



Principali informazioni sull'insegnamento

Denominazione dell'insegnamento	Laboratorio di Informatica (corso A)	
Corso di studio	Informatica e Tecnologie per la Produzione del Software	
Anno Accademico	2024/25	
Crediti formativi universitari (CFU) / European Credit Transfer and Accumulation System (ECTS)	6	
Settore Scientifico Disciplinare	ING-INF/05	
Lingua di erogazione	Italiano	
Anno di corso	Primo	
Periodo di erogazione	2^ semestre, le date esatte sono riportate nel manifesto/regolamento	
Obbligo di frequenza	No, La frequenza è fortemente raccomandata	
Sito web del corso di studio	https://elearning.uniba.it/course/index.php?categoryid=285	

Docente/i	
Nome e cognome	Corrado Loglisci
Indirizzo mail	corrado.loglisci@uniba.it
Telefono	080443407
Sede	Dipartimento di Informatica, Via Orabona 4, 70125, Bari. Stanza n.517, 5^ piano.
Sede virtuale	Piattaforma ADA - https://elearning.di.uniba.it/
Sito web del docente	https://www.uniba.it/it/docenti/corrado-loglisci
Ricevimento (giorni, orari e modalità, es. su appuntamento)	Giovedì 15:00-17:00



Syllabus	
Obiettivi formativi	Il corso si propone di introdurre gli step di base dello sviluppo del software da un punto di vista pratico-applicativo, nonché di raffinare le capacità di problem solving limitata. In particolare, lo studente acquisirà la capacità di progettare una soluzione computazionale attraverso la modularizzazione, capacità di implementare una soluzione computazionale con decisioni adeguate su strutture di controllo e strutture dati, capacità di verificare la correttezza di una soluzione computazionale, attraverso testing e programmazione difensiva. Questo è fatto con l'uso del linguaggio di programmazione C.
Prerequisiti	• da Programmazione: basi della programmazione imperativa, ricorsione, processo di compilazione; • da Matematica Discreta: teoria degli insiemi, strutture algebriche elementari
Contenuti di insegnamento (Programma)	1. Linguaggio C (30 ore) Strutture di controllo Tipi di dati Tipi strutturati Puntatori Stringhe Files 2. Strutture dati dinamiche (13 ore) Lista concatenata Pila Coda Albero binario 3. Programmazione modulare (13 ore) Modularità per librerie, tipi astratti, dati astratti Strutturazione dei file sorgente Strutturazione di progetti in Codeblocks Classi di memoria 4. Testing, Programmazione difensiva, Documentazione (7 ore) Generalità sul testing, Asserzioni, Condizioni limite Casi di test e Programmazione difensiva Documentazione del codice sorgente 5. Algoritmi di ordinamento (3 ore) Bubble sort ricorsivo, Insertion sort, Selection sort 6. Ambiente di sviluppo (3 ore) Debugging Doxygen



Testi di riferimento	P. Deitel e H. Deitel Il linguaggio C – Fondamenti e tecniche di programmazione 8ª edizione - Pearson 2016 - ISBN: 9788891901651 J.R. Hanly, E.B. Koffman, Problem solving e programmazione in C, Apogeo, 2013 (https://flipbook.cantook.net/?d=%2F%2Fedigita.cantook.net%2Fflipbook%2Fpublications%2F64885.js&oid=669&c=&m=&l=&r=&f=pdf) 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	Materiale usato a lezione (slide) disponibile su portale ADA Tracce e svolgimenti passati su richiesta dal docente , nonché presentate e svolte durante le ore di esercitazione Contenuti 1,2,5 disponibili sul testo: P. Deitel e H. Deitel Il linguaggio C – Fondamenti e tecniche di programmazione 8ª edizione - Pearson 2016 - ISBN: 9788891901651 Contenuto 3,4 disponibile sul testo: https://flipbook.cantook.net/?d=%2F%2Fedigita.cantook.net%2Fflipbook%2Fpublications%2F64885.js&oid=669&c=&m=&l=&r=&f=pdf
Organizzazione della didattica	



Ore			
Totali	Didattica frontale	Pratica (laboratorio, progetto, esercitazione, altro)	Studio individuale
69 ore	24 ore	45 ore	81 ore
CFU/ETCS			
6 CFU	3 CFU	3 CFU	

Metodi didattici	
	• Lezioni frontali con l'ausilio di slide che riportano esempi per illustrare gli argomenti trattati. • Esercitazioni pratiche sull'utilizzo dei vari principi e tecniche presentate a lezione attraverso esercizi da svolgere singolarmente o in gruppo • Utilizzo della piattaforma di e-learning del Dipartimento di Informatica la distribuzione del materiale e per le interazioni tra docenti e studenti durante e dopo il corso. Il canale MS Teams dell'insegnamento per le interazioni tra docenti e studenti durante e dopo il corso.

Risultati di apprendimento previsti	
Conoscenza e capacità di comprensione	• Acquisire conoscenze che consentano allo studente di comprendere le fasi dello sviluppo di una soluzione computazionale con un approccio classico unito a quello training-on-job che consente allo studente di poter implementare i concetti visti. • Acquisire la capacità di ragionare ed individuare una soluzione ad un problema (algoritmo) secondo il paradigma della programmazione imperativa strutturata;
Conoscenza e capacità di comprensione applicate	• Comprendere i concetti di programmazione modulare come approccio per la progettazione di un software. • Comprendere i concetti di strutture di controllo e strutture dati come approccio per la implementazione di un software. • Comprendere i concetti di testing come approccio per la verifica della correttezza di un software.



Competenze trasversali	Autonomia di giudizio Lo studente deve dimostrare di aver acquisito autonomia di giudizio e capacità di valutazione del software sviluppato da lui o da terzi. Abilità comunicative Lo studente deve essere in grado di illustrare in modo appropriato le caratteristiche tecniche degli strumenti e delle metodologie informatiche apprese nel corso del primo anno di corso, relative allo sviluppo di programmi di media complessità. Capacità di apprendere in modo autonomo Lo studente dovrà mostrare di essere in grado di orientarsi agevolmente nelle problematiche relative alla comprensione e all'utilizzo delle tecnologie e dei metodi di competenza per progettazione, sviluppo e verifica di software e per la loro traduzione in programmi per computer.

Valutazione	
Modalità di verifica dell'apprendimento	La verifica dei risultati formativi raggiunti avviene durante l'esame finale, che prevede: • Una prova di progetto sviluppato in gruppo e da consegnare entro 48 ore dalla consegna della traccia. • Una prova orale individuale in cui si verificano le competenze applicate al progetto e comunque acquisite durante il corso, nonché la capacità espositive dello studente. Alla prova orale si accede previo superamento della parte progettuale rispetto ai seguenti criteri: • assenza errori di compilazione • possibilità ri-compilazione o ri-esecuzione • al più una (delle funzionalità richieste) non realizzata o realizzata ma con gravi mancanze/incorrettezze • programmazione modulare adeguata • svolgimento di testing o uso di programmazione difensiva L'eventuale ammissione alla prova orale è comunicata tramite mail studenti@uniba.it
Criteri di valutazione	



	● Conoscenza e capacità di comprensione: Lo studente dovrà essere in grado di progettare e sviluppare programmi, di media complessità utilizzando il linguaggio C. ● Conoscenza e capacità di comprensione applicate: Lo studente dovrà acquisire competenze relative a: ● Traduzione di semplici algoritmi in programmi correttamente funzionanti e ben documentati; ● Utilizzo di tecniche di programmazione difensiva, per limitare l'introduzione di malfunzionamenti nei programmi; ● Verifica empirica della correttezza dei programmi mediante testing; ● Autonomia di giudizio: Lo studente deve dimostrare capacità di problem solving e una elevata autonomia per la realizzazione di programmi per scenari reali da progettare e realizzare coerentemente alle tecnologie e concetti visti. ● Abilità comunicative: Lo studente deve mostrare di aver sviluppato capacità per apprendere e orientarsi agilmente nelle problematiche che si presentano durante lo sviluppo di software e comunicarne gli aspetti cruciali. ● Capacità di apprendere: Capacità di problem-solving attraverso l'applicazione di nozioni apprese nelle discipline informatiche di base nella pratica della programmazione.														
Criteri di misurazione dell'apprendimento e di attribuzione del voto finale	<table> <tr> <td>< 18 insufficiente</td><td>● Conoscenze frammentarie e superficiali dei contenuti, errori carenti.</td></tr> <tr> <td>18 - 20</td><td>● Conoscenze dei contenuti sufficienti ma generali, descrizione di concetti teorici.</td></tr> <tr> <td>21 - 23</td><td>● Conoscenze dei contenuti appropriate ma non approfondite, 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 presentare i contenuti in modo articolato.</td></tr> <tr> <td>26 - 27</td><td>● Conoscenze dei contenuti precise e complete, buona capacità di analisi, descrizione chiara e corretta.</td></tr> <tr> <td>28 - 29</td><td>● Conoscenze dei contenuti ampie, complete ed approfondite, 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 di applicare i contenuti, ottima capacità di analisi, di sintesi e di collegamento, descrizione.</td></tr> </table>	< 18 insufficiente	● Conoscenze frammentarie e superficiali dei contenuti, errori carenti.	18 - 20	● Conoscenze dei contenuti sufficienti ma generali, descrizione di concetti teorici.	21 - 23	● Conoscenze dei contenuti appropriate ma non approfondite, capacità di presentare i contenuti in modo semplice.	24 - 25	● Conoscenze dei contenuti appropriate ed ampie, discreta capacità di presentare i contenuti in modo articolato.	26 - 27	● Conoscenze dei contenuti precise e complete, buona capacità di analisi, descrizione chiara e corretta.	28 - 29	● Conoscenze dei contenuti ampie, complete ed approfondite, buona capacità di analisi e di sintesi, descrizione sicura e corretta.	30 30 e lode	● Conoscenze dei contenuti molto ampie, complete ed approfondite, capacità di applicare i contenuti, ottima capacità di analisi, di sintesi e di collegamento, descrizione.
< 18 insufficiente	● Conoscenze frammentarie e superficiali dei contenuti, errori carenti.														
18 - 20	● Conoscenze dei contenuti sufficienti ma generali, descrizione di concetti teorici.														
21 - 23	● Conoscenze dei contenuti appropriate ma non approfondite, capacità di presentare i contenuti in modo semplice.														
24 - 25	● Conoscenze dei contenuti appropriate ed ampie, discreta capacità di presentare i contenuti in modo articolato.														
26 - 27	● Conoscenze dei contenuti precise e complete, buona capacità di analisi, descrizione chiara e corretta.														
28 - 29	● Conoscenze dei contenuti ampie, complete ed approfondite, buona capacità di analisi e di sintesi, descrizione sicura e corretta.														
30 30 e lode	● Conoscenze dei contenuti molto ampie, complete ed approfondite, capacità di applicare i contenuti, ottima capacità di analisi, di sintesi e di collegamento, descrizione.														



Altro	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.di.uniba.it/ I programmi degli insegnamenti sono disponibili qui: • https://programmi.di.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. https://elearning.uniba.it/course/view.php?id=2000



Main information on the course

Course name	Laboratory of Computer Science (track A)	
Degree	Informatica e Tecnologie per la Produzione del Software	
Academic year	2024/25	
European Credit Transfer and Accumulation System (ECTS), in Italian Crediti Formativi Universitari (CFU)	6 CFU (each CFU corresponds to 25 hours (h) of 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	ING-INF/05	
Course language	Italian	
Course year	First	
Course period	Semester/Second Semester - exact dates can be found in the didactic regulations	
Course attendance requirement	None, but it is highly recommended to attend classes	
Website of the Degree	https://elearning.uniba.it/course/index.php?categoryid=285	

Teacher(s)

Name and Surname	Corrado Loglisci
email	corrado.loglisci@uniba.it
phone	080443407
office	Dipartimento di Informatica, Via Orabona 4, 70125, Bari. Room n.517, 5 ^a floor.
e-learning platform	Piattaforma ADA - https://elearning.di.uniba.it/
Teacher's homepage	https://www.uniba.it/it/docenti/corrado-loglisci
Office hours	Thursday 15:00-17:00

Syllabus

Course goals	The course goals to introduce the basic steps of software development from a practical-application point of view, as well as to refine the limited problem solving skills. In particular, the student will acquire the ability to design a computational solution through modularization, ability to implement a computational solution with adequate decisions on control structures and data structures, ability to verify the correctness of a computational solution, through testing and defensive programming. This is done with the use of the C programming language.
Prerequisites/requirements	• from Programming: basics of imperative programming, recursion, compilation process; • from Discrete Mathematics: set theory, elementary algebraic structures
Course program	



	1. C Language (30 hours) Control Structures Data Types Structured Types Pointers Strings Files 2. Dynamic Data Structures (13 hours) Linked List Stack Queue Binary Tree 3. Modular Programming (13 hours) Modularity for libraries, abstract types, abstract data Structuring of source files Structuring projects in Codeblocks Memory Classes 4. Testing, Defensive Programming, Documentation (7 hours) General information on testing, Assertions, Boundary Conditions Test Cases and Defensive Programming Source Code Documentation 5. Sorting Algorithms (3 hours) Recursive Bubble Sort, Insertion Sort, Selection Sort 6. Development Environment (3 hours) Debugging Doxygen
Books of reference	P. Deitel e H. Deitel Il linguaggio C – Fondamenti e tecniche di programmazione 8ª edizione - Pearson 2016 - ISBN: 9788891901651 J.R. Hanly, E.B. Koffman, Problem solving e programmazione in C, Apogeo, 2013 (https://flipbook.cantook.net/?d=%2F%2Fedigita.cantook.net%2Fflipbook%2Fpublications%2F64885.js&oid=669&c=&m=&l=&r=&f=pdf)
Notes to the books	Material used in class (slides) available on the ADA portal Tracks and solutions passed on request by the teacher, as well as presented and carried out during the hours of practice Contents 1,2,5 available in the text: P. Deitel and H. Deitel The C language - Fundamentals and programming techniques 8th edition - Pearson 2016 - ISBN: 9788891901651 Contents 3,4 available in the text: https://flipbook.cantook.net/?d=%2F%2Fedigita.cantook.net%2Fflipbook%2Fpublications%2F64885.js&oid=669&c=&m=&l=&r=&f=pdf



Organization of the didactic activities				
Hours				
Total	Lectures	Practice sessions	Project work	Individual study
Hours 69	Hours 24	hours 45	Hours 0	Hours 81
CFU/ETCS				
CFU 6	CFU 3	CFU 3	CFU	

Teaching methods	
	• lectures by using slides that show examples to illustrate the topics covered. • Practical sessions on the use of the various principles and techniques presented in class through exercises to be done individually or in groups • Use of the e-learning platform of the Department of Computer Science for the distribution of the material and for interactions between teachers and students during and after the course. The MS Teams channel of the teaching for interactions between teachers and students during and after the course.

Expected learning outcomes	
Knowledge and understanding	• Acquire knowledge that allows the student to understand the development phases of a computational solution with a classical approach combined with the training-on-job approach that allows the student to implement the concepts seen. • Acquire the ability to reason and identify a solution to a problem (algorithm) according to the paradigm of structured imperative programming;
Applying knowledge and understanding	• Understand the concepts of modular programming as an approach to designing software. • Understand the concepts of control structures and data structures as an approach to implementing software. • Understand the concepts of testing as an approach to verifying the correctness of software.
Other skills	<i>Making judgements</i> The student must demonstrate that he has acquired autonomy of judgment and ability to evaluate the software developed by him or by third parties. <i>Communication</i> The student must be able to appropriately illustrate the technical characteristics of the IT tools and methodologies learned during the first year of the course, relating to the development of medium-complexity programs.



	<i>Learning skills</i> The student must demonstrate that he or she is able to easily navigate issues related to the understanding and use of technologies and methods of expertise for the design, development and verification of software and for their translation into computer programs.
Assessment	
Assessment methods	The assessment of the training results achieved takes place during the final exam, which includes: • A project test developed in a group and to be delivered within 48 hours of the delivery of the outline. • An individual oral test in which the skills applied to the project and in any case acquired during the course are verified, as well as the student's presentation skills. The oral test is accessed after passing the project part with respect to the following criteria: • absence of compilation errors possibility of re-compilation or re-execution • at most one (of the requested functions) not implemented or implemented but with serious shortcomings/incorrectness • adequate modular programming • carrying out testing or use of defensive programming Possible admission to the oral test is communicated via email studenti@uniba.it
Evaluation criteria	• Knowledge and understanding: The student must be able to design and develop programs of medium complexity using the C language. • Applied knowledge and understanding: The student must acquire skills related to: • Translation of simple algorithms into correctly functioning and well-documented programs; • Use of defensive programming techniques, to limit the introduction of malfunctions in programs; • Empirical verification of the correctness of programs through testing; • Autonomy of judgment: The student must demonstrate problem solving skills and a high level of autonomy for the creation of programs for real scenarios to be designed and implemented consistently with the technologies and concepts seen. • Communication skills: The student must demonstrate that he has developed the ability to learn and orient himself easily in the problems that arise during software development and communicate the crucial aspects.



	● Learning ability: Problem-solving skills through the application of notions learned in basic computer science disciplines in programming practice.	
Measurements and final grade	< 18 insufficient	Fragmentary and superficial knowledge of the contents, errors in applying the concepts, poor description.
	18 - 20	Sufficient but general knowledge of content, simple description, uncertainties in the application of theoretical concepts.
	21 - 23	Appropriate but not in-depth knowledge of the contents, ability to apply theoretical concepts, ability to present the contents in a simple way.
	24 - 25	Appropriate and broad knowledge of the contents, fair ability to apply knowledge, ability to present the contents in an articulated manner.
	26 - 27	Precise and complete knowledge of the contents, good ability to apply knowledge, analytical skills, clear and correct description.
	28 - 29	Broad, complete and in-depth knowledge of the contents, good application of the contents, good analysis and synthesis skills, confident and correct description.
	30 30 with honors	Very broad, complete and in-depth knowledge of the contents, well-established ability to apply the contents, excellent capacity for analysis, synthesis and interdisciplinary connections, mastery of description.
Further information	Students are advised to rely exclusively on the information/communications provided on the official websites of the Department of Computer Science, or on social groups only if created and managed exclusively by the professors of the relevant 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.di.uniba.it/ The courses' programs are available here: ● https://programmi.di.uniba.it/ The information that all students should know is written in the Teaching Regulations and Study Manifestos 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 information and materials circulating on unofficial websites or social groups, as they have often been found to be unreliable, incorrect or incomplete. For any doubts, ask for a meeting with the teacher according to the methods provided for reception.	
	https://elearning.uniba.it/course/view.php?id=2000	