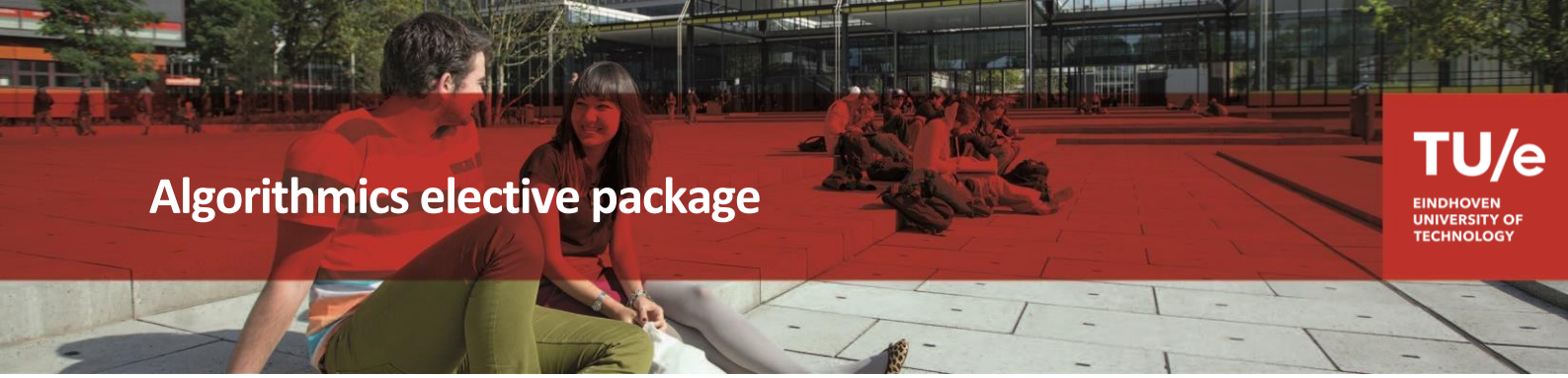


Algorithmics elective package

Algorithmics	
Offered by	Department of Mathematics and Computer Science
Language	English
Primarily interesting for	All IAM students, in particular those interested in algorithms and discrete mathematics.
Prerequisites	Applied Mathematics (note: this package is not for BCS students) Students are assumed to have knowledge of logic reasoning (as offered in the courses 2WF40 Set Theory and Algebra or 2IT60 Logic and Set Theory) and of programming (e.g. acquired from the courses 2WH20 Programmeren en Modelleren or 2IP90 Programming)
Contact person	Prof. dr. M. de Berg (mdberg@win.tue.nl)

Content and composition

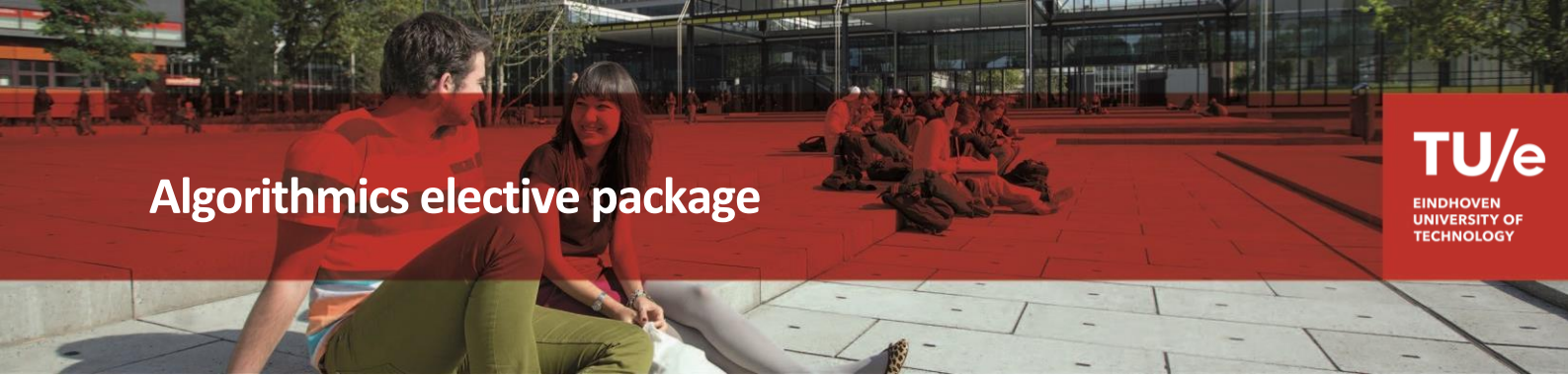
This package deals with automata and algorithmic complexity (which are fundamental concepts in the theory of computing).

Course code	Course name	Level classification
2IL50	Data Structures	2.
2IT90	Automata, Language Theory and complexity	2.
2ILC0	Algorithms	3.

Course description

Data Structures (2IL50)
There are many aspects to the study of data structures, and the algorithms that operate upon them. In this course, the student will learn the basic skills and knowledge to develop efficient algorithms to solve computational problems and to make informed choices between different solutions for the same problem. These include standard data structures and algorithms for frequently appearing problems, algorithm design techniques, how to establish that an algorithm is correct, and how to analyse the efficiency of an algorithm.

Automata, Language Theory and complexity (2IT90)
The Turing machine, named after the Computer Science pioneer Alan Turing, is an abstract representation of the traditional Von Neumann computer: finite control, random-access memory, and input/output. The push-down automaton replaces the random-access memory with a stack. The finite automaton has no memory at all, only finite control. In this course we study the languages, i.e. the sets of strings, which correspond to these three types of automata and identify the classes of decidable, context-free, and regular languages, respectively. For regular and context-free languages we propose concise ways to describe them. For the decidable languages we seek to draw the borderline of languages that can be recognized by a Turing machine in reasonable computation time and those languages for which this is infeasible. This leads to the famous conjecture P ≠NP.



Algorithmics elective package

TU/e
EINDHOVEN
UNIVERSITY OF
TECHNOLOGY

Algorithms (2ILC0)
Building on the course 2IL50 Data structures we take the design and analysis of algorithms one step further. We study general solutions to optimization problems: backtracking, Course code Course name Scheduled (Quarter/Slot) Level 2IL50 Data Structures Quarter 3/Slot B 2 2IT90 Automata, Language Theory and complexity Quarter 1/Slot B 2 2ILC0 Algorithms Quarter 1/Slot E 3April 2020 dynamic programming, and greedy algorithms. Backtracking can be applied to more or less all problems but it generally yields very slow algorithms, a greedy approach can be applied to few problems but it generally gives very fast algorithms, and dynamic programming lies in between. In particular we deal with algorithms for optimization problems on graphs: computing shortest paths, computing maximum flows, and searching for matchings. We also study the limitations of traditional algorithm design and analysis. We consider problems for which it seems impossible to design efficient algorithms, introduce the concept of NP hardness (which formalizes this) and consider approximation algorithms. Finally, we discuss the use of linear programming as a tool to model and solve optimization problems.