



Universitat de Lleida

DEGREE CURRICULUM
**ALGORITHMS AND
COMPLEXITY**

Coordination: PLANES CID, JORDI

Academic year 2020-21

Subject's general information

Subject name	ALGORITHMS AND COMPLEXITY		
Code	102061		
Semester	2nd Q(SEMESTER) CONTINUED EVALUATION		
Typology	Degree	Course	Character
	Double bachelor's degree: Degree in Computer Engineering and Degree in Business Administration and Management	2	COMPULSORY
	Bachelor's Degree in Computer Engineering	2	COMPULSORY
Modality	Attendance-based		
Course number of credits (ECTS)	4.5		
Type of activity, credits, and groups	Activity type	PRALAB	TEORIA
	Number of credits	3	1.5
	Number of groups	3	1
Coordination	PLANES CID, JORDI		
Department	COMPUTER SCIENCE AND INDUSTRIAL ENGINEERING		
Important information on data processing	Consult this link for more information.		
Language	Catalan		

Teaching staff	E-mail addresses	Credits taught by teacher	Office and hour of attention
LAMAS VARELA, ALBA	alba.lamas@udl.cat	3	
PLANES CID, JORDI	jordi.planes@udl.cat	6	
RIBALTA GENÉ, MARC	marc.ribalta@udl.cat	1,5	

Subject's extra information

Suggestions

For any questions it is recommended to send an email to teachers of the course.

We advise you to solve the proposed problems and programming exercises, since it allows to reach the learning objectives.

The following courses are highly recommended: Data Structures and Discrete Mathematics.

Learning objectives

- Characterize formally problems. Analyze the efficiency of algorithms using asymptotic notation.
- Identify the nature of the problem and identify the appropriate strategy algorithms.
- Design and implement data up appropriate structures to represent information of each problem.
- Design and implement strategies for efficient algorithms to solve different types of problems.

Competences

Degree Competences

GII-FB3. Capacity to understand and master the basic concepts of discreet mathematics, logical, algorithmic and computational complexity, and its application to solve engineering problems.

GII-CRI6. Knowledge and application of the basic algorithmic procedures of the computer technologies to design problem solving, analysing the suitability and complexity of the algorithms proposed.

GII-CRI7. Knowledge, design and efficient use of the types and data structure more suitable for solving a problem.

GII-CRI8. Capacity to analyse, design, build and keep safety and efficiency in applications, choosing the paradigm and the most suitable programming languages.

Cross-disciplinary Competences

EPS1. Capacity to solve problems and prepare and defence arguments inside the area of studies.

EPS5. Capacity of abstraction and of critical, logical and mathematical thinking.

Subject contents

Organization of the course topics:

1. Preliminaries: algorithm, notation.
2. Efficiency of algorithms. Asymptotic notation . Analysis of algorithms.
3. Techniques for transformation of recursive algorithms.
4. Algorithmic schemes: divide and conquer.
5. Algorithmic schemes: dynamic programming.
6. Algorithmic schemes: backtracking.

Methodology

The course contents are structured in four learning units. The first is to characterize the formal study of algorithms. In this sense, we will study the formal specification of algorithms based on preconditions and postconditions and analyze the efficiency of algorithms using asymptotic measures to study the run-time of algorithms. The second teaching unit aims to study formal verification techniques for iterative and recursive algorithms and processing techniques for the study of recursive algorithms. The third learning unit is to study algorithmic schemes, i.e., analyze, design and implement algorithms to solve not only a specific problem but a family of problems that share a common set of characteristics. We study three algorithmic schemes: divide and conquer, backtracking and greedy search. The systematic design and analysis of algorithms based on a specific scheme is focused on the study and development of solutions or strategies to solve a problem. A different approach consists in considering globally all algorithms or strategies that may solve a particular problem. This includes all possible algorithms or strategies that have not yet been defined. This approach is considered to be in the field of computational complexity which will be briefly introduced in the last teaching unit. The study of each technique and algorithmic scheme will be tackled, based on solving specific problems for each type. Furthermore, the algorithmic solutions developed throughout the course will be implemented in imperative and functional programming languages. From the standpoint of the implementation of the algorithms, empirical study of their run-time for different instances of the problems treated will be also considered. The empirical study of the run-time of the algorithm implementations will allow to finely compare the different algorithmic strategies developed during the course, as sometimes the asymptotic run-time behaviour may hinder important variations of the run time of different algorithms.

Development plan

The course is organized in large group classes and laboratory classes. Each week students large group race in 2 hours and 2 hours lab group.

A large group classes are the algorithmic schemes and theoretical basis of the subject. For each technique and formal algorithmic scheme proposes a collection of problems which students must solve. Solving the revised large group classes and laboratory.

In the laboratory classes are taught the most important features of python. In addition to discussing implementation problems and collections solution develops the three practical compulsory works.

The first mandatory practice begin during the 3rd week of the course and will be given to the date fixed for the 1st

written test (1st part).

Mandatory practice will begin during the second week of the 10th year and will be delivered to the date set for the 2nd written test (2nd part).

Week	Description	Classroom Activity Big Group	Classroom/independent work
1	Lecture and problems	Lesson 1	3.5h/6h
2	Lecture and problems	Lesson 2	3.5h/6h
3	Lecture and problems	Lesson 3	3.5h/6h
4	Lecture and problems	Lesson 4	3.5h/6h
5	Lecture and problems	Lesson 5	3.5h/6h
6	Lecture and problems	Lesson 6	3.5h/6h
7	Lecture and problems	Lesson 6	3.5h/6h
8	Lecture and problems	Review	3.5h/6h
9	Written tests	First mid-term exam	2h/3h
10	Lecture and problems	Lesson 7	3.5h/6h
11	Lecture and problems	Lesson 7	3.5h/6h
12	Lecture and problems	Lesson 8	3.5h/6h
13	Lecture and problems	Lesson 9	3.5h/6h
14	Lecture and problems	Lesson 10	3.5h/6h
15	Lecture and problems	Review	3.5h/6h
16	Written tests	Second mid-term exam	2h/3h
17	Written tests	Second mid-term exam	
18		Study week	
19	Written tests	Recovery exam	

Evaluation

Acronym	Evaluation Activities	Weighting	Minimum mark	Group	Mandatory	Recuperable
T	Activity Test	20%	-	No	No	No
E1	Exam 1	20%	-	No	No	No
P	Assignment	40%	-	Yes	Yes	Yes
E2	Exam 2	20%	-	No	No	No

The weighting may change in case the exams are not face-to-face.

The evaluation consists of activity test, two exams and one assignment organized as follows:

Activity Test: The knowledge learned in the laboratory will be evaluated before the first partial, in the same place and date of the exam 1.

Exam 1: Formalization. Costs. Recursive and iterative design. Schemes transformation of recursive algorithms. Divide and conquer.

Exam 2: Greedy schemes, backtracking using heuristic optimization.

Assignment : Costs. Recursive and iterative design. Schemes transformation of recursive algorithms. Divide and conquer scheme. Greedy schemes, backtracking using heuristic optimization.

The assignment will only delivery date, will not be handed out this term.

Delivery: Before the date fixed for the second written test.

Validation Assignment 2: In order to define the final mark, will be a written validation date fixed.

Bibliography

Basic References:

- G. Brassard y P. Bratley. Fundamentos de algoritmia. Prentice Hall. 1997.
- Cormen, T.H.; Leiserson, C.E. ; Rivest, R.L.; Stein, C. Introduction to Algorithms, (3ª edición). MIT Press, 2002. * Skiena, S. The Algorithm Design Manual. Springer 2008.

Exercices:

- Baynat B., Chrétienne P. Hanen C., Kedad-Sidhoum S., Munier-Kordon A., Picouleau C. Exercices et problèmes d'algorithmique. Ed. Dunod. 3r. ed. 2010.
- R. Guerequeta y A. Vallecillo. Técnicas de diseño de algoritmos. Servicio de Publicaciones de la Universidad de Málaga. 2nd Ed. 2000. <http://www.lcc.uma.es/~av/Libro/indice.html>
- Gonzalo, J.; Rodríguez, M. Esquemas algorítmicos: enfoque metodológico y problemas resueltos, UNED, 1997.

Implementation:

- R. Sedgewick. Algoritmos en C++. Addison-Wesley / Diaz de Santos.1995.