Principali informazioni sull'insegnamento

Denominazione dell'insegnamento	**Programmazione (track cognomi A-D)**	
Corso di studio	Informatica	
Anno Accademico	2025/26	
Crediti formativi universitari (CFU) / European Credit Transfer and Accumulation System (ECTS)		Crediti formativi universitari (CFU) / European Credit Transfer and Accumulation System (ECTS)
Settore Scientifico Disciplinare	ING-INF/05	
Lingua di erogazione	Italiano	
Anno di corso	Primo	
Periodo di erogazione	1^ semestre, le date esatte sono riportate nel manifesto/regolamento	
Obbligo di frequenza	NO, ma la frequenza è fortemente raccomandata	
Sito web del corso di studio	https://www.uniba.it/it/ricerca/dipartimenti/informatica/didattica/corsi-di-laurea/corsi-di-laurea	

Docente/i

Nome e cognome	Fabio Abbattista
Indirizzo mail	Fabio.abbattista@uniba.it
Telefono	080-5443298
Sede	Dipartimento di Informatica, Via Orabona 4, 70125, Bari. Stanza n.666, 6^ piano.
Sede virtuale	Piattaforma e-learning UNIBA - https://elearning.uniba.it/
Sito web del docente	
Ricevimento (giorni, orari e modalità, es. su appuntamento)	Martedì 15:00 – 16:00, ma anche per e-mail

Syllabus

Obiettivi formativi	Il corso si propone di introdurre gli elementi base della programmazione imperativa strutturata per formulare soluzioni algoritmiche a problemi di complessità limitata. In particolare, lo studente acquisirà la capacità di usare il linguaggio di

<table>
<tr><td></td><td>programmazione C come strumento per modellare problemi e formalizzarne le soluzioni.</td></tr>
<tr><td>Prerequisiti</td><td>Buona comprensione della lingua inglese</td></tr>
<tr><td>Contenuti di insegnamento (Programma)</td><td>
<table>
<tr><th>Mod</th><th>Argomenti (l'indicazione oraria si intende stimata)</th><th>Ore</th></tr>
<tr><td>1</td><td>Presentazione del corso.
Introduzione alla programmazione.
Un primo algoritmo in pseudocodice: cucinare un primo.</td><td>3</td></tr>
<tr><td>2</td><td>Gli algoritmi.
Lo pseudocodice.</td><td>3</td></tr>
<tr><td>3</td><td>Le strutture di controllo:
Il comando di selezione if.
Il comando di selezione if else.
Il comando di iterazione while.</td><td>3</td></tr>
<tr><td>4</td><td>Esercitazione sugli algoritmi: la somma di 2 numeri come incrementi successivi.
Esercitazione sugli algoritmi: il prodotto di 2 numeri come somme successive.</td><td>4</td></tr>
<tr><td>5</td><td>Formulazione degli algoritmi con processo top down per raffinamenti successivi.</td><td>3</td></tr>
<tr><td>6</td><td>Strutture di controllo nidificate.</td><td>3</td></tr>
<tr><td>7</td><td>Esercitazione sulle strutture di controllo nidificate con uso di pseudocodice.</td><td>4</td></tr>
<tr><td>8</td><td>Formulazione degli algoritmi con processo top down per raffinamenti successivi: studio di un caso.</td><td>3</td></tr>
<tr><td>9</td><td>Gli operatori logici.
Operatori di uguaglianza (= =) e di assegnamento (=).</td><td>3</td></tr>
<tr><td>10</td><td>Esercitazione sulle strutture di controllo nidificate con uso di pseudocodice.</td><td>4</td></tr>
<tr><td>11</td><td>Tipi di dato semplici: interi, reali, caratteri e loro rappresentazione in C.</td><td>3</td></tr>
<tr><td>12</td><td>I vettori.
La dichiarazione dei vettori.
Esempi sui vettori.</td><td>3</td></tr>
<tr><td>13</td><td>Esercitazione sui vettori: ricerca del massimo.</td><td>4</td></tr>
<tr><td>14</td><td>Studio di un caso: calcolare la media e la mediana usando i vettori.
La ricerca nei vettori (lineare e binaria).</td><td>3</td></tr>
<tr><td>15</td><td>Uso del compilatore.
Ambiente di sviluppo Eclipse.
Cenni al debugging.</td><td>3</td></tr>
<tr><td>16</td><td>Esercitazione sui vettori: ricerca di un valore, ricerca di un valore vincolato.</td><td>4</td></tr>
<tr><td>17</td><td>Istruzioni di controllo in C: selezione e iterazione.</td><td>3</td></tr>
<tr><td>18</td><td>Istruzioni di Input/Output del C.</td><td>3</td></tr>
<tr><td>19</td><td>Esercitazione in C: codifica degli algoritmi realizzati in precedenza.</td><td>4</td></tr>
<tr><td>20</td><td>Astrazione sui dati.
Tipi di dato utente.
Operazioni di accesso alle strutture dati utente.</td><td>3</td></tr>
<tr><td>21</td><td>Matrici e tabelle come dati utente: i record.</td><td>3</td></tr>
<tr><td>22</td><td>Manipolazione delle matrici.
Esercitazione sui vettori bidimensionali.</td><td>4</td></tr>
<tr><td>23</td><td>Astrazione funzionale: funzioni e procedure e loro realizzazione in pseudocodice.</td><td>3</td></tr>
</table>
</td></tr>
</table>

	24	Le funzioni in C: le definizioni di funzione e i prototipi di funzione.	3
	25	Esercitazione con le funzioni in C: decomporre un semplice programma in 3 funzioni, input, elaborazione e output.	4
	26	Lo stack delle chiamate di funzione e i record di attivazione. Regole di visibilita' in C.	3
	27	Invocare le funzioni: chiamata per valore e per riferimento.	3
	28	Esercitazione con le funzioni in C: lo scambio di 2 valori, somma di vettori.	4
	29	Vettori statici ed automatici. Passare i vettori come argomento delle funzioni.	3
	30	Puntatori: dichiarazione e inizializzazione. Operatore di indirizzo (&) e operatore di dereferenziazione (*). Puntatori come parametri di funzioni.	3
	31	Esercitazione sui puntatori: trovare il massimo e il minimo di un vettore passato come argomento alla funzione.	4
	32	La gerarchia dei dati. I file e gli stream. Creare un file ad accesso sequenziale. Leggere e scrivere i dati da un file ad accesso sequenziale.	3
	33	I file ad accesso casuale. Creare un file ad accesso casuale. Leggere e scrivere dati da un file ad accesso casuale.	3
	34	Esercitazione sui file sequenziali: creazione di un file di testo e aggiornamento del contenuto del file di testo	4
	35	Introduzione alla ricorsione: fattoriale ricorsivo, ricerca binaria ricorsiva e Torre di Hanoi	3

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: J.R. Hanly, E.B. Koffman, Problem solving e programmazione in C, Apogeo, 2013. ISBN: 8838786410 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 utilizzerà delle slide che verranno fornite. Il testo di riferimento contiene tutti gli argomenti del corso attinenti al linguaggio di programmazione C, pertanto si consiglia di studiare dal testo e di svolgere in autonomia e costantemente tutti gli esercizi inseriti alla fine di ogni capitolo trattato a lezione. Sul sito del corso (v. sopra 'sito web del corso') sono disponibili: ● Materiale didattico utilizzato a lezione; ● alcune tracce di prove scritte di esami, con esempi di tracce svolte;
Organizzazione della didattica	

Ore			
Totali	Didattica frontale	Laboratorio ed esercitazioni	Studio individuale
300 ore	72 ore	45 ore	183 ore
CFU/ETCS			
12 CFU	9 CFU	3 CFU	

Metodi didattici	
	Lezioni frontali, esercitazioni ed attività autonome e di gruppo in aula e a casa (come dettagliato nel programma). Gli studenti non frequentanti possono lavorare singolarmente prendendo accordi con il docente.

Risultati di apprendimento previsti	
Conoscenza e capacità di comprensione	● Acquisire conoscenze che consentano allo studente di comprendere come si può indicare ad un elaboratore elettronico (macchina automatica di impiego universale, hardware) la soluzione di un problema o di una classe di problemi, che l'elaboratore può risolvere, con un metodo ed un linguaggio appropriato, creando un apposito programma (software) eseguibile dall'elaboratore. ● 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 l'uso di un linguaggio di progettazione non convenzionale (es. pseudocodice) per descrivere con un formalismo semplice un algoritmo; ● Comprendere il lessico, la sintassi e la semantica del linguaggio di programmazione C; ● Acquisire la capacità di scrivere un programma strutturato in linguaggio C; ● Acquisire la capacità di individuare casi di test per il dominio cui fa riferimento il programma creato; ● Acquisire la capacità di utilizzare un ambiente di sviluppo (es. Eclipse) per trasformare il programma sorgente (in C) in programma eseguibile ed eseguirlo.
Competenze trasversali	**Autonomia di giudizio** ● Acquisire la capacità di verificare che l'algoritmo individuato risponda alle specifiche di un problema; ● Acquisire la capacità di verificare che i risultati ottenuti dopo l'esecuzione del programma siano quelli attesi.

	Abilità comunicative ● Imparare a commentare il codice prodotto al fine di renderlo comprensibile e agevolmente modificabile da altri professionisti, con l'obiettivo di sviluppare in team. **Capacità di apprendere in modo autonomo** ● Capacità di approfondire concetti attraverso lo studio autonomo di materiale prodotto e proposto dal docente; ● Capacità di completare autonomamente il percorso formativo previsto dal testo di riferimento, oltre i contenuti previsti dal programma dell'insegnamento.

Valutazione	
Modalità di verifica dell'apprendimento	Alcune prove di valutazione intermedie, con valore esonerante, si svolgono nelle ore di esercitazione del corso, a partire da inizio novembre. Le prove consistono nella soluzione di un problema individuando l'algoritmo e sviluppando il relativo programma in C, analogamente a quanto spiegato nel corso delle lezioni. Il voto sarà espresso in booleano. L'esonero puo' essere utilizzato esclusivamente nella prima sessione di esami (gennaio/febbraio). La votazione è in trentesimi. Gli studenti che scelgono di non partecipare alle prove di valutazione intermedie, ovvero non le superano, sostengono l'esame a partire da gennaio. La modalità della prova d'esame regolare è analoga a quella descritta sopra per le prove intermedie. Materiali permessi per sostenere le prove di valutazione intermedia e la prova scritta d'esame: testo di riferimento in formato esclusivamente cartaceo. Il voto finale conseguito viene proposto esclusivamente sulla piattaforma Esse3. Incentivi alla frequenza: l'eventuale lode viene più frequentemente attribuita agli studenti che per la stragrande maggioranza delle lezioni hanno frequentato, interagito nel corso della lezione, proposto soluzioni e risolto i casi proposti dal docente a lezione.
Criteri di valutazione	● **Conoscenza e capacità di comprensione:** ○ Lo studente dovrà essere in grado di analizzare e risolvere semplici problemi e di generalizzare soluzioni per una classe di problemi con lo stile della programmazione strutturata. ● **Conoscenza e capacità di comprensione applicate:** ○ Lo studente dovrà essere in grado di codificare le soluzioni ideate descrivendole in pseudocodice e nel linguaggio di programmazione C;

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

Lo studente dovrà essere in grado di utilizzare un ambiente di sviluppo e dimostrare di conoscere il linguaggio C;

Autonomia di giudizio:

Lo studente dovrà essere in grado di correggere e validare il corretto funzionamento dei programmi sviluppati.

Abilità comunicative:

Lo studente dovrà essere in grado di rendere il codice scritto comprensibile ad altri, mediante la sua descrizione generale e commenti specifici alle istruzioni e alle strutture di controllo utilizzate.

Capacità di apprendere:

Lo studente dovrà essere in grado di trasformare autonomamente algoritmi descritti con pseudo-codice in programmi in linguaggio C;
Lo studente dovrà essere in grado di utilizzare le soluzioni alternative descritte nel testo di riferimento, se non descritte nel corso delle lezioni, come ad esempio le diverse modalità di dichiarazione delle variabili.

</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
https://www.uniba.it/it/ricerca/dipartimenti/informatica
https://elearning.uniba.it/

I programmi degli insegnamenti sono disponibili qui:

https://elearning.uniba.it/

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

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.

_____

Main information on the course

Course name	**Programmazione (track cognomi A-D)**	
Degree	Informatica	
Academic year	2025/26	
European Credit Transfer and Accumulation System (ECTS), in Italian Crediti Formativi Universitari (CFU)	European Credit Transfer and Accumulation System (ECTS) , in Italian Crediti formativi universitari (CFU)	
Settore Scientifico Disciplinare	ING-INF/05	
Course language	Italian	
Course year	First	
Course period	1st semester, the exact dates are indicated annually in the degree course regulations	
Course attendance requirement	It is highly recommended to attend classes	
Website of the degree	e-learning UNIBA - https://elearning.uniba.it/	

Teacher(s)

Name and Surname	Fabio Abbattista
email	Fabio.abbattista@uniba.it
phone	080-5443298
office	Dipartimento di Informatica, Via Orabona 4, 70125, Bari. Stanza n.666, 6^ piano.
e-learning platform	Microsoft Teams – Canale Programmazione A – Informatica – 2024-25
Teacher's homepage	
Office hours	Tuesday 15:00 – 16:00, and by email

Syllabus

Course goals

The course aims to introduce the basic elements of structured imperative programming to formulate algorithmic solutions to problems of limited complexity. In particular, the student will acquire the ability to use the C programming language as a tool for modeling problems and formalizing their solutions.

Prerequisites/requirements

Good understanding of the English language.

Course program

Mod	Topics	Hours
1	Course presentation. Introduction to computer programming. A first algorithm: to cook a first course.	3
2	Algorithms. Pseudo-code.	3
3	Control structures: Selection structure (if-then). Binary selection structure (if-then-else). Iterative structure (while).	3
4	Algorithm exercise: Summing 2 numbers by using increments. Algorithm exercise: Multiplying 2 numbers by using increments.	4
5	Designing algorithms by stepwise refinements.	3

	6	Nested control structures.	3
	7	Nested structure control exercise using pseudo-code.	4
	8	Designing algorithms by stepwise refinements: Case study	3
	9	Boolean operators. Equality operator (= =) and assignment operator (=).	3
	10	Nested structure control exercise using pseudo-code.	4
	11	Simple data types: Integer, real, character and their representation in the C language.	3
	12	Array. Array statement. Array examples.	3
	13	Array exercise: Finding the maximum element.	4
	14	Case study: Computing average and median using arrays. Serach algorithms for array (linear and binary algorithms).	3
	15	Using a compiler. Eclipse development environment. Brief introduction to debugging.	3
	16	Array exercise: Searching an element, searching a bounded element.	4
	17	Control structures in C language: Selection and iteration.	3
	18	Input/Output instructions in C language.	3
	19	C language exercise: Coding the previously designed algorithms.	4
	20	Data abstraction. User defined data types. Access procedures to user defined data structures.	3
	21	User defined matrices and tables: Record data structure.	3
	22	Handling matrices. Exercises on bi-dimensional arrays.	4
	23	Functional abstraction: Functions, procedures and their design with pseudo-code.	3
	24	Functions in C language: Functions definition and prototype.	3
	25	Function exercise in C language: Decomposing a function into 3 functions, input, processing and output.	4
	26	Function calls stack and the activation record. Visibility rules in C language.	3
	27	Function calls: By value and by reference.	3
	28	Function exercise in C language: Swapping the values of two variables, summing the elements of an array.	4
	29	Static and automatic vectors. Array as function arguments.	3
	30	Pointers: Definition and initialization Address operator (&) e dereference operator (*) Pointers as functions arguments.	3
	31	Pointers exercise: Finding maximum and minimum of an array passed as argument to the function.	4
	32	Data hierarchy Files and streams. To create a sequential access file. To read and write data from a sequential access file.	3
	33	Random access file. To create a random access file. To read and write data from a random access file.	3
	34	Sequentiual access file exercise: Creating a text file, updating the content of the text file.	4
	35	Introduction to recursive algorithms: recursive factorial algorithm, Recursive binary search algorithm, Hanoi tower algorithm.	3
Books of reference	Text to study from::		

	P. Deitel e H. Deitel Il linguaggio C – Fondamenti e tecniche di programmazione 8^edizione - Pearson 2016 - ISBN: 9788891901651 Additional text, optional: J.R. Hanly, E.B. Koffman, Problem solving e programmazione in C, Apogeo, 2013. ISBN: 8838786410 Students who wish can obtain texts on loan from the Library. Could it be convenient to check their availability through 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 lessons the teacher will use slides that retrace the contents of the book, therefore they will not be provided. The reference text contains all the topics of the course, therefore it is advisable to study from the text and to carry out independently and constantly all the exercises included at the end of each chapter covered in class. On the Google groups platform (see above 'virtual office') are available: • supporting video material used in class; • some traces of written tests of exams, with examples of traces carried out;

Organization of the didactic activities

Hours			
Total	Lectures	Practice sessions	Individual study
300 hours	72 hours	45 hours	183 hours
CFU/ETCS			
12 CFU	9 CFU	3 CFU	

Teaching methods

	Lectures, exercises and autonomous and group activities in the classroom and at home (as detailed in the program). Non-attending students can work individually by making arrangements with the teacher.

Expected learning outcomes

Knowledge and understanding	• Acquire knowledge that allows the student to understand how it is possible to indicate to an electronic computer (automatic machine for universal use, hardware) the solution of a problem or a class of problems, which the computer can solve, with a method and a appropriate language, creating a special program (software) executable by the computer. • 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 use of an unconventional design language (eg pseudocode) to describe an algorithm with a simple formalism; • Understand the lexicon, syntax and semantics of the C programming language;

	• Acquire the ability to write a structured program in C language; • Acquire the ability to identify test cases for the domain to which the created program refers; • Acquire the ability to use a development environment (eg Eclipse) to transform the source program (in C) into an executable program and run it.
Other skills	*Making judgements* • Acquire the ability to verify that the identified algorithm meets the specifics of a problem; • To acquire the ability to verify that the results obtained after the execution of the program are those expected. *Communication* • Learn how to comment the product code in order to make it understandable and easily modifiable by other professionals, with the aim of developing in a team. *Learning skills* • Ability to deepen concepts through the independent study of video material produced and proposed by the teacher; • Ability to autonomously complete the educational path envisaged by the reference textbook, in addition to the contents envisaged by the teaching programme.

Assessment

Assessment methods	Some mid-term evaluation tests, with an exempt value, are held in the course of the lessons. The test consists in solving a problem by identifying the algorithm and developing the relative program in C, similarly to what is explained in the lessons. The vote will be expressed in thirtieths. The grades from the exemption can only be used in the first exam session (January/February). Students who choose not to take part in the mid-term assessment tests, or fail to pass them, take the exam starting in January. The modality of the regular exam is similar to that described above for the two intermediate tests (written exam and test). Materials allowed to take the first midterm evaluation test and the written exam test: reference text in paper format only. The results of all the tests are communicated with a public list (matriculation number and grade achieved) by email or social media. The final grade obtained is proposed exclusively on the Esse3 platform. Attendance incentives: any praise is most frequently given to students who, for the vast majority of lessons, have attended, interacted during the lesson, proposed solutions and solved the cases proposed by the teacher in class.
Evaluation criteria	
Measurements and final grade	**Mark** | **Description**

< 18 poor	Fragmentary and superficial knowledge of the contents, errors in applying the concepts, deficient description.
18 - 20	Sufficient but general content knowledge, simple description, uncertainties in the application of theoretical concepts.
21 - 23	Appropriate but not in-depth knowledge of content, ability to apply theoretical concepts, ability to present content in a simple way.
24 - 25	Appropriate and extensive knowledge of the contents, good ability to apply knowledge, ability to present the contents in an articulated way.
26 - 27	Precise and complete content knowledge, good ability to apply knowledge, analytical skills, clear and correct description.
28 - 29	Wide, complete and in-depth knowledge of the contents, good application of the contents, good capacity for analysis and synthesis, safe and correct description.
30 30 e lode	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 Computer Science Department, or on social groups only if set up and administered exclusively by the teachers of the related courses:

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

Course schedules are available here:
• https://programmi.di.uniba.it/

The information that all students should know is written in the Teaching regulations and study posters available on the site:
• https://www.uniba.it/it/ricerca/dipartimenti/informatica/didattica/corsi-dilaurea/corsi-di-laurea

Students are advised to be wary of information circulating on unofficial sites or social groups, as they are often found to be unreliable, incorrect or incomplete.