UNIVERSITÀ DEGLI STUDI DI PALERMO

DEPARTMENT	Matematica e Informatica		
ACADEMIC YEAR	2017/2018		
BACHELOR'S DEGREE (BSC)	MATHEMATICS		
INTEGRATED COURSE	PROGRAMMING WITH LABORATORY		
CODE	10664		
MODULES	Yes		
NUMBER OF MODULES	2		
SCIENTIFIC SECTOR(S)	INF/01		
HEAD PROFESSOR(S)	TEGOLO DOMENICO	Professore Associato	Univ. di PALERMO
OTHER PROFESSOR(S)	TEGOLO DOMENICO EPIFANIO CHIARA	Professore Associato Ricercatore	Univ. di PALERMO Univ. di PALERMO
CREDITS	12		
PROPAEDEUTICAL SUBJECTS			
MUTUALIZATION			
YEAR	1		
TERM (SEMESTER)	Annual		
ATTENDANCE	Not mandatory		
EVALUATION	Out of 30		
TEACHER OFFICE HOURS	**EPIFANIO CHIARA** Tuesday 14:30 17:00 Dipartimento di Matematica ed Informatica, via Archirafi 34, Room 104, primo piano/ first floor **TEGOLO DOMENICO** Wednesday 15:00 17:00 Dipartimento Matematica e InformaticaVia Archirafi 3490123Palermo		

DOCENTE: Prof. DOMENICO TEGOLO

PREREQUISITES	No prerequisite is required
LEARNING OUTCOMES	Conoscenza e capacita' di comprensione: Acquisizione dei fondamenti sui sistemi di elaborazione e dei principi della programmazione strutturata; Apprendere i concetti di base sulle strutture dati statiche e dinamiche; Comprendere semplici algoritmi fondamentali sull'ordinamento o sulla ricerca alfanumerica; Assimilare i concetti sulla soluzione di semplici problemi attraverso la programmazione ricorsiva; possesso delle competenze sui costrutti fondamentali del linguaggio di programmazione C. Capacita' di applicare conoscenza e comprensione: Comprendere i processi cognitivi utili per l'individuazione di una soluzione ad un problema numerico semplice o complesso. Capacita' di programmazione nel linguaggio di programmazione C. Capacita' di individuare e risolvere errori sintattici e semantici emersi in fase di compilazione i primi ed esecuzione i secondi. Essere in grado di suddividere problemi complessi in problemi elementari. Autonomia di giudizio: Essere in grado di valutare la bonta' di metodi e i contenuti nella redazione di un programma. Ampia comprensione dei concetti avanzati sull'uso dei costrutti e delle strutture dati. Essere in grado di valutare le implicazioni sull'uso dei costrutti e sul passaggio dei parametri alle funzioni. Il raggiungimento di un'autonomia di giudizio sugli applicativi sviluppati basati su di una soluzione algoritmica efficiente. Abilita' comunicative: Proprieta' di espressione nella presentazione delle nozioni di base dell'arte della programmazione e del linguaggio di programmazione C. Capacita' d'apprendimento: Sapere approcciarsi alla programmazione, al problem solving e a sapere contestualizzare le abilita' acquisite in problemi concreti.
ASSESSMENT METHODS	Il voto finale terra' conto del voto delle prove in itinere e di una prova finale o in alternativa terra' conto del voto ottenuto nella prova integrata finale costituita da una prova scritta e da un colloquio orale. Le prove scritte saranno costituite da 10 tra domande a risposte multiple o aperte, e da 2 esercizi di programmazione. Le domande saranno valutate con un punteggio da 0 a 2, mentre ogni esercizio avra' un punteggio massimo di 5, la prova scritta sara' considerata superata con un punteggio maggiore o uguale a 15/30. La prova orale, valutata in trentesimi, sara' costituita dalla discussione delle prove scritte (prove in itinere o prova scritta finale), e da domande al fine di accertare la conoscenza degli argomenti del corso. Il voto finale terra' conto del voto riportato nelle prove in itinere o nella prova finale scritta ed orale. Pertanto si ribadisce, che sono previste delle prove scritte in itinere secondo il calendario didattico la cui natura e' del tutto simili alle prove scritte delle sessioni d'esame, il superamento della prova in itinere comporta, su richiesta dello studente, l'esonero (parziale o totale) della prova scritta finale. La valutazione finale, sara' formulata sulla base delle seguenti condizioni: a) Conoscenza molto buona, ampia ed elevata degli argomenti proposti; capacita' di applicarli con rigore e in piena autonomia; possesso di ottime capacita' comunicative (voto 29-30L). b) Buona conoscenza degli argomenti proposti e capacita' di applicarli con rigore matematico e metodologico, ma non in piena autonomia; possesso di buona proprieta' di linguaggio (voto 26-28); c) Conoscenza discreta degli argomenti proposti e sufficiente capacita' di applicarli autonomamente; discreta capacita' di portare a termine un ragionamento rigoroso e buona proprieta' di linguaggio (voto 22-25); d) Conoscenza di base degli argomenti proposti e capacita' limitata di applicarli autonomamente; sufficiente capacita' di portare a termine un ragionamento rigoroso e sufficiente proprieta' di linguaggio (voto 18-21); e) Conoscenza insufficiente: lo studente non possiede una conoscenza accettabile dei contenuti e degli argomenti trattati nell'insegnamento e non ha alcuna capacita' di applicare autonomamente le conoscenze acquisite.
TEACHING METHODS	The achievement of the teaching objectives will be achieved through frontal lessons and laboratory experiences.

MODULE
STRUCTURED PROGRAMMING

Prof. DOMENICO TEGOLO

SUGGESTED BIBLIOGRAPHY

Paul J. Deitel - Harvey M. Deitel, Il linguaggio C - Fondamenti e tecniche di programmazione • 8/Ed.
A. Bellini, A.Guidi. Linguaggio C - guida alla programmazione. Mc Graw Hill.

AMBIT	50194-Formazione informatica
INDIVIDUAL STUDY (Hrs)	78
COURSE ACTIVITY (Hrs)	72

EDUCATIONAL OBJECTIVES OF THE MODULE

The module aims to provide theoretical and workroom methodologies aimed at acquiring the basic concepts for programming in computer system environment. Both the representation of data and the elementary constructs of an imperative programming language will be analyzed, and simple algorithms will be used, which will use control, sequencing, selection and iteration structures.
Due to propaedeutic and considerable diffusion in the market, and in order to guarantee more versatility in the world of work, the programming language C will be taken into account.

SYLLABUS

Hrs	Frontal teaching
2	Introduction to the module, organization of a computer, hardware evolution, evolution of operating systems, evolution of programming languages.
4	Computer: Hardware and software. The binary system: definition, operations. Switching from decimal to binary and vice versa. Information and measurements: bit and byte. Introduction to C development environments, introduction to programming. Definition of Algorithm. Simple meta-language programs. Understand the complexity of an algorithm.
4	Introduction to different programming paradigms: imperative paradigm (structured programming and object programming), declarative paradigm (functional programming and logic programming). Structured programming. The Böhm-Jacopini theorem. The C language and the structures of a program. The Sequence construct, If ... else, switch..case constructor. Identifiers. Input / output functions.
2	Constants and variables, instruction of assignment. Standard data types: Integer, character and their representation, float and double types. ASCII code and other character codes. Representing real numbers in memory. Operators in C and their priority. Incremental and decreasing operators of a integer variable.
4	Loop control structures: The loop control FOR, the loop control WHILE, the loop control WHILE ...DO. Equivalence among loop controls.
2	Array in C. Array unidimensional and applications. Definition and display of arrays. N-dimensional arrays: matrices. Definition of assignment and display of a matrix.
2	Linear Search, Iterative Binary Search. Sorting Algorithms. Strings and Function Library on strings <string.h>.
2	Functions in C: Declaring, Defining and Calling Functions. How to send parameters to a function and visibility of variables.
2	Pointers. Array and Pointers. Arithmetic of pointers. Loot functions on arrays. Recursion. Recursive functions and algorithms.

Hrs	Workshops
4	Laboratory with exercise on assignment, increase and decrease operators. Basic Input / Output Instructions
4	Laboratory with arithmetic exercise on Integer and Real variables. Equal and Relational Operators in C.
4	Laboratory with exercise on selection constructions with and without nesting.
4	Laboratory with exercise on iterative construct FOR
4	Laboratory with exercise on iterative construct WHILE
4	Laboratory with exercise on iterative construct DO..WHILE
4	Laboratory with tutorial on multiple selection construction SWITCH and BREAK-CONTINUE.
4	Laboratory with tutorial on logical operators and the equivalence of the iterative cycles.
4	Laboratory with tutorial about standard and user-defined function: prototypes of a function and parameters passage. Rules of visibility.
4	Laboratory with tutorial on arrays: search and sorting.
4	Laboratory and tutorial on pointers, expressions and arithmetic. Relationship between pointer and array.
4	Laboratory and tutorial on recursion programming; an example of algorithm: the Hanoi Towers.

MODULE
ADVANCED PROGRAMMING

Prof.ssa CHIARA EPIFANIO

SUGGESTED BIBLIOGRAPHY

- C. Demetrescu, I. Finocchi, G.F. Italiano, Algoritmi e strutture dati, McGraw-Hill.

- A. Bellini, A.Guidi. Linguaggio C - guida alla programmazione. Mc Graw Hill.

AMBIT	10709-Attività formative affini o integrative
INDIVIDUAL STUDY (Hrs)	94
COURSE ACTIVITY (Hrs)	56

EDUCATIONAL OBJECTIVES OF THE MODULE

This part of the course deepens some advanced topics concerning programming. In particular we analyze some dynamic data structures defined with the aid of pointers and we will study the sorting problem. Moreover, we study how to manage files.

SYLLABUS

Hrs	Frontal teaching
3	Two models based on pointer lists: Stack and Queue. Operations on stacks: push and pop; operations on queues: enqueue and dequeue.
6	Trees. Binary trees. Preorder, inorder, postorder recursive traversals. Breadth first traversal of a tree. Iterative version of the preorder tree traversal with the aid of a stack.
5	Binary expression trees. Connection between tree trasversals and prefix, infix and postfix algebraic expressions.
6	Binary search trees (BST). Construction. Searching, Insertion and deletion of an element in a BST. Complexity of Dictionary operations in BSTs.
3	Files management.
5	Graphs. Graphs implementation: adjacency matrix and adjacency array. Operations on graphs and their complexity. Breadth first search (BFS). Depth first search (DFS). Acyclicity of a graph.
4	Sorting. Problem definition. Lower bound theorem for sorting algorithms based on comparisons. Iterative algorithms: SelectionSort, InsertionSort, BubbleSort. Recursive algorithms: MergeSort, QuickSort. Bucketsort.

Hrs	Practice
2	Exercices on stack and queue, operations on stacks: push and pop, operations on queues: enqueue and dequeue.
4	Exercises on trees, binary trees, preorder, inorder, postorder recursive traversals, breadth first traversal of a tree, iterative version of the preorder tree traversal with the aid of a stack.
5	Exercises on binary expression trees, connection between tree trasversals and prefix, infix and postfix algebraic expressions.
2	Exercises on binary search trees (BST), construction, searching, insertion and deletion of an element in a BST, complexity of Dictionary operations in BSTs.
3	Exercises on files management
4	Exercises on graphs, graphs implementation: adjacency matrix and adjacency array, operations on graphs and their complexity, breadth first search (BFS), depth first search (DFS), acyclicity of a graph.
4	Exercices on sorting, lower bound theorem for sorting algorithms based on comparisons, iterative algorithms: SelectionSort, InsertionSort, BubbleSort, recursive algorithms: MergeSort, QuickSort. Bucketsort.