Subject:	Imperative Programming	**Code**	72.31
Credits:	9		
Department	Digital Systems and Data	**Version**	2019

Course: Computer Science Engineering, Mechanical Engineering

Curriculum: M22, S10 A - Rev18, S10-Rev23, S10 - Rev18

Objectives:

No.	Description
1	Apply modularized design techniques to decompose problems.
2	Use and adapt standard algorithms.
3	Write correct, efficient and well-styled algorithms for cross-platform environments.

Contents:

Characteristics of the Imperative Paradigm. The algorithm as a computational process model. Software testing and debugging. Syntax and basic semantics of high-level languages. Simple data types, operators and expressions. Variables and assignment. Data input and output. Control structures. Function concept. Use and construction of libraries. Recursion. Concept of pointers and references. Types of structured data : arrays, strings and records. File handling. Application workshop with an imperative ANSI C type language under Linux.

Required bibliography:

No.	Description
1	No bibliography has been uploaded.

Optional bibliography:

No.	Description
1	No additional bibliography has been uploaded.
2	No additional bibliography has been uploaded.
3	No additional bibliography has been uploaded.

Subject:	Imperative Programming	**Code**	72.31
Credits:	9		
Department	Digital Systems and Data	**Version**	2019

Course transcript:

No.	Description
1	**Introduction to Programming** Modularized and structured programming. Good programming style. Concept of Algorithm. Pseudocode and refinement. High level languages. Compilers and preprocessors. Software testing: black box, white box and Pandora's box.
2	**Types, Operators and Expressions** Data types and size. Constants. Declarations. Operators: arithmetic, relational, logical, increment and decrement, bitwise and conversion. Assignment operators and expressions. Operator precedence. Basic I/O instructions: printf and getchar. Input and output redirection.
3	**Control structures.** Prepositions and blocks. Storage and scope of variables. External and static variables. Decision structures: if-else and switch. Repetition structures: while, for and do-while. Use of break y continue. goto instruction
4	**Functions.** Concept of function. Prototyping. Input parameters. Standard C library (basic functions). Header files. The C Preprocessor. Input and output parameters. Formatted input: scanf and sscanf. Construction of libraries. Functions with variable arguments. Recursion
5	**Arrays.** One-dimensional arrays. Character arrays. Array ordering. Searching in one-dimensional arrays. Multidimensional arrays.
6	**Files** Concept of data flow: stream. File access. Binary and text files. Error handling. File processing in C language
7	**Advanced C language concepts** Pointers. Pointer arithmetic. Pointers to characters. Structures. Unions. Use of the Heap. Pointers to functions. Generic pointers. Implementation of ADTs (Abstract Data Types): stacks, queues, lists.

Practical assignments:

No.	Description
1	**Practical assignments work** To integrate all the topics of the subject, students carry out a special mandatory group work.

Laboratory assignments:

No.	Description
1	**Laboratory Practice** For all topics, students carry out individual practical work in the laboratory. Practical exercise guides are available for this, as well as detailed solutions for the individual exercises.

Professor in charge:	Garberoglio, Marcelo Fabio
Head of Department:	Bolo, Mario Enrique