Computer Science for Data Science

Offered by: Department of Mathematics and Computer Science
Language: English
Primarily interesting for: Major Data Science
Prerequisites: Programming (JBI010 or 2IP90), Foundations of computing (JBI020)
Contact person: dr. Marloes van Lierop (M.v.Lierop@tue.nl)

What is this package about?
This package provides courses to students in the Data Science major, who wish to advance their computer science knowledge. The logic and set theory course is about propositional logic and predicate logic, and you learn to reason with logical formulas. The data structures course extends the basic concepts provided in the foundations of computing course. Finally, the algorithmic aspects of data analysis course focuses on the application of algorithmic techniques in data analysis problems.

<table>
<thead>
<tr>
<th>Course name</th>
<th>Scheduled</th>
<th>Level</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. 2IT60 Logic and set theory</td>
<td>Q1</td>
<td>1</td>
</tr>
<tr>
<td>2. 2IL50 Data structures</td>
<td>Q3</td>
<td>2</td>
</tr>
<tr>
<td>3. JBI040 Algorithmic aspects of data analysis (or 2ILC0 Algorithms*)</td>
<td>Q4 (or Q1)</td>
<td>3</td>
</tr>
</tbody>
</table>

* JBI040 may be replaced by 2ILC0 if that fits an individual study program better

In Osiris the level of JBI040 is wrongly presented as level 2 instead of level 3.

Course descriptions

Logic and set theory
Logical reasoning is an indispensable tool when designing a solution to any complex technical problem. This course discusses the principles of correct logical reasoning. You learn to formulate statements in unambiguous logical language, and to manipulate statements in a structured and logically valid manner. At the end of the course you are able to give simple mathematical proofs, in particular using the technique of mathematical induction.

Data structures
For solving algorithmic problems many aspects need to be mastered: efficient ways of storing and manipulating (large amounts of) data, algorithm design techniques, how to establish that an algorithm is correct, and how to analyze the efficiency of an algorithm. Design techniques: Incremental algorithms, recursion, divide & conquer. Correctness: induction and invariants. Efficiency analysis: O-notation, recurrences. Sorting: MergeSort, InsertionSort, HeapSort, sorting in linear time, lower bounds for sorting. Selection Algorithms. Data structures: abstract data structures, heaps, hashing, search trees (incl. red-black trees), augmenting data structures, union-find. Basic graph algorithms: adjacency list, adjacency matrix, DFS, BFS, topological sort, minimum spanning trees.
Algorithmic aspects of data analysis
This course focuses on the application of algorithmic techniques in data analysis problems. We will consider theoretical aspects—one goal is to show students how data analysis problems can be formulated in precise mathematical terms as optimization or decision problems—but the students will also have to implement various algorithms. We will use an important data-analysis task, namely clustering, as a vehicle for this. Clustering is the task of partitioning a set of input elements into groups by their similarity. The course will give an overview of different approaches to this problem, starting with evaluation metrics—how can we define what a good clustering looks like—and with a focus on algorithms for computing a clustering. We explore how certain data structures and algorithmic techniques can be used to gain considerable speedups in certain situations. We will discuss how the curse of dimensionality affects the clustering problem, and discuss dimension-reduction techniques that can be used to mitigate these effects.