UNIVERSITÀ DEGLI STUDI DI PALERMO

DEPARTMENT	Ingegneria		
ACADEMIC YEAR	2023/2024		
BACHELOR'S DEGREE (BSC)	CYBERNETIC ENGINEERING		
INTEGRATED COURSE	ELECTRONIC CALCULATORS - INTEGRATED COURSE		
CODE	18794		
MODULES	Yes		
NUMBER OF MODULES	2		
SCIENTIFIC SECTOR(S)	ING-INF/05		
HEAD PROFESSOR(S)	GENTILE ANTONIO	Professore Associato	Univ. di PALERMO
OTHER PROFESSOR(S)	AGATE VINCENZO	Ricercatore a tempo determinato	Univ. di PALERMO
	GENTILE ANTONIO	Professore Associato	Univ. di PALERMO
CREDITS	12		
PROPAEDEUTICAL SUBJECTS			
MUTUALIZATION			
YEAR	1		
TERM (SEMESTER)	Annual		
ATTENDANCE	Not mandatory		
EVALUATION	Out of 30		
TEACHER OFFICE HOURS	**AGATE VINCENZO** Tuesday 15:00 17:00 Ricevimento in modalita telematica **GENTILE ANTONIO** Friday 10:00 12:00 Studio del docente presso DINFO, Edificio 6, III pianoDietro prenotazione per email/sito o telefono: 091-238.62603		

DOCENTE: Prof. ANTONIO GENTILE

PREREQUISITES	None
LEARNING OUTCOMES	Knowledge and understanding At the end of the course, the student will acquire both a good knowledge of information representation techniques and its processing through sequential and logic circuits, and a good knowledge of the optimization techniques of such systems. The student will also learn the syntax and programming techniques in C language, data structures and fundamental algorithms. Ability to apply knowledge and understanding The student will be able to apply the techniques studied to design logic and sequential circuits at a logical and functional level and analyze their functioning. The student will be able to use development tools and environments for programming in C language and will be able to implement software solutions. Autonomy of judgment The student will be able both to carry out the analysis of a problem and to design, starting from a verbal description, a suitable software solution. He will be able to evaluate the quality of a software solution in terms of simplicity, readability, efficiency and possibility of reuse. He will be able to understand how a computer works. Communication skills The student will acquire the ability to communicate and express problems relating to the subject of the course. He will be able to describe a logic circuit with appropriate terminology. He will be able, using a simple and clear language, to describe the analysis and synthesis processes of software solutions. Learning skills The student will have the ability to apply the methodologies studied in different contexts and to learn analysis and synthesis processes related to software programs in structured programming and logic circuits.
ASSESSMENT METHODS	Learning is assessed through two written tests that focus on the two modules of the course and a a possible oral interview. The written test concerning the topics of the Digital Computation module consists mainly in the answer to some questions with closed or open answers and in questions about the design of sequential and combinational circuits. The questions will tend to verify a) the acquired knowledge; b) the ability to design circuits that meet certain specifications, c) the ability to organize and display technical knowledge. The maximum score is obtained if the verification ascertains the full possession of the following three aspects: ability to carry out the questions concerning the representation of information, the ability to design or optimize logic circuits that perform specific tasks, the ability to describe and compare different circuit solutions. The written test on the topics of the Fundamentals of Informatics module will consist of programming exercises, relating to the generation of one or more C language programs, based on some technical specifications, and open-ended questions. The test will verify the knowledge of the C programming language, the ability to find solutions to simple problems typical of the discipline, the ability to implement running software, the knowledge of the topics covered during the course.The evaluation will be on a range from 18 to 30 cum laude. The final grade will be calculated as the average of the evaluations of the 3 tests. The general assessment is based on the following general criteria: a) excellent (30 - 30 cum laude): excellent knowledge of the topics, excellent language properties, good analytical skills, the student is able to apply the knowledge to solve the proposed problems; b) very good (26 - 29): good command of the topics, full ownership of language, the student is able to apply the knowledge to solve the proposed problems; c) good (24 - 25): basic knowledge of the main topics, fair language properties with limited ability to autonomously apply the knowledge to solve the proposed problems; d) fair (21 - 23): limited command of the main teaching topics, fair language properties, fair ability to independently apply the acquired knowledge; e) sufficient (18 - 20): minimal basic knowledge of the main teaching topics and technical language, poor ability to independently apply the acquired knowledge f) insufficient: does not possess a minimum acceptable knowledge of the contents of the topics covered in the teaching.
TEACHING METHODS	Class lectures Lab and practice sessions

MODULE
LOGIC CIRCUITS

Prof. ANTONIO GENTILE

SUGGESTED BIBLIOGRAPHY	
1. M. Morris Mano, Charles R. Kime, Reti Logiche, III Edizione italiana, Pearson Education Italia, ISBN: 88-7192-142-9	
AMBIT	50283-Matematica, informatica e statistica
INDIVIDUAL STUDY (Hrs)	96
COURSE ACTIVITY (Hrs)	54
EDUCATIONAL OBJECTIVES OF THE MODULE	
The student will be able to apply the methodologies studied in different contexts and to learn analysis and synthesis processes related to logic circuits.	

SYLLABUS

Hrs	Frontal teaching
12	Introduction to the course, the concept of information and its logarithmic nature. Representation of integers in a generic basis, binary numbers. Conversion of numbers from one base to another, representation of floating point numbers and negative numbers. Numerical representation of images and sounds, ASCII code. Introduction to Boolean algebra. Logic gates, Boolean functions, negated logic, theorems and properties of algebra, normal forms. Karnaugh maps, boolean function minimization. Synthesis of combinatorial networks. Integrated circuits (full adder, adders, multiplexers, decoders).
16	Sequential networks. Mealy and Moore models. Latches (SR, D), Flip-Flops (JK, T, D, SR). Master slave. Analysis and design of synchronous sequential circuits. State diagrams, Flip Flop characteristic equations, Design of sequential networks with Flip Flops of various types (D, T, JK, SR). Sequence recognizers. Registers, counters, ROM, PAL, PLA
8	Computer architecture (von Neumann model), CPU, ALU, RAM, BUS. What is a program. CISC and RISC systems. Microprograms and wired logic. Introduction to operating

Hrs	Practice
4	How to represent information. Boolean algebra
7	Analysis and Design of combinational networks. Karnaugh maps; Normal forms
7	Analysis and Design of sequential networks. Design of sequence recognizers

MODULE
PRINCIPLES OF COMPUTER SCIENCE

Prof. VINCENZO AGATE

SUGGESTED BIBLIOGRAPHY

1. J. Glenn Brookshear - Stephen G. Kochan, Fondamenti di informatica e programmazione in C, Pearson, ISBN-10: 8865183691
2. Brian W. Kernighan , Dennis M. Ritchie, Il linguaggio C - Principi di programmazione e manuale di riferimento, Pearson, ISBN-10: 8891908231
3. Paul J. Deitel, Harvey M. Deitel, Maselli, Il linguaggio C, Fondamenti e tecniche di programmazione, ISBN-10: 8891901652, Pearson
Per approfondimenti: Igor Zhirkov, Low-Level Programming: C, Assembly, and Program Execution on Intel® 64 Architecture, ISBN-13 (pbk): 978-1-4842-2402-1, ebook disponibile su SpringerLink per gli utenti Unipa
Inoltre, Al Aho and Jeff Ullman, Foundations of Computer Science, libro fuori commercio ma reso disponibile online dagli autori al link http://infolab.stanford.edu/~ullman/focs.html

AMBIT	50283-Matematica, informatica e statistica
INDIVIDUAL STUDY (Hrs)	96
COURSE ACTIVITY (Hrs)	54

EDUCATIONAL OBJECTIVES OF THE MODULE

At the end of the course, the student will be able to evaluate, analyze, communicate and implement possible software solutions to simple problems using the acquired skills of the C language.

SYLLABUS

Hrs	Frontal teaching
3	Introduction to the course. Concept of algorithm, representation of algorithms, pseudo code and flow-charts.
2	Programming languages, historical perspective. Compilation of programs. Machine independence. Concepts of traditional programming. Procedural units. Language implementation. Compilation of the first program. Run the first program.
4	Variables, data types and arithmetic expressions. Difference between variables and constants. Assignment. Operators. Priority among operators. Introduction to the C language. Variables, constants, types, operators in the C language.
3	Iterative algorithms. Efficiency and correctness of an algorithm. Introduction to the computational analysis of an algorithm: time and space complexity.
5	Iterative programs in C language. Control flow, iterative structures (for, while and do-while). Structures of selection (if-else) and selection among several alternatives (switch-case). Nesting of structures. Understanding and simulation of algorithms / programs.
4	Functions and structure of programs. Arguments, return values and local variables. Visibility and scope of a variable. Global, automatic and static variables. Body of a function and function calls. Recursion. Modular software design: bottom-up and top-down techniques.
8	The concept of abstraction. Elementary data structures. Arrays, matrices, records, lists, queues and stacks and their logical functioning. Pointers in C language. Pointers and arrays. Pointer arithmetics. Multi-dimensional arrays. Implementation of custom structures and types. Implementation of elementary data structures. Data search and sorting in data collections.
2	Character strings. Using strings in C: character arrays, variable length character strings, escape sequences.
2	Input and output management in C. File management. Enumerations. Preprocessor.

Hrs	Practice
6	Design of simple algorithms through flow charts. From flowcharts to C code. First C programs: compilation, linking and execution. Implementation of programs for the manipulation of numerical data.
3	Design of algorithms involving selection and iterative structures through flow-charts. Corresponding implementation in C language.
3	Exercises on modular software design through correct use of functions. Design and implementation of algorithms that use recursion.
6	Design and implementation of algorithms using character strings, vectors and arrays. Pointers. Implementation of lists, stacks and queues.
3	Software implementation for file I / O. Preparation for the written test.