Degree competences to which the subject contributes

Specific:
1. CE-2. Solve problems in Mathematics, through basic calculation skills, taking in account tools availability and the constraints of time and resources.
2. CE-3. Have the knowledge of specific programming languages and software.
3. CE-4. Have the ability to use computational tools as an aid to mathematical processes.

Generical:
4. CB-1. Demonstrate knowledge and understanding in Mathematics that is founded upon and extends that typically associated with Bachelor's level, and that provides a basis for originality in developing and applying ideas, often within a research context.
5. CB-2. Know how to apply their mathematical knowledge and understanding, and problem solving abilities in new or unfamiliar environments within broader or multidisciplinary contexts related to Mathematics.
6. CB-3. Have the ability to integrate knowledge and handle complexity, and formulate judgements with incomplete or limited information, but that include reflecting on social and ethical responsibilities linked to the application of their knowledge and judgements.
7. CG-1. Show knowledge and proficiency in the use of mathematical language.
8. CG-2. Construct rigorous proofs of some classical theorems in a variety of fields of Mathematics.
9. CG-3. Have the ability to define new mathematical objects in terms of others already know and ability to use these objects in different contexts.
10. CG-4. Translate into mathematical terms problems stated in non-mathematical language, and take advantage of this translation to solve them.
11. CG-6 Detect deficiencies in their own knowledge and pass them through critical reflection and choice of the best action to extend this knowledge.

Transversal:
11. SELF-DIRECTED LEARNING. Detecting gaps in one's knowledge and overcoming them through critical self-appraisal. Choosing the best path for broadening one's knowledge.
200152 - PM - Mathematical Programming

Teaching methodology

(Section not available)

Learning objectives of the subject

(Section not available)

Study load

| Total learning time: 187h 30m | Hours large group: 45h 24.00% | Hours medium group: 0h 0.00% | Hours small group: 30h 16.00% | Guided activities: 0h 0.00% | Self study: 112h 30m 60.00% |
## Content

| **Introduction** | **Learning time:** 23h 30m  
Theory classes: 4h 30m  
Practical classes: 3h  
Self study: 16h |
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Description:</strong></td>
<td>The Mathematical Programming. Building methodology of Mathematical Programming models. The paper of the models in the decision making process. Main types of Mathematical Programming: linears, integers, network flows, nonlinear, stochastics, etc.</td>
</tr>
</tbody>
</table>

| **Linear Programming** | **Learning time:** 47h 30m  
Theory classes: 13h 30m  
Practical classes: 6h  
Laboratory classes: 3h  
Self study: 25h |
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Description:</strong></td>
<td>Definition and examples of linear programming problems. The geometry of linear programming: feasible sets, convex sets and polyhedrons; optimal solutions, extreme points and basic solutions. The primal simplex algorithm: development, convergence and computational complexity. Duality theory: definition of the dual problem and examples, duality theorems. Duality and the max flow - min cut theorem. Dual simplex algorithm: development and convergence. Sensitivity analysis.</td>
</tr>
</tbody>
</table>

| **Integer Linear Programming** | **Learning time:** 18h 30m  
Theory classes: 6h  
Practical classes: 4h  
Self study: 8h 30m |
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Description:</strong></td>
<td>Definition of linear integer programming problem and examples. Linear relaxation. Valid, strong and ideal formulations. Algorithms for linear integer programming: branch and bound, Gomory's cutting planes, branch and cut.</td>
</tr>
</tbody>
</table>
There will be a non eliminatory midterm exam (ExP), a final exam (ExF), and a mark for practical assignments (Pr).

The final mark NF of the course will be:

\[ NF = \max\{ExF, 0.8 \cdot ExF + 0.2 \cdot Pr, 0.6 \cdot ExF + 0.2 \cdot ExP + 0.2 \cdot Pr\} \]

An extra exam will take place on July for students that failed during the regular semester.

If the student fails, the extra evaluation will only consist of a resit exam (neither Pr nor ExP/ExF will be considered).

**Qualification system**

**Unconstrained Nonlinear Programming**

Learning time: 28h 30m

Theory classes: 7h 30m
Practical classes: 5h
Self study: 16h

Description:

**Constrained Nonlinear Programming**

Learning time: 34h 30m

Theory classes: 11h 30m
Practical classes: 7h
Self study: 16h

Description:

**Bibliography**

Basic: