Algorithms and data structures

Academic Year: (2023 / 2024) **Review date: 18-04-2023**

Department assigned to the subject: Computer Science and Engineering Department

Coordinating teacher: GONZALEZ CARRASCO, ISRAEL

Type: Compulsory ECTS Credits : 6.0

Year : 1 Semester : 2

REQUIREMENTS (SUBJECTS THAT ARE ASSUMED TO BE KNOWN)

- Programming (Course: 1 / Semester: 1)
- Calculus (Course: 1 / Semester: 1)

SKILLS AND LEARNING OUTCOMES

¿ Understand the concept of abstract data types
¿ Know and apply linear and non-linear abstract data types.
¿ Know how to select the most appropriate data structures to solve a problem by assessing complexity, storage needs and performance.

DESCRIPTION OF CONTENTS: PROGRAMME

1. Abstract Data Type
2. Linear Abstract Data Types: stacks, queues, linked lists.
3. Complexity of Algorithms.
4. Recursive Algorithms.
5. Trees
6. Graphs.
7. Divide and Conquer.

LEARNING ACTIVITIES AND METHODOLOGY

THEORETICAL-PRACTICAL CLASSES. [44 hours with 100% classroom instruction, 1.67 ECTS]
Knowledge and concepts students must acquire. Student receive course notes and will have basic reference texts to facilitate following the classes and carrying out follow up work. Students partake in exercises to resolve practical problems and participate in workshops and evaluation tests, all geared towards acquiring the necessary capabilities.

TUTORING SESSIONS. [4 hours of tutoring with 100% on-site attendance, 0.15 ECTS]
Individualized attendance (individual tutoring) or in-group (group tutoring) for students with a teacher.

STUDENT INDIVIDUAL WORK OR GROUP WORK [98 hours with 0 % on-site, 3.72 ECTS]

WORKSHOPS AND LABORATORY SESSIONS [8 hours with 100% on site, 0.3 ECTS]

FINAL EXAM. [4 hours with 100% on site, 0.15 ECTS]
Global assessment of knowledge, skills and capacities acquired throughout the course.

METHODOLOGIES

THEORY CLASS. Classroom presentations by the teacher with IT and audiovisual support in which the subject's main concepts are developed, while providing material and bibliography to complement student learning.

PRACTICAL CLASS. Resolution of practical cases and problem, posed by the teacher, and carried out individually or in a group.

TUTORING SESSIONS. Individualized attendance (individual tutoring sessions) or in-group (group tutoring sessions) for students with a teacher as tutor.

LABORATORY PRACTICAL SESSIONS. Applied/experimental learning/teaching in workshops and laboratories under the tutor's supervision.

ASSESSMENT SYSTEM

% end-of-term-examination:	50
% of continuous assessment (assigments, laboratory, practicals…):	50

EVALUATION SYSTEMS

SE1 - FINAL EXAM. [50 %]
Global assessment of knowledge, skills and capacities acquired throughout the course.

SE2 - CONTINUOUS EVALUATION. [50 %]
Assesses papers, projects, class presentations, debates, exercises, internships and workshops throughout the course.

The evaluation includes the following tests:

Primer exercise of continuos assessment (lessons 1, 2 y 3) (SE2): 15%.
Second exercise of continuos assessment (lessons 3, 4 y 5) (SE2): 15%.
Lab test: (SE2): Two tests. Total 20%.
Final Exam (all the lessons) (SE1): 50%. The maximum mark is 5 points.
Minimum mark: 2 over 5.

The final mark for the course is obtained by summing all marks of the evaluation continuous system. To pass the course, it is necessary to obtain at least 50 points (over 100).

If a student decides not to follow the continuous assessment (that is, he/she renounces the marks obtained in the continuous evaluation), he/she will be entitled to take a final exam (same date and place as the ordinary exam). The grade obtained in this exam is equivalent to 60% of the final grade (that is, it is necessary to get 8.3 over 10 to pass the subject).

In the extraordinary call, the final exam will be 100% of the grade. The continuous evaluation may be applied if it is more beneficial for the student (partial tests and 50% of the final exam grade).

A solution (for example, the implementations of data structures and their algorithms) will be considered correct only if it meets the following criteria:
- It should meet the specifications described in the problem statement.
- Its implementation should be robust and correct. That is, in addition to not containing syntax errors, for each possible input it receives, it produces the correct output. The objective of unittests is that students can easily verify the correctness of their algorithms.
- The solution should be as efficient as possible, both in terms of temporal and spatial complexity.
- The solution should be clear, clean and fully legible, and understandable. It should be easy to maintain. A solution that does not meet these criteria will not be evaluated. The implementation should follow the Zen of Python (https://peps.python.org/pep-0020/). The code should be refactored.

BASIC BIBLIOGRAPHY

- Karumanchi, N Data Structure and Algorithmic Thinking with Python: Data Structure and Algorithmic Puzzles. , CareerMonk Publications, 2015

- Michael T. Goodrich and Roberto Tamassia Data Structures and Algorithms in Python, , John Wiley & Sons, 2013

ADDITIONAL BIBLIOGRAPHY

- Isabel Segura Bedmar, Harith AlJumaily, Julian Moreno Schneider, Juan Perea & Nathan D. Ryan Algorithms and Data Structures, OCW-UC3M: http://ocw.uc3m.es/ingenieria-informatica/algorithms-and-data-structures, 2011

BASIC ELECTRONIC RESOURCES

- Isabel Segura Bedmar, Harith AlJumaily, Julian Moreno Schneider, Juan Perea & Nathan D. Ryan . ALGORITHMS AND DATA STRUCTURES: http://ocw.uc3m.es/ingenieria-informatica/algorithms-and-data-structures