Universidad Zaragoza

Academic Year/course: 2022/23

29707 - Fundamentals of computing

Syllabus Information

Academic Year: 2022/23
Subject: 29707 - Fundamentals of computing
Faculty / School: 110 - Escuela de Ingeniería y Arquitectura
Degree: 434 - Bachelor's Degree in Mechanical Engineering
ECTS: 6.0
Year: 1
Semester: 434-First semester o Second semester
330-Second semester
107-Second semester
Subject Type: Basic Education
Module:

1. General information

1.1. Aims of the course

The objectives of the course are fundamentally of two types:

1. Train the student so that they can propose the solution of an information processing problem by creating simple programs. Therefore, the basic and core content of the course is programming and, in particular, the specification of problems, the approach of a range of solutions as possible alternative algorithms, the choice of the best solution based on experimentation or previous experiences, and the translation of these solutions into programs executable by a computer in a general purpose programming language. In addition, students must be able to relate the problems solved in this course with other problems that arise in matters of the degree.
2. That the student knows the constituent elements of a computer, understands its basic operation, and is capable of searching for information and applying the knowledge of programming and problem solving in the tools and software applications of interest for the degree. Among the software applications that will be studied there will be general applications such as operating systems or tools to develop programs, and others more specific to the degree such as tools for Computer Aided Design, Manufacturing and Engineering (CAD/CAM/CAE). Students are not expected to have a deep understanding of these software tools, but rather to be able to analyze the hardware and software needs to be able to use them.

These approaches and objectives are aligned with the following Sustainable Development Goals (SDGs) of the United Nations 2030 Agenda (https://www.un.org/sustainabledevelopment/), in such a way that the acquisition of the learning outcomes of the course provides training and competence to contribute to some extent to its achievement. Specifically, they are aligned with the following objectives:

- **Goal 9: Industry, innovation and infrastructures.**
  - **Target 9.1**: Develop quality, reliable, sustainable and resilient infrastructure, including regional and transborder infrastructure, to support economic development and human well-being, with a focus on affordable and equitable access for all.
- **Goal 16: Peace, justice and strong institutions**
  - **Target 16.5**: Substantially reduce corruption and bribery in all their forms.

1.2. Context and importance of this course in the degree

"Fundamentals of Computing" is a basic course taught in the first year of the degree. This particular temporary location allows students to apply the knowledge acquired in this course in most of the courses of the degree, where to a greater or lesser extent they will need to rely on computer tools for problem solving or for the automatic processing of information.

Throughout the course, the material of this course will be related to other courses of the degree in two ways:

1. Identifying the most common hardware and software needs a mechanical engineer has in carrying out his professional work, emphasizing design tools (CAD/CAM/CAE).

2. Identifying the contributions of computer programming to problem solving in the field of Mechanical Engineering.

1.3. Recommendations to take this course

This course introduces the Engineering student to problem solving using programming as a tool. The tool is introduced from the beginning, both from a general perspective of use, and in particular aspects aimed at solving specific problems. To take this course, the student is recommended to carry out **continuous work** in order to better develop problem-solving skills using a computer. In addition, it is convenient that each student who takes this course has a facility for understanding and analyzing problems and the logical deduction of solutions. An adequate mathematical training in previous studies is very convenient.

2. Learning goals

2.1. Competences

- **CE03.-** Basic knowledge of the use and programming of computers, operating systems, databases and computer programs with application in engineering.
- **CT04.-** Ability to solve problems and make decisions with initiative, creativity and critical reasoning.
- **CT06.-** Ability to use the engineering techniques, skills and tools necessary for its practice.
- **CT10.-** Ability to learn continuously and develop autonomous learning strategies.
- **CT11.-** Ability to apply information and communication technologies in Engineering.

2.2. Learning goals

To pass this course, each student must demonstrate the following results:

- **RA1**: Knows and uses with ease the tools to retrieve information from sources on digital media (including browsers, search engines, and catalogues).
- **RA2**: Knows the basic functioning of computers, operating systems and databases and performs simple programs on them.
- **RA3**: Use computer equipment effectively, taking into account its logical and physical properties.
- **RA4**: Knows and uses environments for program development.
- **RA5**: Analyze and generate solutions to low-medium complexity information processing problems in the world of engineering.

2.3. Importance of learning goals

This course supposes the first contact with the concepts and skills that constitute the "engineer's way of thinking", and that allow them to be put into practice with real problems from the beginning. If we attend to problem solving, Computer Science deals with the knowledge, design and exploitation of computation and computer technology, constituting a discipline that:

1. Develops the ability to express solutions as algorithms, and their role in approaching areas such as system design, problem solving, simulation, and modeling.
2. Requires a disciplined approach to problem solving, from which quality solutions are expected.
3. Controls the complexity of problems, first through abstraction and simplification, to then design solutions through the integration of components.
4. Facilitates the understanding of the opportunities offered by the automation of processes, and how people interact with computers.
5. Facilitates learning, through experimentation, of basic principles such as conciseness and elegance, as well as recognizing bad practices.

3. Assessment (1st and 2nd call)

3.1. Assessment tasks (description of tasks, marking system and assessment criteria)

Each student must demonstrate that they have achieved the intended learning outcomes through the following assessment activities:

The evaluation is divided in each call into two grades P1 and P2:

- **P1. Written exam (theoretical part)**, in which each student has to answer, where appropriate, conceptual questions and solve problems. Each student must take this test and obtain a minimum grade of 4.0 points in order

to pass the course. If this minimum grade is exceeded, it weights 70% in the grade for the course. Otherwise, the qualification of this written exam is the one that will appear in the minutes of the course.

- **P2. Practical part**. If the written exam is passed, this part weighs 30% of the grade for the course. In the first call, it can be passed through carrying out activities during the semester of the course (**P2A**) or through a global practice exam (**P2B**). In the second call, this practical part can only be passed through the global practice exam (**P2B**).
    1. **P2A Continuous assessment**. Activities to be carried out during the four-month period of teaching the course:
    - Delivery of results of the laboratories of the course: throughout the semester, the deadlines for delivery of the programming problems that arise in the laboratory sessions of the course will be announced. Some of these problems will be corrected and qualified (although not necessarily all the laboratories or all the works stated in them) by the teaching staff or by other students of the course through a peer evaluation system. The teaching staff will indicate if the programming problems corresponding to the laboratory sessions must be done individually or in teams. **The activities not delivered will be weighted in the calculation of P2A as if their grade were 0. Likewise, in accordance with the SDG 16.5 goal, when plagiarism or other irregular practices are detected in the deliveries of practices, all those involved (both supposed plagiarists as alleged plagiarized) will be graded 0.**
    2. **P2B Overall assessment**. Global practice exam, in which programming work must be carried out individually on a computer in a predetermined time and in whose qualification it will be evaluated that the resulting program correctly solves the problem posed and the quality of its source code. The P2B rating is an alternative to the P2A rating.

Only those who do not appear for the written exam will be considered not presented in the first call.

Those who do not appear neither for the written exam nor for the global practice exam in September will be considered not presented in the second call.

4. Methodology, learning tasks, syllabus and resources

4.1. Methodological overview

The learning methodology is based on the following:

1. The presentation of the contents of the course in lectures by the teachers.
2. The resolution of problems raised in class.
3. The personal study of the course by the students.
4. The development of laboratory work by students, guided by teachers, which develop theoretical knowledge.
5. The development of simple programs of increasing difficulty proposed by the teachers.

It should be borne in mind that the course has both a theoretical and practical orientation. For this reason, the learning process emphasizes the student's attendance at lectures, laboratory practices, simple programs of increasing difficulty, and individualized study.

4.2. Learning tasks

- In the theoretical classes the program of the course will be developed.
- In the classes of problems, problems of application of the concepts and techniques presented in the program of the course will be solved.
- Laboratory sessions in which programs will be developed in front of a computer. In the different sessions, each student must carry out programming work, individually or in a team, fine-tuning one or more programs.

4.3. Syllabus

1. Introduction. Structure and functions of a computer. Hardware of a computer. Introduction to Operating Systems, application software of interest for the degree. Programming languages, compilers and interpreters.
2. Basic data types, operators and expressions. Data input and output (display)
3. Control structures. Sequential, conditional, and iterative composition. Procedures and functions.
4. Composite data types. Data structures. Vectors and matrices. Strings.
5. Text files. Files with data separated by delimiter characters.

4.4. Course planning and calendar

The teaching organization of the planned course is as follows:

- Theoretical classes (2 hours per week)
- Problem classes (1 hour per week)
- Laboratory sessions (2 hours each week). They are programming work sessions, supervised by one or two teachers, in which students participate in small groups.

The schedules of all the classes and the dates of the laboratory sessions will be announced in advance through the websites of the center and the course.

The proposed programming work in laboratory sessions will be delivered on the dates indicated.

The exam schedule will be the one established by the School and the delivery dates for assessment work will be announced well in advance.

4.5. Bibliography and recommended resources

The recommended bibliography for this course can be consulted at this link of the Library of the University of Zaragoza.