The MIT Press, Cambridge, Massachusetts London, England, 3rd edition, January 2009. ISBN 978-0-262-53305-8.
Supplementary Readings	Kurt Mehlhorn and Peter Sanders. Algorithms and Data Structures: The Basic Toolbox. Springer, Berlin Heidelberg, 2008 edition, November 2010. ISBN 978-3-642-09682-2.
Further Information	Java (https://openjdk.org)
Sustainable Development Goals (SDGs)	Quality education