Principali informazioni sull'insegnamento	
Denominazione dell'insegnamento	**LABORATORIO DI INFORMATICA (TRACK MARINA MILITARE)**
Corso di studio	INFORMATICA E COMUNICAZIONE DIGITALE (MARINA MILITARE)
Anno Accademico	2024/25
Crediti formativi universitari (CFU) / European Credit Transfer and Accumulation System (ECTS)	6 CFU
Settore Scientifico Disciplinare	INF/01
Lingua di erogazione	Italiano
Anno di corso	Primo
Periodo di erogazione	2^ semestre
Obbligo di frequenza	La frequenza è fortemente raccomandata
Sito web del corso di studio	https://www.uniba.it/it/ricerca/dipartimenti/informatica/didattica/corsi-di-laurea/informatica-icd-taranto-270/laurea-triennale-in-informatica-e-comunicazione-digitale-sede-di-taranto-d.m.-270

Docente/i	
Nome e cognome	ALESSANDRO PAGANO
Indirizzo mail	alessandro.pagano@uniba.it
Telefono	080 5715200
Sede	Dipartimento di Informatica, Via Orabona 4, 70125, Bari. Stanza n. 773, 7^ piano.
Sede virtuale	Piattaforma e-learning - https://elearning.uniba.it/
Sito web del docente	https://www.uniba.it/it/docenti/pagano-alessandro
Ricevimento (giorni, orari e modalità, es. su appuntamento)	Mercoledì 14.10 – 15.30 - Previo appuntamento via mail o MsTeams.

Syllabus	
Obiettivi formativi	Il corso si propone di far acquisire le conoscenze necessarie per progettare e realizzare unità strutturali che siano composte da algoritmi, strutture dati e interfacce attraverso cui queste componenti comunicano con l'utente. Si introdurranno gli elementi della programmazione imperativa strutturata per rendere in grado gli studenti di produrre soluzioni algoritmiche a problemi di complessità avanzata. In particolare, lo studente acquisirà la capacità di usare il linguaggio di programmazione C come strumento per modellare problemi e formalizzarne le soluzioni.
Prerequisiti	Le seguenti conoscenze preliminari facilitano ed accelerano la comprensione degli argomenti dell'insegnamento: • da Programmazione: basi della programmazione imperativa, strutture dati e dati strutturati, problem-solving, tecniche di rappresentazione di una soluzione, Programmazione Strutturata; • da Architetture degli Elaboratori e SO: tipologia di dati e occupazione della memoria, algebra di Boole e porte logiche, gestione dei processi.
Contenuti di insegnamento (Programma)	**Ripasso dei principali concetti di programmazione e dei costrutti in C (10 ore)** • Dal problem solving al programma • Costrutti di base della programmazione strutturata • Cenni sulla memoria e l'occupazione dei dati • Dati strutturati e Strutture Dati: definizione e gestione • Prime esercitazioni: soluzioni di problemi **IDE per la creazione di progetti (3 ore)** • Introduzione alle funzionalità avanzate degli IDE per la gestione dei progetti e dei casi di studio • Introduzione a piattaforme per le esercitazioni online **Programmazione avanzata (25 ore)** • Elaborazione avanzata di stringhe, funzioni per la gestione delle stringhe • Puntatori concetto astratto e utilizzo concreto nel linguaggio C • Array di puntatori e aritmetica dei puntatori • Record gestione e manipolazione • File di testo e file binari: gestione e manipolazione • Esercitazioni: soluzione di problemi complessi **Stile di Programmazione (3 ore)** • Uso appropriato dei nomi • Scrittura appropriata di espressioni e istruzioni • Consistenza ed espressioni idiomatiche • Commenti

	• Convenzioni di programmazione • Esercitazioni: modificare e migliorare i programmi realizzati **Programmazione modulare (10 ore)** • Modularizzazione e strutturazione dei programmi. • Astrazione dati e funzionale • Information Hiding • Classi di memoria • Scope degli identificatori • Header file • Esercitazioni: creazione di programmi modulari **Programmazione Difensiva (2 ore)** • Tecniche per limitare i malfunzionamenti nei programmi **Testing e Debugging (3 ore)** • Metodologie di testing: white box e black box • Tecniche di testing • Strumenti e tecniche di debugging **Algoritmi Fondamentali (13 ore)** • Algoritmi di Ricerca (esaustiva, con sentinella, binaria) • Algoritmi di Ordinamento (bubble sort, selection sort, shell sort, insertion sort, merge sort, quick sort) • Complessità Computazionale (cenni)
Testi di riferimento	**Testo da cui studiare:** P. Deitel e H. Deitel Il linguaggio C – Fondamenti e tecniche di programmazione 8^edizione - Pearson 2016 - ISBN: 9788891901651 (vanno bene anche le edizioni successive e precedenti dalla 4^ in poi) **Testo integrativo, facoltativo:** W. B. Kernighan, D.M. Ritchie. Il linguaggio C. Principi di programmazione e manuale di riferimento. Gli studenti che lo desiderano possono ottenere i testi in prestito dalla Biblioteca. Può convenire 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 ai testi di riferimento	Nel corso delle lezioni il docente illustrerà i concetti con l'ausilio di slide che sintetizzano i contenuti del corso. Le slide saranno rese disponibili al termine di ogni lezione sulla piattaforma e-learning di Uniba (v. sopra 'sede virtuale'). Sulla piattaforma sono disponibili, inoltre, tutte le indicazioni per realizzare il caso di studio, oltre ad una traccia di esempio e al format per la realizzazione del documento di progettazione.
Organizzazione della didattica	

Ore			
Totali	Didattica frontale	Laboratorio	Studio individuale
150 ore	24 ore	45 ore	81 ore
CFU/ETCS			

6 CFU	3 CFU	3 CFU	

Metodi didattici	Il corso è organizzato in lezioni frontali svolte con l'ausilio di slide, in esercitazioni guidate svolte in aula e esercitazioni svolte autonomamente a casa. Le esercitazioni guidate saranno svolte in aula con l'approccio del Bring Your Own Device (BYOD). Le esercitazioni svolte in aula e a casa dovranno essere svolte singolarmente da ciascuno studente. Sarà richiesta la consegna delle stesse sulla piattaforma di e-learning o la piattaforma online per la codifica seguendo le indicazioni e le scadenze comunicate durante le lezioni frontali. Le esercitazioni consegnate saranno utili al docente per verificare la partecipazione alle esercitazioni e la comprensione degli argomenti svolti a lezione. **Agli studenti non frequentanti non è richiesta la sottomissione delle esercitazioni.**

Risultati di apprendimento previsti	
Conoscenza e capacità di comprensione	• Acquisizione conoscenze approfondite del linguaggio di programmazione C. • Acquisire capacità di progettazione e sviluppo di programmi di complessità elevata. • Migliorare le capacità di problem-solving anche per problemi complessi.
Conoscenza e capacità di comprensione applicate	• Saper tradurre algoritmi in programmi (in C) correttamente funzionanti e ben documentati. • Saper utilizzare tecniche di programmazione difensiva, per limitare l'introduzione di malfunzionamenti nei programmi. • Saper condurre una verifica empirica della correttezza dei programmi mediante tecniche di testing. • Saper realizzare soluzioni più efficienti.
Competenze trasversali	**Autonomia di giudizio** • Saper verificare l'aderenza di un programma alle specificità del problema da risolvere • Saper valutare l'efficienza dei programmi sviluppati. **Abilità comunicative** • Saper illustrare in modo appropriato le caratteristiche tecniche degli strumenti e delle metodologie informatiche relative allo sviluppo di programmi. **Capacità di apprendere in modo autonomo** • Essere in grado di orientarsi agevolmente nelle problematiche relative alla comprensione e all'utilizzo delle tecnologie e dei metodi per lo sviluppo di programmi, nonché ai diversi linguaggi di programmazione

Valutazione	
Modalità di verifica dell'apprendimento	**Valutazione Sommativa (Esame):** **La prova di valutazione finale si tiene in modalità scritta** • **Prima Prova**. La prima prova consiste nella soluzione di un problema complesso o parti di esso individuando un algoritmo e la relativa codifica in C. Problemi e modalità di svolgimento della prova saranno in linea con quanto illustrato durante le lezioni. L'obiettivo della prova è valutare l'acquisizione di competenze e abilità di problem-solving e programmazione. • **Seconda prova**. La seconda prova consiste in un test a risposta multipla chiusa e/o domande aperte con l'obiettivo di valutare l'acquisizione delle conoscenze teoriche di programmazione. La prova durerà orientativamente 30 minuti. Sarà valutata la possibilità di tenere le due prove nella stessa data. Entrambe le prove saranno valutate in 30simi e il voto finale sarà costituito dalla media dei voti ottenuti nelle due prove. Il voto finale si ottiene come media aritmetica dei voti ottenuti alle prove proposte. La media aritmetica è arrotondata per difetto o per eccesso in base ad una valutazione del docente relative alla qualità delle prove sostenute. Tutte le valutazioni saranno comunicate utilizzando la piattaforma di e-learning e/o email inviate tramite Esse3. Per partecipare all'esame è necessario essersi prenotati all'esame orale tramite Esse3.
Criteri di valutazione	**Conoscenza e capacità di comprensione:** • Lo studente dovrà dimostrare di aver acquisito conoscenze approfondite del linguaggio di programmazione C • Lo studente dovrà dimostrare di aver acquisito capacità di progettazione e sviluppo di programmi di complessità elevata. • Lo studente dovrà dimostrare di avere capacità di problem-solving anche per problemi complessi. **Conoscenza e capacità di comprensione applicate:** • Lo studente dovrà dimostrare di saper produrre programmi (in C) funzionanti, saperli documentare. • Lo studente dovrà dimostrare di saper utilizzare tecniche di programmazione difensiva e di condurre una verifica empirica della correttezza dei programmi mediante tecniche di testing. • Lo studente dovrà dimostrare di saper applicare realizzare soluzioni più efficienti **Autonomia di giudizio:** • Abilità di valutazione della qualità di un approccio risolutivo adottato o adottabile per definire la soluzione ad un problema **Abilità comunicative:**

<table>
<tr><td></td><td>

- Lo studente dovrà dimostrare di saper illustrare in modo appropriato la soluzione creata utilizzando un linguaggio tecnico corretto.

Capacità di apprendere:

- Capacità di astrazione, di ragionamento per analogia e dimostrazione di creatività nella risoluzione dei problemi

</td></tr>
<tr><td>Criteri di misurazione dell'apprendimento e di attribuzione del voto finale</td><td>

<table>
<tr><th>Voto</th><th>Descrittori</th></tr>
<tr><td>< 18
insufficiente</td><td>Conoscenze frammentarie e superficiali dei contenuti, errori nell’applicare i concetti, descrizione carente.</td></tr>
<tr><td>18 - 20</td><td>Conoscenze dei contenuti sufficienti ma generali, descrizione semplice, incertezze nell’applicazione di concetti teorici.</td></tr>
<tr><td>21 - 23</td><td>Conoscenze dei contenuti appropriate ma non approfondite, capacità di applicare i concetti teorici, capacità di presentare i contenuti in modo semplice.</td></tr>
<tr><td>24 - 25</td><td>Conoscenze dei contenuti appropriate ed ampie, discreta capacità di applicazione delle conoscenze, capacità di presentare i contenuti in modo articolato.</td></tr>
<tr><td>26 - 27</td><td>Conoscenze dei contenuti precise e complete, buona capacità di applicare le conoscenze, capacità di analisi, descrizione chiara e corretta.</td></tr>
<tr><td>28 - 29</td><td>Conoscenze dei contenuti ampie, complete ed approfondite, buona applicazione dei contenuti, buona capacità di analisi e di sintesi, descrizione sicura e corretta.</td></tr>
<tr><td>30
30 e lode</td><td>Conoscenze dei contenuti molto ampie, complete ed approfondite, capacità ben consolidata di applicare i contenuti, ottima capacità di analisi, di sintesi e di collegamenti interdisciplinari, padronanza di descrizione.</td></tr>
</table>

</td></tr>
<tr><td>Altro</td><td>

Si suggerisce agli studenti di affidarsi esclusivamente alle informazioni/comunicazioni fornite sui siti ufficiali del Dipartimento di Informatica, ovvero sui gruppi social solo se costituiti e amministrati esclusivamente dai docenti dei relativi insegnamenti:

- https://www.uniba.it/it/ricerca/dipartimenti/informatica/didattica/corsi-di-laurea/corsi-di-laurea
- https://www.uniba.it/it/ricerca/dipartimenti/informatica
- https://elearning.uniba.it/

I programmi degli insegnamenti sono disponibili qui:

- https://elearning.uniba.it/

Le informazioni che tutti gli studenti dovrebbero conoscere sono scritte nei Regolamenti didattici e manifesti degli studi disponibili nel sito:

- https://www.uniba.it/it/ricerca/dipartimenti/informatica/didattica/corsi-di-laurea/corsi-di-laurea

Si suggerisce agli studenti di diffidare delle informazioni e dei materiali circolanti su siti o gruppi social non ufficiali, poiché spesso sono risultati non affidabili, non corretti o incompleti. Per ogni dubbio, chiedere un incontro al docente secondo le modalità previste per il ricevimento.

</td></tr>
</table>

Main information on the course

Course name	Computer Science Laboratory (Track surnames MZ)	
Degree	Computer Science	
Academic year	2023/24	
European Credit Transfer and Accumulation System (ECTS), in Italian Crediti Formativi Universitari (CFU)		6 CFU (each CFU corresponds to 25 hours (h) of the student's time); CFU are of type T1, T2 or T3 T1 = 8 h lecture + 17 h individual study T2 = 15 h practice + 10 h individual study T3 = 25 h individual study
Settore Scientifico Disciplinare	INF/01	
Course language	Italian	
Anno di corso	First	
Periodo di erogazione	2nd semester	
Obbligo di frequenza	It is highly recommended to attend classes	
Sito web del corso di studio	https://www.uniba.it/it/ricerca/dipartimenti/informatica/didattica/corsi-di-laurea/informatica-icd-taranto-270/laurea-triennale-in-informatica-e-comunicazione-digitale-sede-di-taranto-d.m.-270	

Teacher(s)

Name and Surname	Alessandro Pagano
email	Alessandro.pagano@uniba.it
phone	+39 080 5715200
office	Dipartimento di Informatica, Via Orabona 4, 70125, Bari. Stanza n. 773, 7^ floor.
e-learning platform	e-learning platform- https://elearning.uniba.it/
Teacher's homepage	https://www.uniba.it/it/docenti/pagano-alessandro
Office hours	Thursday 14:10 -15:30 (by appointment to be agreed by email with the lecturer)

Syllabus

Course goals	The course aims to provide students with the necessary knowledge to design and implement structural units that are composed of algorithms, data structures and interfaces through which these components communicate with the user. The elements of structured imperative programming will be introduced to enable students to produce algorithmic solutions to problems of advanced complexity. In particular, the student will acquire the ability to use the C programming language as a tool to model problems and formalise their solutions.
Prerequisites/requirements	The following prior knowledge facilitates and accelerates understanding of the teaching topics:

	• from Programming: basics of imperative programming, data structures and structured data, problem-solving, techniques for representing a solution, Structured Programming; • from Computer Architectures and OS: data types and memory occupancy, Boolean algebra and logic gates, process management.
Course program	Review of main programming concepts and constructs in C (10 hours) - From problem solving to programming - Basic constructs of structured programming - Hints on memory and data occupation - Structured data and Data Structures: definition and management - First exercises: problem solving IDE for creating projects (3 hours) - Introduction to the advanced features of IDEs for managing projects and case studies - Introduction to online tutorial platforms Advanced programming (25 hours) - Advanced string processing, functions for string handling - Pointers abstract concept and concrete use in the C language - Pointer arrays and pointer arithmetic - Record management and manipulation - Text and binary files: handling and manipulation - Exercises: solving complex problems Programming style (3 hours) - Appropriate use of names - Appropriate writing of expressions and instructions - Consistency and idiomatic expressions - Comments - Programming conventions - Exercises: modifying and improving implemented programmes Modular programming (10 hours) - Modularisation and structuring of programmes - Data and functional abstraction - Information hiding - Memory classes - Identifier scopes - Header files - Exercises: Creating modular programmes Defensive Programming (2 hours) - Techniques for limiting program failures Testing and Debugging (3 hours) - Testing methodologies: white box and black box - Testing techniques - Debugging tools and techniques Fundamental Algorithms (13 hours) - Search Algorithms (exhaustive, sentinel, binary) - Sorting Algorithms (bubble sort, selection sort, shell sort, insertion sort, merge sort, quick sort) - Computational Complexity (Overview)
Books of reference	Text to study from: P. Deitel and H. Deitel

	The C language - Fundamentals and programming techniques 8th edition - Pearson 2016 - ISBN: 9788891901651 (Later and earlier editions from 4th onwards are also fine) Supplementary text, optional: W. B. Kernighan, D. M. Ritchie. The C language. Programming principles and reference manual. Students who wish to do so can obtain the texts on loan from the Library. You may wish to check their availability via the University Library System https://opac.uniba.it/easyweb/w8018/index.php? and contact the library to arrange the loan.
Notes to the books	During the lectures, the lecturer will illustrate the concepts with the aid of slides summarising the course content. The slides will be made available at the end of each lesson on Uniba's e-learning platform (see 'virtual venue' above). Also available on the platform are all the instructions for carrying out the case study, as well as an example outline and the format for producing the design document.

Organization of the didactic activities

Hours			
Total	Lectures	Practice sessions	Individual study
150 hours	24 hours	45 hours	81 hours
CFU/ETCS			
CFU 6	CFU 3	CFU 3	

Teaching methods

The course is organised into lectures with the aid of slides, guided exercises conducted in the classroom and exercises conducted independently at home.

The guided exercises will be carried out in the classroom using the Bring Your Own Device (BYOD) approach.

The exercises carried out in the classroom and at home must be carried out individually by each student. They will be required to be handed in on the ADA platform following the instructions and deadlines communicated during the lectures. The exercises handed in will be useful for the lecturer to check participation in the exercises and understanding of the topics covered in class.

Non-attending students are not required to submit the exercises.

Expected learning outcomes

Knowledge and understanding

- Acquire in-depth knowledge of the C programming language.
- Acquire skills in the design and development of highly complex programmes.
- Improve problem-solving skills even for complex problems.

Applying knowledge and understanding	• Know how to translate algorithms into correctly functioning and well-documented programs (in C). • Know how to use defensive programming techniques to limit the introduction of malfunctions into programs. • Know how to conduct empirical verification of the correctness of programs using testing techniques. • Know how to implement more efficient solutions.
Other skills	*Making judgements* Being able to verify the adherence of a programme to the specifics of the problem to be solved Being able to assess the efficiency of the programmes developed. *Communication* Being able to appropriately illustrate the technical characteristics of computer tools and methodologies related to programme development. *Learning skills* To be able to easily orientate oneself in the problems related to the understanding and use of technologies and methods for the development of programmes, as well as the various programming languages

Assessment	
Assessment methods	The final assessment test is held in written form: ● First Test. The first test consists of solving a complex problem or parts thereof by identifying an algorithm and its coding in C. Problems and the way the proof is carried out will be in line with what is explained during the lectures. The objective of the test is to assess the acquisition of problem-solving and programming skills. ● Second test. The second test consists of a closed multiple-choice test or/and open questions to assess the acquisition of theoretical programming knowledge. The test will last approximately 30 minutes. The possibility of holding the two tests on the same date will be considered. Both tests will be marked in 30s and the final mark will be the average of the marks obtained in the two tests. The final mark is obtained as the arithmetic mean of the marks obtained in the proposed tests. The arithmetic mean is rounded up or down according to the teacher's assessment of the quality of the tests taken. All assessments will be communicated using the e-learning platform and/or email sent via Esse3.

<table>
<tr><td></td><td>To participate in the examination, you must have booked the oral examination via Esse3.</td></tr>
<tr><td>Evaluation criteria</td><td>Knowledge and Understanding:

The student must demonstrate deep knowledge of the C programming language.
The student must demonstrate proficiency in designing and developing highly complex programs.
The student must exhibit problem-solving skills, even for complex problems.

Applied Knowledge and Understanding:

The student must demonstrate the ability to produce functioning programs (in C) and document them.
The student must show proficiency in using defensive programming techniques and conducting empirical verification of program correctness through testing.
The student should demonstrate the ability to implement more efficient solutions.

Autonomy of Judgement:

Ability to evaluate the quality of a chosen or potential problem-solving approach to define a solution.

Communicative Skills:

The student must be able to appropriately illustrate the solution created using correct technical language.

Learning Skills:

Capacity for abstraction, reasoning by analogy, and demonstrating creativity in problem-solving.</td></tr>
<tr><td>Measurements and final grade</td><td>
<table>
<tr><th>Grade</th><th>Descriptors</th></tr>
<tr><td>< 18 insufficient</td><td>Fragmented and superficial knowledge of the content, errors in applying concepts, deficient description.</td></tr>
<tr><td>18 - 20</td><td>Sufficient knowledge of the content but general, simple description, uncertainties in applying theoretical concepts.</td></tr>
<tr><td>21 - 23</td><td>Appropriate but not in-depth knowledge of the content, ability to apply theoretical concepts, ability to present content in a simple manner.</td></tr>
<tr><td>24 - 25</td><td>Appropriate and extensive knowledge of the content, fair ability to apply knowledge, ability to present content in an articulate manner.</td></tr>
<tr><td>26 - 27</td><td>Precise and complete knowledge of the content, good ability to apply knowledge, analytical skills, clear and correct description.</td></tr>
<tr><td>28 - 29</td><td>Extensive, complete, and in-depth knowledge of the content, good application of content, good analytical and synthesis skills, confident and correct description.</td></tr>
<tr><td>30
30 e lode</td><td>Excellent knowledge of the content, very extensive, complete, and in-depth, well-consolidated ability to apply the content, excellent analytical, synthesis, and interdisciplinary connection skills, mastery of description.</td></tr>
</table>
</td></tr>
</table>

Further information	Students are advised to rely exclusively on information/communications provided on the official websites of the Department of Computer Science, or on social groups only if they are established and administered exclusively by the teachers of the respective courses: • https://www.uniba.it/it/ricerca/dipartimenti/informatica/didattica/corsi-di-laurea/corsi-di-laurea • https://www.uniba.it/it/ricerca/dipartimenti/informatica • https://elearning.uniba.it/ The course programs are available here: • https://elearning.uniba.it/ The information that all students should be aware of is outlined in the didactic regulations and study manifests available on the website: • https://www.uniba.it/it/ricerca/dipartimenti/informatica/didattica/corsi-di-laurea/corsi-di-laurea Students are advised to be wary of information and materials circulating on unofficial websites or social groups, as they are often found to be unreliable, incorrect, or incomplete. For any doubts, students should request a meeting with the teacher according to the methods provided for appointments.