Universitat de Lleida

DEGREE CURRICULUM

# ALGORITHMS AND COMPLEXITY

Academic year 2014-15

# ALGORITHMS AND COMPLEXITY 2014-15

## Subject's general information

| | |
|---|---|
| **Subject name** | ALGORITHMS AND COMPLEXITY |
| **Code** | 102011 |
| **Semester** | 2n Semester |
| **Typology** | Compulsory |
| **ECTS credits** | 6 |
| **Groups** | 3+1 |
| **Theoretical credits** | 3 |
| **Practical credits** | 3 |
| **Department** | Informàtica i Enginyeria Industrial |
| **Modality** | Presencial |
| **Important information on data processing** | Consult this link for more information. |
| **Language** | Catalan |
| **Degree** | Degree in Computer Enginneering |
| **Distribution of credits** | Jordi Planes Cid 12<br>Maria Teresa Alsinet Bernadó 3.6 |
| **E-mail addresses** | jplanes@diei.udl.cat<br>tracy@diei.udl.cat |

# ALGORITHMS AND COMPLEXITY 2014-15

Jordi Planes Cid
Maria Teresa Alsinet Bernadó

## Subject's extra information

**Suggestions**

For questions or related issues, it is recommended to email the teachers of the course. We advise you to solve the proposed problems and programming exercises. It allows to reach the learning objectives.

Course taught in the 2nd semester of 2nd year of Degree in Computer Engineering. It corresponds to the Subject "Information" within the module "Basic Training".

## Learning objectives

Characterized formally problems. To analyze the efficiency of algorithms using asymptotic notation for the study of the cost temporary or runtime algorithms. To analyze the efficiency of algorithms using asymptotic notation for the study of spatial algorithms cost. Using the techniques of formal verification algorithms applied to recursive algorithms and iterative. Using the techniques of transformation of recursive algorithms. Using the techniques of optimization algorithms.

Identify the nature of the problem and identify the appropriate strategy algorithms. Design and implement appropriate strategies algorithms to solve different types of problems. Design and implement solutions using algorithms technique divides and conquers. Designing and implementing solutions using the technique voracious algorithms. Design and implement solutions using algorithms technique setback. Optimize algorithms based on technical solutions retreat through the design and implementation of heuristic pruning the search space. Design and implement solutions algorithms using dynamic programming technique. To analyze the spatial and temporal complexity of algorithms strategies adopted.

Design and implement data structures to represent up appropriate information of each problem. Design and implement efficiently the operations associated with data structures identified. Efficiently integrate data structures, algorithms and strategies necessary to solve complex problems. Optimize the efficiency of the solutions designed.

Design and implement strategies for efficient algorithms to solve different types of problems. Using functionalities of programming languages ??for implementing solutions. Use a software development environment based on a high level programming language. Develop efficient implementations.

## 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.

**University Competences**

**CT5.** Acquire knowledge in scientific 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: backtracking.

9. Improving the backtracking scheme by using heuristics.

10. Algorithmic schemes: dynamic programming.

11. 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 Ocaml and C++. 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).

## Evaluation

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.
Percentage 20% (no minimum requested)

**Mandatory Practice 1:** Formalization. Costs.
The practice will only delivery date, will not be handed out this term and can not be recovered.

Percentage 20% (no minimum requested)

Performing in groups of two.

Delivery: In week four.

**Mandatory Practice 2:** 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 recovered.
Percentage 20% (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.
Percentage 20% (no minimum requested)

**Mandatory Practice 3:** Greedy schemes, backtracking using heuristic optimization.

The practice will only delivery date, will not be handed out this term, and can not be recovered.
Percentage 20% (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.
* M. Goodrich y R. Tamassia. Algorithm Design. John Wiley & Sons. 2011.

**Especification and verification:**

* R. Peña Marí. Diseño de programas. Formalismo y abstracción 3 ed. Prentice Hall 2003
* J.L. Balcázar. Programación Metódica. McGraw-Hill, 1993.

**Dada structures and algorithms:**

* J. Kleinberg and E. Tardos, Algorithm Design, Addison Wesley 2006.
* Cormen, T.H.; Leiserson, C.E. ; Rivest, R.L.; Stein, C. Introduction to Algorithms, (3ª edición). MIT Press, 2002.
* M. Weiss. Estructuras de datos y algoritmos. Addison-Wesley Iberoamericana.1995.
* R. Kruse. Estructuras de Datos y Diseño de programas. .Prentice Hall. 1998.

* Skiena, S. The Algorithm Design Manual. Springer 2008.
* PROGRAMACION Y ESTRUCTURAS DE DATOS AVANZADAS, Lourdes Araujo Serna, Raquel Marti?nez Unanue, Miguel Rodri?guez ArtachoEd Ramo?n Areces, 2011.
* ESQUEMAS ALGORITMICOS. ENFOQUE METODOLO?GICO YPROBLEMAS RESUELTOS (1a)**.** Rodriguez Artacho, Miguel ; Gonzalo Arroyo, Julio ; UNED
* Tecnicas de Diseno de Algoritmos.GUEREQUETA, R. y VALLECILLO, A.:Ed. Universidad de Ma?laga (1998).

**Computational complexity:**

* J.E. Hopcroft, R. Motwani, J. D. Ullman. Teoría de autómatas,lenguajes y computación. Pearson. Addison Wesley. Tercera Edición, 2007

**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.
* Martí, N.; Ortega, Y.; Verdejo, J. A. Estructuras de datos y métodos algorítmicos. Prentice Práctica, 2003.
* ESTRUCTURAS DE DATOS Y METODOS ALGORITMICOS:EJERCICIOS RESUELTOS (1a) Marti Oliet, Narciso ; Ortega Mallen, Yolanda ; VerdejoLopez, Jose Alberto ; PEARSON ALHAMBRA

**Implementation:**

* Stroustrup, B. The C++ Programming Language. Addison-Wesley. 3rd edition. 1997.

* Meyers, S. Effective C++. 3rd edition. 2005.

* Meyers, S. More Effective C++. 1995.

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

* M. Weiss. Estructuras de datos en JAVA. Addison Wesley/ Pearson Education. Madrid 2000.