UNIVERSITÀ DEGLI STUDI DI PALERMO

DEPARTMENT	Ingegneria		
ACADEMIC YEAR	2019/2020		
BACHELOR'S DEGREE (BSC)	DIGITAL ENTERPRISE INNOVATION ENGINEERING		
INTEGRATED COURSE	ELECTRONIC CALCULATORS - INTEGRATED COURSE		
CODE	18794		
MODULES	Yes		
NUMBER OF MODULES	2		
SCIENTIFIC SECTOR(S)	ING-INF/05		
HEAD PROFESSOR(S)	GAMBINO ORAZIO	Ricercatore	Univ. di PALERMO
OTHER PROFESSOR(S)	GAMBINO ORAZIO	Ricercatore	Univ. di PALERMO
CREDITS	12		
PROPAEDEUTICAL SUBJECTS			
MUTUALIZATION			
YEAR	1		
TERM (SEMESTER)	Annual		
ATTENDANCE	Not mandatory		
EVALUATION	Out of 30		
TEACHER OFFICE HOURS	**GAMBINO ORAZIO** Monday 10:00 12:00 Chat di Teams, previo appuntamento concordato via email. Tuesday 10:00 12:00 Chat di Teams, previo appuntamento concordato via email.		

DOCENTE: Prof. ORAZIO GAMBINO

PREREQUISITES	Nothing.
LEARNING OUTCOMES	Conoscenza e capacita' di comprensione Lo studente acquisira' approfondita conoscenza della programmazione strutturata in linguaggio C. Conoscera' i principali strumenti di programmazione. Acquisira' elementi di rappresentazione delle informazioni nei calcolatori e metodologie di base per la progettazione e l'analisi di reti logiche combinatorie e sequenziali. Lo studente acquisira' conoscenze di base sulle architetture dei calcolatori. Capacita' di applicare conoscenza e comprensione Lo studente sara' in grado di valutare le possibili soluzioni software a problemi di complessita' media e affrontarne l'implementazione utilizzando strumenti e ambienti di sviluppo per la programmazione in linguaggio C. Sara' in grado di affrontare semplici problemi di rappresentazione binaria delle informazioni. Sara' in grado di progettare a livello funzionale circuiti logici per la soluzione di semplici problemi. Autonomia di giudizio Lo studente sara' in grado di affrontare in autonomia l'analisi, la progettazione e l'implementazione di software utilizzando la programmazione strutturata. Sara' in grado di valutare la qualita' del software in termini di semplicita, leggibilita, strutturazione ed efficienza. Abilita' comunicative Lo studente sara' in grado di esporre, efficacemente e con proprieta' di linguaggio, analisi e soluzioni di problemi affrontabili con la programmazione strutturata e con la progettazione funzionale di circuiti logici, nonche' di problemi di rappresentazione delle informazioni. Capacita' d'apprendimento Lo studente sara' in grado di affrontare in maniera autonoma problemi di programmazione strutturata individuando e integrando soluzioni parziali gia' disponibili, sia formalizzate sia implementate. Sara' in grado di approfondire in autonomia la conoscenza di moduli software e interfacce di programmazione. Sara' in grado di approfondire la conoscenza dei linguaggi e paradigmi di programmazione, dei sistemi operativi, delle architetture dei calcolatori e dei circuiti logici.
ASSESSMENT METHODS	La valutazione dell'apprendimento avviene mediante una prova in itinere e un progetto finale al calcolatore. La prova in itinere consiste nella compilazione di un questionario comprendente quindici quesiti a risposte multiple sugli argomenti relativi alle reti logiche, alla struttura dei calcolatori elettronici e alla rappresentazione dell'informazione a basso livello. La risposta a ogni quesito viene valutata con un punteggio pari a 2, se la risposta e' corretta, 0 se la risposta non e' fornita, o -1 se la risposta e' errata. Il voto della prova, in trentesimi, e' ottenuto sommando i punteggi di tutte le risposte ai quesiti. La prova in itinere e' ritenuta superata con voto uguale o superiore a 18/30. La prova al calcolatore richiede la soluzione di otto problemi di programmazione in Linguaggio C che concorrono alla costruzione di un programma applicativo completo e di due esercizi di implementazione, sempre in Linguaggio C, di algoritmi e strutture dati astratte. La risposta a ogni quesito viene valutata con un punteggio da 0, nel caso di soluzione non fornita, a 5, corrispondente a una soluzione priva di errori. Il voto della prova, in trentesimi, e' ottenuto mediante somma pesata dei punteggi di tutte le risposte ai quesiti. Gli studenti non in corso o che non abbiano superato la prova in itinere, sono tenuti a sostenere una prova complessiva integrante le due prove, con le medesime modalita, rispettivamente, della prova in itinere e della prova finale al calcolatore. I voti della prova scritta e di quella del progetto sono sommati con pesi rispettivamente di 1/4 e 3/4 per ottenere una valutazione complessiva in trentesimi. La discussione del progetto vertera' sulla realizzazione del progetto e sugli argomenti del corso. Il voto finale in trentesimi, nell'intervallo 18/30-30/30 con Lode, e' ottenuto mediante media delle valutazioni della prova orale e di quella complessiva relativa alla prova scritta e al progetto. La formulazione delle prove fornisce una valutazione dei risultati attesi in relazione al voto finale come segue: - da 18/30 a 20/30: sufficiente conoscenza e capacita' di comprensione degli argomenti trattati, capacita' di applicazione delle conoscenze acquisite per la risoluzione dei problemi proposti, autonomia di giudizio, abilita' comunicative e

	capacita' di apprendere. - da 21/30 a 23/30: discreta conoscenza e capacita' di comprensione degli argomenti trattati, capacita' di applicazione delle conoscenze acquisite per la risoluzione dei problemi proposti, autonomia di giudizio, abilita' comunicative e capacita' di apprendere. - da 24/30 a 26/30: buona conoscenza e capacita' di comprensione degli argomenti trattati, capacita' di applicazione delle conoscenze acquisite per la risoluzione dei problemi proposti, autonomia di giudizio, abilita' comunicative e capacita' di apprendere. - da 27/30 a 30/30 e lode: eccellente conoscenza e capacita' di comprensione degli argomenti trattati, capacita' di applicazione delle conoscenze acquisite perla risoluzione dei problemi proposti, autonomia di giudizio, abilita' comunicative e capacita' di apprendere.
TEACHING METHODS	Frontal lessons and exercises in classroom.

MODULE
LOGIC CIRCUITS

Prof. ORAZIO GAMBINO

SUGGESTED BIBLIOGRAPHY

- J. Glenn Brookshear, Stephen G. Kochan, “Fondamenti di informatica e programmazione in C”, Pearson
- M. Morris Mano, Charles R. Kime, “Reti Logiche”, Pearson

AMBIT	50283-Matematica, informatica e statistica
INDIVIDUAL STUDY (Hrs)	102
COURSE ACTIVITY (Hrs)	48

EDUCATIONAL OBJECTIVES OF THE MODULE

Il modulo si propone di fornire agli studenti i concetti di base sull'algebra di Boole e sulle reti logiche, e sulla rappresentazione e l’elaborazione delle informazioni dal livello del bit fino alle strutture dati astratte.
Gli studenti acquisiranno una conoscenza di base delle problematiche inerenti le metodologie di progettazione di reti logiche combinatorie e sequenziali e di algoritmi e strutture dati.

SYLLABUS

Hrs	Frontal teaching
4	Sistema di numerazione esadecimale. Codici. Codice BCD. Rappresentazione di interi con segno. Information representation. Internal and external representation.
4	Representation of information. Representation in complement to the base. Representation of real numbers: fixed point, floating point. Representation of alphanumeric characters. ASCII code. Representation of images.
3	Boolean Algebra. Operators and logic gates. Functions. Tables of truth. Diagrams and logic circuits. Fundamental identities. Duality principle. De Morgan's theorem. The complement of a function. Canonical forms. Minterms. Maxterms. Synthesis on two levels.
3	Boolean Algebra. Maps of Karnaugh. Implicants, prime implicants and essential prime implicants principles of Boolean functions. Minimization of Boolean functions. XOR operator. Functionally complete operators.
2	Combinatorial networks. Decoder and encoder. Expansion of decoders in series. Encoder with priority. Multiplexer and demultiplexer.
2	Combinatorial networks. Synthesis with decoder. Synthesis with multiplexer. Half-adder and Full-adder.
4	Sequential networks. Models by Mealy and Moore. Latch. Flip-Flop.
2	Sequential networks. Synthesis of synchronous sequential networks.
2	Algorithms and data structures. Search for values in an array: linear search and dichotomic search. Comparison-based sorting algorithms: sorting by selection, sorting by insertion, mergesort. Sorting algorithms not based on comparisons: Integer sort.
2	Algorithms and data structures. Stack and expressions in reverse Polish notation (RPN).
2	Algorithms and data structures. Basic operations on linked lists: creation of an element, addition to the head, scanning of the list.
2	Algorithms and data structures. Basic operations on lists: concatenation, addition to the queue, cancellation. Advanced operations on lists: inversion, ordering, map.
2	Algorithms and data structures. Binary trees. Binary search trees. N-th trees. Deep research.
2	Algorithms and data structures. Graphs. Graph visits.
2	Algoritmi e strutture dati. Operazioni di base su liste concatenate: creazione di un elemento, aggiunta in testa, scansione della lista, concatenazione, aggiunta in coda, cancellazione. Operazioni avanzate su liste: inversione, ordinamento, map.

Hrs	Practice
2	Representation of information. Boolean algebra.
2	Combinatorial networks
2	Sequential networks
2	Algoritmi e strutture dati. Ricerca di valori in un array: ricerca lineare e ricerca dicotomica. Algoritmi di ordinamento basati sui confronti: ordinamento per selezione, ordinamento per inserimento, mergesort. Integersort. Stack. Stack ed espressioni in notazione polacca inversa (RPN).
2	Algorithms and data structures. Binary trees. Binary search trees. N-th trees. Deep research. Graphs. Graph visits.

MODULE
PRINCIPLES OF PROGRAMMING

Prof. ORAZIO GAMBINO

SUGGESTED BIBLIOGRAPHY

- J. Glenn Brookshear, Stephen G. Kochan, “Fondamenti di informatica e programmazione in C”, Pearson
- B. W. Kernighan, D. M. Ritchie, “Linguaggio C”, Pearson

AMBIT	50283-Matematica, informatica e statistica
INDIVIDUAL STUDY (Hrs)	102
COURSE ACTIVITY (Hrs)	48

EDUCATIONAL OBJECTIVES OF THE MODULE

The module aims to provide students with the basic concepts necessary for understanding the structure of computers programmable digital electronics and Unix-like operating systems. Students will be able to analyze, communicate and implement possible software solutions to application problems using the acquired mastery of the C language.

SYLLABUS

Hrs	Frontal teaching
2	Programmable digital electronic computers. Structure of a calculator. The Von Neumann model: CPU, memory, input / output. Microprocessors.
2	Basic software and application software. Operating systems. Hardware abstraction. Driver. File system. Process. Unix-like systems. Command line interfaces, terminal and shell.
2	Programming languages and software development. Machine language. Compilation and interpretation. Source code, object code and executable code. Compilation: preprocessor, compiler, assembler and linker. Modules and libraries. Loading in memory and execution of programs. Command line arguments and exit code.
2	Structured programming. Variables and data types. Data structures. Constants and literals. Assignment instructions. Control instructions. Procedures and functions.
2	Data abstractions. Elementary data structures. Array. Lists, stacks and queues. Trees. Static and dynamic structures. Types of data defined by the user. Abstract data types.
2	Structured programming and language C. Structure of programs in C language. Pre-processor directives. Instructions and blocks. Functions. Variables and constants. Default data types. Main function. Input and output. Standard input / output. Formatted input and output. Standard library.
2	Structured programming and language C. Expressions. Arithmetic operators. Relational and logical operators. Bit-bit operators. Type conversions. Conditional expressions. Precedence and evaluation order.
2	Structured programming and C language. Increment and decrement operators. Iterative cycles: for, while and do-while. Instructions for interrupting and continuing cycles.
2	Structured programming and language C. Control structures. Construct if-else. Switch construct.
2	Structured programming and C language. Definitions and function prototypes. Formal parameters and automatic variables. Execution stack. Passing parameters by copy and by reference. Recursion.
2	Structured programming and language C. Pointers, addresses and memory management. Address arithmetic. Pointers to pointers. Generic pointers. Strings. Array. Dynamic memory allocation. Moving data into memory. String duplication.
2	Structured programming and language C. Aggregate types. Structures. Bit fields. Union. Definition of synonyms for types (typedef).
2	Structured programming and C language. File. Main file access operations.
2	Structured programming and language C. Preprocessor. Header file. Conditional compilation.
2	Structured programming and C language. The standard library. Mathematical functions.
2	Programmazione strutturata e linguaggio C. Puntatori a funzioni.

Hrs	Practice
2	Structured programming and language C. Structure of programs in C language. Pre-processor directives. Instructions and blocks. Functions. Variables and constants. Default data types. Function main. Input and output. Standard input/output. Formatted input and output. Standard library. Expressions. Arithmetic operators. Relational and logical operators. Bit-bit operators. Conversions of type. Conditional expressions. Precedence and evaluation order.
2	Structured programming and C language. Increment and decrement operators. Iterative cycles: for, while and do-while. Instructions for interrupting and continuing cycles.
2	Structured programming and language C. Control structures. Construct if-else. Switch construct. Definitions and function prototypes. Formal parameters and automatic variables. Execution stack. Passing parameters by copy and by reference. Recursion.
2	Structured programming and language C. Control structures. Construct if-else. Switch construct. Definitions and function prototypes. Formal parameters and automatic variables. Execution stack. Passing parameters by copy and by reference. Recursion.

2	Structured programming and language C. Pointers, addresses and memory management. Address arithmetic. Pointers to pointers. Generic pointers. Strings. Array. Dynamic memory allocation. Moving data into memory. String duplication.
2	Structured programming and language C. Aggregate types. Structures. Bit fields. Union. Definition of synonyms for types (typedef).
2	Structured programming and C language. File. Main file access operations.
2	Structured programming and language C. Preprocessor. Header file. Conditional compilation. The standard library. Mathematical functions.
2	Structured programming and C language. Functions pointers.