Principali informazioni sull'insegnamento	
Denominazione dell'insegnamento	**Laboratorio di Informatica**
Corso di studio	Informatica e Tecnologie dell'Informazione
Anno accademico	2025/26
Crediti formativi universitari (CFU) / European credit transfer and accumulation system (ECTS)	6
Settore scientifico disciplinare	IINF-05/A
Lingua di erogazione	Italiano
Anno di corso	Primo
Periodo di erogazione	Secondo semestre
Obbligo di frequenza	La frequenza è fortemente raccomandata
Sito web del corso di studio	https://www.uniba.it/it/corsi/informatica-tecnologie-informazione-taranto/

Docente	
Nome	Gennaro Vessio
E-mail	gennaro.vessio@uniba.it
Sede dell'ufficio	Dipartimento di Informatica, Via Orabona 4, 70125 - Bari, stanza n. 673
Sede virtuale	https://elearning.uniba.it/
Sito web del docente	https://www.gennarovessio.com/
Ricevimento	Su appuntamento

Syllabus	
Obiettivi formativi	L'insegnamento di Laboratorio di Informatica ha l'obiettivo di fornire agli studenti le competenze fondamentali per la progettazione, lo sviluppo e la documentazione di soluzioni algoritmiche a problemi reali. Il corso pone particolare attenzione agli aspetti pragmatici della programmazione, con enfasi sulla progettazione modulare, sullo stile di codifica, sulla scelta accurata della nomenclatura, sul testing e sul

	debugging. Attraverso lo sviluppo guidato di un caso di studio su un problema reale, gli studenti imparano a realizzare soluzioni corrette, leggibili, ben progettate e documentate, consolidando al tempo stesso le proprie capacità di problem solving e di comunicazione tecnica.
Prerequisiti	Nessuno
Contenuti di insegnamento	1. Il linguaggio C 2. Funzioni, array e stringhe 3. Puntatori e gestione dinamica della memoria 4. Strutture e strutture dati dinamiche 5. File 6. Progettazione modulare 7. Stile di programmazione 8. Complessità computazionale (cenni) 9. Documentazione del codice 10. Testing 11. Debugging
Testo suggerito	Il corso non adotta un testo di riferimento; tuttavia, i contenuti didattici possono essere integrati dal seguente testo suggerito: H. M. Deitel, P. J. Deitel, "Il linguaggio C. Fondamenti e tecniche di programmazione", Pearson, 2022 Gli studenti che lo desiderano possono prendere in prestito il testo dalla biblioteca. Può essere conveniente verificarne la disponibilità mediante il Sistema Bibliotecario di Ateneo (https://opac.uniba.it/easyweb/w8018/index.php) e contattare la biblioteca per concordare il prestito.
Note aggiuntive	I contenuti didattici saranno trasferiti principalmente attraverso: dispense del docente; risoluzioni di esercitazioni; esempi di casi di studio del passato.

Organizzazione della didattica

Ore

Totali	Didattica frontale	Pratica (esercitazioni, caso di studio)	Studio individuale
150	24	45	81
CFU/ECTS			
6	3	3	–

Metodi didattici

Lezioni frontali, esercitazioni in linguaggio C, caso di studio in coppia.

Risultati di apprendimento previsti	
Conoscenza e capacità di comprensione	Al termine dell'insegnamento, i discenti conosceranno alcuni dei principali aspetti pragmatici della programmazione, quali la programmazione difensiva, un corretto stile di scrittura, l'uso appropriato della nomenclatura, testing e debugging, progettazione modulare, ecc.
Conoscenza e capacità di comprensione applicate	Al termine dell'insegnamento, posti di fronte a un problema reale, i discenti saranno in grado di implementare una soluzione non solo corretta, ma anche chiara, ben progettata e ben documentata. Inoltre, avranno ampliato le proprie capacità di problem solving.
Competenze trasversali	Al termine dell'insegnamento, i discenti avranno sviluppato la capacità di condurre un caso di studio reale dall'analisi dei requisiti, all'implementazione di una soluzione algoritmica e al suo testing e debugging. I discenti apprenderanno inoltre a redigere una documentazione di progetto e a comunicarla con efficacia a interlocutori specialisti e non, enfatizzandone i punti chiave.

Valutazione	
Modalità di verifica dell'apprendimento	L'esame prevede un caso di studio—pensato per essere svolto in coppia—su un problema reale complesso scelto da un insieme di possibili tracce. Il caso di studio, discusso in sede di orale, e corredato da una documentazione di progetto e dal codice e i dati a esso associati, dovrà dimostrare la capacità di produrre codice corretto, leggibile, ben documentato, e correttamente progettato e strutturato. Poiché l'insegnamento è di natura fortemente pratica, non prevede esoneri; tuttavia, i casi di studio potranno iniziare a essere svolti già a metà del programma per consentire agli studenti di sostenere l'esame già al primo appello. Il caso di studio svolto viene mantenuto per tutto l'anno accademico.
Criteri di valutazione	Ci si aspetta che i discenti abbiano appreso a: scrivere programmi che garantiscano un'adeguata correttezza e aderenza ai requisiti di progetto; applicare metodologie di progettazione modulare per ottimizzare la struttura del codice sorgente; scrivere correttamente una documentazione di progetto; scrivere codice sorgente che risponda ai dettami del corretto stile di programmazione, garantendo leggibilità e semplicità; applicare correttamente metodologie di testing del codice sorgente e strumenti di debugging; esporre efficacemente l'elaborato.
Criteri di misurazione dell'apprendimento e di attribuzione del voto finale	Il voto finale sarà espresso in trentesimi e l'esame si riterrà superato con un voto conseguito maggiore o uguale a 18. Il maggior contributo al voto finale verrà dalla consegna di un programma funzionante, una documentazione chiara ed esaustiva, e codice ben scritto e strutturato. Parte della valutazione riguarderà l'uso di quanto non espressamente richiesto dalla traccia, come i puntatori o la gestione dinamica della memoria. Premialità saranno infine riconosciute a gruppi che avranno arricchito il caso di studio con estensioni non trattate durante il corso.
Informazioni aggiuntive	Si raccomanda vivamente ai discenti di impegnarsi a svolgere le esercitazioni (da soli o in gruppo) senza attenderne la risoluzione, al fine di sviluppare capacità di analisi e giudizio critico. Si suggerisce altresì agli studenti di affidarsi esclusivamente alle informazioni/comunicazioni fornite sui siti ufficiali del Dipartimento di

	Informatica, o sui gruppi social costituiti e amministrati esclusivamente dai docenti degli insegnamenti di interesse: • https://www.uniba.it/it/ricerca/dipartimenti/informatica • https://elearning.uniba.it/ Le informazioni che tutti gli studenti dovrebbero conoscere sono riportate nei regolamenti didattici e nei manifesti degli studi disponibili sul sito: https://www.uniba.it/it/corsi/corsi-di-laurea Si suggerisce agli studenti di diffidare delle informazioni che circolano su siti o gruppi social non ufficiali, poiché sono risultate spesso inaffidabili, errate o incomplete. Gli studenti saranno invitati a unirsi a un gruppo Telegram che sarà creato all'inizio delle lezioni.

Main course information

Title	**Computer Science Laboratory**
Degree	Computer Science and Information Technologies
Academic year	2025/26
European credit transfer and accumulation system (ECTS) / Crediti formativi universitari (CFU)	6
Scientific disciplinary sector	IINF-05/A
Language	Italian
Course year	First
Delivery period	Second semester
Required attendance	Attendance is strongly recommended
Website	https://www.uniba.it/it/corsi/informatica-tecnologie-informazione-taranto/

Lecturer

Name	Gennaro Vessio
E-mail	gennaro.vessio@uniba.it
Office location	Department of Computer Science, Via Orabona 4, 70125 Bari, room 673
Virtual platform	https://elearning.uniba.it/
Personal website	https://www.gennarovessio.com/
Office hours	By appointment

Syllabus

Learning objectives	The Computer Science Laboratory course aims to provide students with the fundamental skills required for designing, developing, and documenting algorithmic solutions to real-world problems. The course places particular emphasis on the practical aspects of programming, focusing on modular design, coding style, proper naming conventions, testing, and debugging. Through the guided

	development of a case study based on a real-world problem, students will learn how to produce solutions that are correct, readable, well-designed, and well-documented, while also improving their problem-solving abilities and technical communication skills.
Prerequisites	None
Course content	1. The C programming language 2. Functions, arrays, and strings 3. Pointers and dynamic memory management 4. Structures and dynamic data structures 5. File handling 6. Modular design 7. Coding style 8. Computational complexity (introduction) 9. Code documentation 10. Testing 11. Debugging
Recommended textbook	The course does not adopt a mandatory textbook. However, the following reference book is suggested to complement the teaching material: H. M. Deitel, P. J. Deitel, "Il linguaggio C. Fondamenti e tecniche di programmazione", Pearson, 2022 Students who wish to consult this book may borrow it from the university library. Availability can be checked through the University Library System (https://opac.uniba.it/easyweb/w8018/index.php), and students are encouraged to contact the library to arrange a loan.
Additional notes	Teaching materials will primarily consist of instructor-provided handouts, solutions to exercises, and examples of past case studies.

Teaching organization

Hours

Total	Lectures	Practice (exercises, case study)	Independent study
150	24	45	81
ECTS/CFU			
6	3	3	–

Teaching methods

Lectures, programming exercises in the C language, and the development of a case study in pairs.

Expected learning outcomes	
Knowledge and understanding	At the end of the course, students will be familiar with the main practical aspects of programming, such as defensive programming, proper coding style, effective naming conventions, testing, debugging, and modular design.
Applied knowledge and understanding	Upon completion of the course, students will be able to implement solutions to real-world problems that are not only correct, but also clear, well-structured, and properly documented. In addition, they will enhance their problem-solving abilities and learn to work effectively on practical programming tasks.
Other skills	By the end of the course, students will have developed the ability to manage an entire real-world case study, from requirement analysis to the implementation of an algorithmic solution, as well as its testing and debugging. They will also learn to prepare effective project documentation and to communicate technical results clearly to both specialist and non-specialist audiences.

Assessment	
Verification modality	The final exam consists of a case study, designed to be carried out in pairs, based on a complex real-world problem selected from a set of possible scenarios. The project must be submitted together with complete project documentation, source code, and related data, and will be discussed during an oral examination. Since the course is highly practical, no partial exams are planned. However, students are allowed to begin working on their case studies halfway through the course, enabling them to take the exam as early as the first available session. Once completed, the case study remains valid for the entire academic year.
Assessment criteria	Students are expected to demonstrate their ability to: write programs that ensure correctness and adhere to project requirements; apply modular design principles to improve code structure and maintainability; produce clear, accurate, and well-organized documentation; write clean, readable, and efficient code following proper programming practices; apply testing methodologies and use debugging tools effectively; present and explain their work clearly and effectively.
Criteria and final grade	The final grade will be expressed on a scale of thirty points, and the exam will be considered passed with a score of at least 18. The assessment will primarily focus on the delivery of a fully functioning program, supported by clear, accurate, and comprehensive documentation, as well as well-structured and readable code. Additional consideration will be given to solutions that demonstrate the use of advanced concepts not explicitly required by the assignment, such as pointers or dynamic memory management. Further credit may also be awarded to groups that enhance their case study with meaningful extensions or innovative features beyond the topics covered during the course.
Additional information	Students are strongly encouraged to complete all programming exercises—individually or in groups—without waiting for official solutions, in order to strengthen their analytical skills and critical thinking. Students are advised to rely exclusively on information and announcements provided through the official Department of Computer Science website or the e-learning platform: • https://www.uniba.it/it/ricerca/dipartimenti/informatica

	• https://elearning.uniba.it/ Official academic regulations and program guidelines can be found at: https://www.uniba.it/it/corsi/corsi-di-laurea Students are strongly advised not to rely on unofficial websites or social groups, as the information they provide is often incomplete, inaccurate, or misleading. At the beginning of the course, students will be invited to join an official Telegram group managed by the instructor.