DEGREE CURRICULUM

ALGORITHMS AND COMPLEXITY

Coordination: PLANES CID, JORDI

Academic year 2017-18

ALGORITHMS AND COMPLEXITY 2017-18

Subject's general information

Subject name	ALGORITHMS AND COMPLEXITY
Code	102011
Semester	2nd Q(SEMESTER) CONTINUED EVALUATION
Typology	

Degree	Course	Typology	Modality
Bachelor's Degree in Computer Engineering	2	COMPULSORY	Attendance-based

ECTS credits	6
Groups	1GG,3GM
Theoretical credits	3
Practical credits	3
Coordination	PLANES CID, JORDI
Department	INFORMATICA I ENGINYERIA INDUSTRIAL
Important information on data processing	Consult this link for more information.
Language	Catalan
Distribution of credits	60 hours of lecturing, 90 hours of home work
Office and hour of attention	For questions or related issues, it is recommended to email the teachers of the course.

ALGORITHMS AND COMPLEXITY 2017-18

Teaching staff	E-mail addresses	Credits taught by teacher	Office and hour of attention
PLANES CID, JORDI	jplanes@diei.udl.cat	10,5	

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, predicate logic, proof techniques.

2. Formal specification of algorithms based on pre-post conditions.

3. Efficiency of algorithms. Asymptotic notation . Analysis of algorithms.

4. Formal verification of iterative and recursive algorithms.

5. Techniques for transformation of recursive algorithms.

6. Algorithmic schemes: divide and conquer.

7. Algorithmic schemes: greedy search.

8. Algorithmic schemes: dynamic programming.

9. Algorithmic schemes: backtracking.

10. Introduction to computational complexity.

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 Ocaml and C++. 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 tought 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	4h/6h
2	Lecture and problems	Lesson 2	4h/6h
3	Lecture and problems	Lesson 3	4h/6h
4	Lecture and problems	Lesson 4	4h/6h
5	Lecture and problems	Lesson 5	4h/6h
6	Lecture and problems	Lesson 6	4h/6h
7	Lecture and problems	Lesson 6	4h/6h
8	Lecture and problems	Review	4h/6h
9	Written tests	**First mid-term exam**	2h/3h
10	Lecture and problems	Lesson 7	4h/6h
11	Lecture and problems	Lesson 7	4h/6h
12	Lecture and problems	Lesson 8	4h/6h
13	Lecture and problems	Lesson 9	4h/6h
14	Lecture and problems	Lesson 10	4h/6h
15	Lecture and problems	Review	4h/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

Acornym	Evaluation activities	Weighting	Minimum score	Group work	Compulsory	Recoverable
P1	Practice 1	10%	3	Yes	Yes	No
T1	1st mid-term Exam	10%	3	No	Yes	Yes
A1	Activities 1,2,3,4	20%	-	No	Yes	No
P2	Practice 2	10%	3	Yes	Yes	No
T2	2on mid-term Exam	10%	3	No	Yes	Yes
A2	Activities 5,6,7	20%	-	No	No	No

E	Extra work	20%	3	No	Yes	Yes

The evaluation consists of two and three practical examinations organized as follows:

Written test 1: Formalization. Costs. Recursive and iterative design. Schemes transformation of recursive algorithms. It divides and conquers.
(no minimum requested)

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

The practice will only delivery date, will not be handed out this term, and can not be resit.
(no minimum requested)

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

Validation practice 1 and 2: In order to define the final practice, will be a written validation date set for the written test first.

Written test 2: Greedy schemes, backtracking using heuristic optimization.
(no minimum requested)

Mandatory Practice 2: Greedy schemes, backtracking using heuristic optimization.

The practice will only delivery date, will not be handed out this term, and can not be resit.
(no minimum requested)

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

Validation practice 2: In order to define the final practice, will be a written validation date fixed for the second written test.

At the end of the year if the final <5 students will be introduced to improve the grade in the written tests.

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:

- R. Guerequeta y A. Vallecillo. Tecnicas 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.