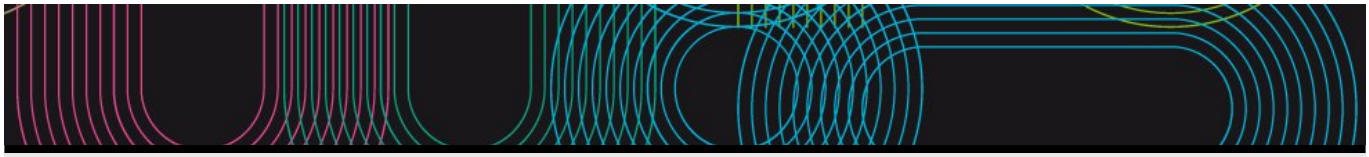

IDENTIFYING DATA				
Programming I				
Subject	Programming I			
Code	V05G300V01205			
Study programme	(*)Grao en Enxeñaría de Tecnoloxías de Telecomunicación			
Descriptors	ECTS Credits	Choose	Year	Quadmester
	6	Mandatory	1st	2nd
Teaching language	Spanish			
Department				
Coordinator	Rodríguez Hernández, Pedro Salvador			
Lecturers	García Palomares, Ubaldo Manuel Rodríguez Hernández, Pedro Salvador Santos Suárez, José Manuel			
E-mail	pedro.rodriguez@uvigo.es			
Web	http://faitic.uvigo.es			
General description	The aim of the course is to provide students with basic skills to program in a high level language.			

Competencies	
Code	
B4	CG4: The ability to solve problems with initiative, to make creative decisions and to communicate and transmit knowledge and skills, understanding the ethical and professional responsibility of the Technical Telecommunication Engineer activity.
B9	CG9: The ability to work in multidisciplinary groups in a Multilanguage environment and to communicate, in writing and orally, knowledge, procedures, results and ideas related with Telecommunications and Electronics.
C6	CE6/T1: The ability to learn independently new knowledge and appropriate techniques for the conception, development and exploitation of telecommunication systems and services
C12	CE12/T7: The knowledge and use of basics in telecommunication networks, systems and service programming.
D2	CT2 Understanding Engineering within a framework of sustainable development.
D4	CT4 Encourage cooperative work, and skills like communication, organization, planning and acceptance of responsibility in a multilingual and multidisciplinary work environment, which promotes education for equality, peace and respect for fundamental rights.

Learning outcomes			
Expected results from this subject	Training and Learning Results		
Express the solution of a simple problem by means of algorithms using top-down design.		C12	
Identify the data needed to solve a problem and associate them with appropriate datatypes based on their features (size, range, associated operators)		C12	
Code simple algorithms using the basic types of statements: assignment, selection and iteration.		C12	
Declare and define functions with a proper use of parameters.		C12	
Handle I/O operations and file management.		C12	
Define and use structured data types.		C12	
Define and manage dynamic data structures (lists, stacks, queues and trees).		C12	
Create modules and library functions and use them in programs.		C6 C12	
Predict the result of a sequence of statements, knowing the input data.		C12	
Handle basic tools in an integrated development environment: text editor, compiler, linker, debugger and documentation tools.		C6	
Develop a small scale project following all the phases: requirements analysis, design, implementation, testing and documentation.	B4 B9	C6 C12	D2 D4

Contents	
Topic	
Lecture 1: The algorithm and the programming languages.	1. The algorithm and its different representations: flowchart, pseudocode, natural language. 2. Algorithm implementation by means of a programming language. Programming paradigms: modular programming and structured programming. 3. C language and the function main(). Source code and object code. The compiler and the interpreter. 4. Input/output exercises: human-computer interface. The standard input/output files: stdin, stdout. The #include directive. Library functions.
Lecture 2: Grammar and basic elements of C language.	1. The alphabet. Recursive derivations of sintactically valid sequences. Identifiers, numbers. Symbolic constants: The #define directive and macros. Use of the const qualifier. 2. Variables and their attributes: name, value, address, types. Pointer variables. Declaration of simple variables and pointers: the direction & and reference * operators. 3. The sizeof operator. Arithmetical operators. The assignment operator. Automatic type conversion and by means of the cast operator. 4. Syntactic notation for expressions and statements. Simple and compound statements.
Lecture 3: Sequential, iteration and selection statements	1. Evaluation of expressions with relational operators and boolean operators. 2. Decision statements: switch, if, nested if. The ternary operator (?:) 3. The iterative statements and their importance in modular programming: while, do while and for. The break and continue statements.
Lecture 4: Functions: Introduction	1. Pointer arithmetic. Arrays and pointers. Dynamic memory allocation to 1 and 2 dimension arrays: the malloc(), calloc(), realloc() functions. 2. Arrays of characters: The end of string character. Library functions for dealing with arrays of characters. 3. Functions declaration and definition. Local variables in a compound statement. Parameter passing by value and by reference: use of pointers. Function return. 4. Static variables and global variables.
Lecture 5: Functions: special cases	1. Command line arguments passing: argc and argv. 2. Recursive functions: advantages and disadvantages. 3. Creation and use of function libraries. The conditional directives in a header file. 4. Functions that return addresses.
Lecture 6: struct variables	1. struct variables: global declaration. Fields of a struct. Pointers to struct. The . (Point) and -> (arrow) operators. 2. struct and a pointer to struct as a funcion parameter and return value. 3. typedef with non trivial declarations. 4. More complex data structures: nested structs, array of structs. 5. Dynamic management in creating linear lists, circular lists and trees. 6. Insertion and removal of variables in a list.
Lecture 7: Files	1. Text files: fopen() and fclose() functions. 2. Different file input/output functions: fprintf (), fscanf(), fgets(), feof(). 3. Functions with direct access to files. 4. Information management between files and lists. 5. Node structure in simple linked lists. 6. File to list conversion and vice versa.

Planning			
	Class hours	Hours outside the classroom	Total hours
Introductory activities	2	0	2
Master Session	24	24	48
Laboratory practises	12	16	28
Projects	8	24	32
Practical tests, real task execution and / or simulated.	5	15	20
Troubleshooting and / or exercises	5	15	20
*The information in the planning table is for guidance only and does not take into account the heterogeneity of the students.			

Methodologies

	Description
Introductory activities	Introduction to theoretical and practical activities.
Master Session	Plenary sessions that include the realisation of works and programs. Through this methodology the competencies CE12 and CT2 are developed.
Laboratory practises	During the first weeks of the term the student codifies, compiles and documents programs guided by the instructor. Some of these activities will be evaluated. Through this methodology the competencies CG4, CE12 and CT2 are developed.
Projects	In the last part of the term, the student must complete a low complexity project, under the instructor supervision, which includes individual and in group activities. Through this methodology the competencies CG4, CG9, CE6, CE12, CT2 and CT4 are developed.

Personalized attention

Methodologies	Description
Master Session	The webpage of the course informs on the prescheduled office hours that students can consult the instructors. This consulting will be devoted to answer the questions posed in the lectures, laboratory activities and the development of the project.
Laboratory practises	The webpage of the course informs on the prescheduled office hours that students can consult the instructors. This consulting will be devoted to answer the questions posed in the lectures, laboratory activities and the development of the project.
Projects	The webpage of the course informs on the prescheduled office hours that students can consult the instructors. This consulting will be devoted to answer the questions posed in the lectures, laboratory activities and the development of the project.

Assessment

	Description	Qualification	Training and Learning Results
Projects	The student will develop a project in the last weeks of the term, and will submit a report. The project will be assessed in the final laboratory test.	30	B4 C6 D4 B9 C12
Practical tests, real task execution and / or simulated.	Every 4 weeks, the student will take a practical individual test in the laboratory. At the end of the term, the student will take a comprehensive final practical test. All of them will consist in the development of a program in the computer. Those tests will assess the student's progress with the laboratory practices and with the project.	20	C6 C12
Troubleshooting and / or exercises	Every 4 weeks, the student will take a test that will assess the advancement of the students skill in the resolution of problems. At the end of the term, the student will take a comprehensive final test on the whole contents of the subject.	50	B4 C12

Other comments on the Evaluation

The **course planning in lectures** and the estimated time of the **most important assessment milestones** is detailed below:

- Week 1: Lecture 1/2
- Week 2: Lecture 3 - Practice 1
- Week 3: Lecture 3 - Practice 1/2
- Week 4: Lecture 4 - **Theory Test 1** (PT1) - **Laboratory Test 1** (PP1)
- Week 5: Lecture 4 - Practice 2/3
- Week 6: Lecture 4 - Practice 3/4
- Week 7: Lecture 5 - Practice 4/5
- Week 8: Lecture 5 - **Theory Test 2** (PT2) - **Laboratory Test 2** (PP2)
- Week 9: Lecture 5/6 - Practice 5/6
- Week 10: Lecture 6 - Practice 6 - Project (1h)
- Week 11: Lecture 6 - Project (2h)
- Week 12: Lecture 7 - Project (1h) - **Theory Test 3** (PT3) - **Laboratory Test 3** (PP3)
- Week 13: Lecture 7 - Project (2h)

- Week 14: Project (2h)
- Finals: **Final Theory Test** (PFT) - **Final Laboratory Test** (PFP)

In all courses the School offers two evaluation modes: **Continuous evaluation** and **comprehensive evaluation**.
The student must opt to the latter one explicitily, no latter than the week before the Theory Test 2 (PT2) is taken.
The **continuous evaluation** will be considered as "passed" if the student has submitted a report for the project developed from the 10th to the 14th week, and if the final grade obtained by the student is at least 5. This final grade is the harmonic mean between the theory and laboratory, calculated as follows:
NF = (2*NT*NP)/(NT+NP)
where:

- NP = 0.1*PP1+0.1*PP2+0.2*PP3+0.6*PFP
- NT = 0.1*PT1+0.1*PT2+0.2*PT3+0.6*PFT

The Final Theory Test (PFT) assesses the mastership of the contents explained in the lectures.
The Final Practice Test (PFP) assesses the proper application and coding in C to deal with a medium level project. Indirectly, it also assesses the mastership of the contents introduced in the lectures and the laboratory practices.
The use of the harmonic means implies that the course is not passed if either NP or NT has a grade under 3.3.
No test in the continuous evaluation mode is repeatable; that is, the instructor has no obligation to reschedule an evaluated activity missed by a student.
The date and procedures for the revision of the grades will be known before the evaluation tests. The students will have the chance of reviewing the grades preferably within two weeks after the evaluation.
In order to pass the course by the **comprehensive evaluation mode**, the student must submit a project report similar to the one submitted by the continuous evaluation students, and the final grade obtained by the student must be at least 5. This mode will consist of a theory test (PFT) and a laboratory test (PFP, which will include the evaluation of the project). The final grade is the harmonic mean between the theory and practice, calculated as follows:
NF = (2*NT*NP)/(NT+NP)
where:

- NP = PFP
- NT = PFT

Both the **continuous evaluation grade** and the **comprehensive evaluation grade** will be computed to all students that take the final tests (theory and practice). The final grade will be the higher one.
A "No Present" grade will be granted:

- If the student opts for the continuous evaluation mode, when no test is taken after the Laboratory Test 1 (PP1)
- If the student opts for the comprehensive evaluation mode, when no final test (PFT and PFP) is taken.

University regulations allow students to take an additional test to approve the course (extra evaluation). In order to pass the course using this extra evaluation, the student must submit a project report similar to the one submitted by the continuous evaluation students, and the final grade obtained by the student must be at least 5. This extra evaluation will consist of a theory test and a laboratory test (which will include the evaluation of the project). The final grade is the harmonic mean between the theory and practice, calculated as follows:
NF = (2*NT*NP)/(NT+NP)
where:

- If the student takes the extra theory test, NT will be the grade achieved in that test. Otherwise, NT will be the theory grade obtained for the theoretical tests in his/her regular evaluation.
- If the student takes the extra laboratory test, NP will be the grade achieved in that test. Otherwise, NP will be the practice grade obtained for the practical tests in his/her regular evaluation.

All the partial and final grades will only be valid for the term the student is enrolled to, that is, in case the student repeates the subject, he or she will not retain any of the grades of the previous year.

If plagiarism is detected in any of the works/test taken, the student will receive a failing grade (0) and the professors will report the fact to the school authorities.

Sources of information
Manuel Caeiro Rodríguez, Enrique Costa Montenegro, Ubaldo García Palomares, Cristina López Bravo, J, **Practicar Programación en C**, 2014,
Osvaldo Cairo Battistuti, **Fundamentos de Programación**, 2006,
José Rafael García-Bermejo Giner, **Programación Estructurada en C**, 2008,
Brian W. Kernighan & Dennis M. Ritchie, **El Lenguaje de Programación C**, 1986 (reimpreso en 1995),
James L. Antonakos & Kenneth C. Mansfield Jr., **Programación Estructurada en C**, 1997 (reimpreso en 2004),
Jorge A. Villalobos S. & Rubby Casallas G., **Fundamentos de Programación: Aprendizaje Activo Basado en Casos**, 2006,
Web resources

- http://www.Cprogramming.com
- José R. García-Bermejo Giner: http://maxus.fis.usal.es/FICHAS_C.WEB/11xx_PAGS/11xx.html

Recommendations
Subjects that continue the syllabus
Programming II/V05G300V01302

Subjects that it is recommended to have taken before
Informatics: Computer Architecture/V05G300V01103

Other comments
Programming II course continues this course in the second year.