

Programming Techniques

Code: 43851
ECTS Credits: 6

Degree	Type	Year	Semester
4315985 Geoinformation	OT	0	2

The proposed teaching and assessment methodology that appear in the guide may be subject to changes as a result of the restrictions to face-to-face class attendance imposed by the health authorities.

Contact

Name: Wladimir Szczerban

Email: Wladimir.Szczerban@uab.cat

Use of Languages

Principal working language: spanish (spa)

Teachers

Wladimir Szczerban

Carles Ros Visus

Prerequisites

There are no prerequisites. However it is advisable that students have basic knowledge on information technologies.

Objectives and Contextualisation

Programming techniques are the field of computer science dedicated to software applications development. Some of the subjects it encompasses are object-oriented programming, functional programming, recursive programming and web applications programming, among others. This course is focused on object-oriented programming and web applications programming.

Competences

- Apply programming methodologies and procedures, and those for implementation of geospatial applications for different types of platforms (desktop, web, mobile), using different programming paradigms and environments.
- Continue the learning process, to a large extent autonomously.
- Develop imaginative, creative and innovative ideas in projects for geospatial information systems, services, products or applications.
- Use acquired knowledge as a basis for originality in the application of ideas, often in a research context.

Learning Outcomes

1. Continue the learning process, to a large extent autonomously.
2. Develop imaginative, creative and innovative ideas in projects for geospatial information systems, services, products or applications.
3. Know and apply the different programming paradigms and environments.

4. Programme applications using integrated development environments.
5. Use acquired knowledge as a basis for originality in the application of ideas, often in a research context.

Content

Object-oriented programming

1. Introduction to object-oriented programming (OOP).

Concepts of object, class and method.

Introduction to *Java* programming.

Introduction to *Eclipse* development environment.

Our first program.

2. Data types.

Data types.

Variables declaration.

Arithmetic, logic and comparison operators.

Reserved words in *Java*.

3. Object manipulation.

Advantages of using objects.

Introduction to methods.

Data reading from the keyboard.

4. Conditional statements.

5. Iterative statements.

6. *Arrays*.

7. Principles of object-oriented programming.

Encapsulation.

Inheritance.

Polymorphism.

Abstract classes and methods.

Web programming

1. Introduction to web and HTML programming.

Client-server architecture.

Basic elements of HTML language.

HTML5 vs HTML4.

2. *Cascading Style Sheets (CSS).*

How to insert CSS.

Types of selectors.

Box model.

Positioning.

Colours in HTML.

3. *JavaScript.*

Object-oriented programming.

How to insert *JavaScript*.

Data types.

Variables.

Operators.

Control statements.

Predefined objects.

Functions

Browser Object Model.

Document Object Model.

JavaScript Object Notation (JSON).

Asynchronous JavaScript and XML (AJAX).

4. *Hypertext PreProcessor (PHP).*

Introduction.

Form processing example.

Database access example.

5. Model-View-Controller paradigm.

Introduction.

Data flow.

Methodology

Learning is achieved by means of three types of activities:

Directed activities: Directed activities are theoretical and practical lectures in a computer lab. They include solving case studies and practical exercises, using as the main method a problem based learning approach. Lectures serve to systematize all the content, to present the state of the art of the different subjects, to provide methods and techniques for specific tasks, and to sum up the knowledge to learn. Lectures organize also the autonomous and complementary work done by the students.

Supervised activities: Supervised activities are focused on the execution of a semester project, consisting of a real case study, carried out through workshop hours, autonomous work and tutorials. This semester project allows to apply together all the knowledge and technical skills learnt in all the courses of the semester. The semester project is a milestone for the students and the actual demonstration that they had achieved the learning goals of all the courses of the semester. It is also the main evidence for evaluation as students should have to submit at the end of the semester a report that summarizes the whole project and do an oral presentation.

Autonomous activities: Autonomous work of the students includes personal readings (papers, manuals, relevant reports, etc.), data and documentation search, complementary exercises and the personal development of the semester project.

The activities that could not be done onsite will be adapted to an online format made available through the UAB's virtual tools. Exercises, projects and lectures will be carried out using virtual tools such as tutorials, videos, Teams sessions, etc. Lecturers will ensure that students are able to access these virtual tools, or will offer them feasible alternatives.

Annotation: Within the schedule set by the centre or degree programme, 15 minutes of one class will be reserved for students to evaluate their lecturers and their courses or modules through questionnaires.

Activities

Title	Hours	ECTS	Learning Outcomes
Type: Directed			
Theoretical and practical lectures	36	1.44	3, 4, 1
Type: Supervised			
Semester project	15	0.6	3, 4, 2, 1
Type: Autonomous			
Personal work	69	2.76	3, 4, 2, 1

Assessment

In the event that assessment activities cannot be taken onsite, they will be adapted to an online format made available through the UAB's virtual tools (original weighting will be maintained). Homework, activities and class participation will be carried out through forums, wikis and/or discussion on Teams, etc. Lecturers will ensure that students are able to access these virtual tools, or will offer them feasible alternatives.

CONTINUOUS EVALUATION

a) Evaluation procedure and activities:

Evaluation of the course is based mostly on the semester project, that comprises two evaluation activities. The elaboration and submission of a synthesis report and the oral presentation of the project done. Given the technical content of the course, the weight assigned to the project report is 45% of the total course grading, assuming that it is the most appropriate means to explain all the technical details of the project, and a weight of 25% at the oral presentation. The course assessment is completed with the evaluation of the practical exercises done along the course, that account for another 30% of the total course grading.

Except when expressly noticed, all the evaluation activities (report and oral presentation of the semester project, as well as practical exercises) have to be carried out individually.

Time assigned to each evaluation activity includes the time spent in making all the material evidences for evaluating each activity (e.g., writing of the report, preparing the presentation slides, etc.).

b) Evaluation schedule:

2nd semester project report: Making during all the semester. Submission at the end of semester, on April 3rd 2022.

2nd semester project oral presentation: Making during all the semester. Oral presentation at the end of semester, on April 8th 2022.

Course practical exercises: Making and submission weekly or biweekly along the semester.

c) Grade revision:

Once the grades obtained are published, students will have one week to apply for a grade revision by arranging an appointment with the corresponding teachers.

d) Procedure for reassessment:

2nd semester project report: It could be reassessed in the following two weeks after the submission date scheduled. Reassessment will require the submission of a new whole report in case of negative evaluation of the former report submitted.

2nd semester project oral presentation: It could be reassessed in the following week after the date scheduled for the oral presentation. Reassessment will require doing again the oral presentation in case of negative evaluation of the former presentation done.

Course practical exercises: Can not be reassessed.

To have right to a reassessment the student will have to have been previously evaluated in a set of activities that account for at least two thirds of the total course grading. Therefore he or she will have to have been evaluated of the 1st semester project report (50%) and of the 1st semester project oral presentation (20%) in the dates scheduled.

The right to a reassessment will only be granted to students that, having not passed the course (e.g., having a total course grade below 5 over 10), had obtained at least a total course grade above 3,5 over 10.

e) Conditions for a 'Not assessable' grade:

Students will receive the grade 'Not assessable' instead of 'Fail' if they had not submitted neither the 2nd semester project report nor the 2nd semester project oral presentation. That is, if they only submit all or part of the course practical exercises.

f) UAB regulations on plagiarism and other irregularities in the assessment process:

In the event of a student committing any irregularity that may lead to a significant variation in the grade awarded to an assessment activity, the student will be given a zero for this activity, regardless of any disciplinary process that may take place. In the event of several irregularities in assessment activities of the same subject, the student will be given a zero as the final grade for this subject.

Assessment activities with a zero grade because of irregularities can not be reassessed.

Assessment Activities

Title	Weighting	Hours	ECTS	Learning Outcomes
Oral presentations	25	6	0.24	3, 4, 1, 5

Practical exercises	30	9	0.36	3, 4, 2, 1
Report submissions	45	15	0.6	3, 4, 1

Bibliography

Sikora, Zbigniew M, *Java: practical guide for programmers*. Amsterdam, Boston: Morgan Kaufmann, 2003.

Terry Felke-Morris, *Web development and design foundations with HTML5*. Addison-Wesley, 2012.

Medinets, David, *PHP3 programing browser-based applications*. McGraw-Hill, cop. 2000.

During the course web reference resources will be given and its use explained.

Software

Eclipse

XAMPP

Nodejs

Visual Studio Code