Principali informazioni sull'insegnamento

Denominazione dell'insegnamento	**Programmazione (track MM)**
Corso di studio	Informatica e Comunicazione Digitale
Anno Accademico	2024/25
Crediti formativi universitari (CFU) / European Credit Transfer and Accumulation System (ECTS)	12 CFU
Settore Scientifico Disciplinare	INF/01
Lingua di erogazione	Italiano
Anno di corso	Primo
Periodo di erogazione	1^ semestre, le date esatte sono indicate annualmente nel manifesto/regolamento
Obbligo di frequenza	La frequenza è fortemente raccomandata
Sito web del corso di studio	https://www.uniba.it/it/corsi/cdl-informatica-comunicazione-digitale-taranto/corso-di-laurea-triennale-in-informatica-e-comunicazione-digitale

Docente/i

Nome e cognome	Giovanni Dimauro
Indirizzo mail	giovanni.dimauro@uniba.it
Telefono	080-5443294
Sede	Dipartimento di Informatica, Via Orabona 4, 70125, Bari. Stanza n.617, 6^ piano.
Sede virtuale	Piattaforma e-learning UNIBA - https://elearning.uniba.it/
Sito web del docente	http://www.di.uniba.it/~dimauro/
Ricevimento (giorni, orari e modalità, es. su appuntamento)	appuntamento per email

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. Lettura individuale da parte dello studente del 1^ capitolo del testo di riferimento (v. sotto): “Introduzione ai computer, a Internet e al web”</td></tr>
<tr><td>Contenuti di insegnamento (Programma)</td><td>
<table>
<tr><th>Mod</th><th>Argomenti</th><th>Ore</th></tr>
<tr><td>1</td><td>Presentazione del corso, contenuti, modalità d'esame, esoneri, frequenza alle lezioni, orari, modalità di esercitazione in aula, ecc.
Introduzione alla programmazione
Un semplice programma C: visualizzare una riga di testo.
Un altro semplice programma C: sommare due interi.</td><td>3</td></tr>
<tr><td>2</td><td>Uso del compilatore
ambiente di sviluppo Xcode
cenni al debugging
esercitazioni e approfondimenti degli argomenti: Un semplice programma C: visualizzare una riga di testo.</td><td>3</td></tr>
<tr><td>3</td><td>Nozioni sulla memoria
L'aritmetica del C
Prendere delle decisioni: gli operatori di uguaglianza e relazionali.
Gli algoritmi
Lo pseudocodice.
Le strutture di controllo
Il comando di selezione if</td><td>3</td></tr>
<tr><td>4</td><td>Il comando di selezione if else.
Il comando di iterazione while.
Formulazione degli algoritmi: studio di un caso: l'iterazione controllata da un contatore.
Formulazione degli algoritmi con processo top down per raffinamenti successivi: studio di un caso: iterazione controllata da un valore sentinella.
Conversione di tipo, precisione.</td><td>4</td></tr>
<tr><td>5</td><td>Strutture di controllo nidificate
esercitazione con uso di pseudocodice
Formulazione degli algoritmi con processo top down per raffinamenti successivi: studio di un caso: strutture di controllo nidificate</td><td>3</td></tr>
<tr><td>6</td><td>Gli operatori di incremento e di decremento
Gli elementi dell’iterazione
Iterazione controllata da un contatore
il comando di iterazione FOR
esempi di utilizzo del comando FOR
Il comando di selezione multipla switch
introduzione al comando break, studio di un caso.
Il comando di iterazione do...while
Le istruzioni break e continue</td><td>4</td></tr>
<tr><td>7</td><td>Gli operatori logici
Operatori di uguaglianza (= =) e di assegnamento (=);
Riassunto della programmazione strutturata.
I moduli di programma in C</td><td>3</td></tr>
<tr><td>8</td><td>Le funzioni della libreria matematica
Le funzioni
Le definizioni di funzione
I prototipi di funzione
Lo stack delle chiamate di funzione e i record di attivazione
I file di intestazione</td><td>3</td></tr>
</table>
</td></tr>
</table>

	Invocare le funzioni: chiamata per valore e per riferimento Generazione di numeri casuali	
9	Esercitazione/autovalutazione: sviluppo flowchart, attività svolta con assistenza del docente	4
10	Esempio: un gioco d'azzardo Le classi di memoria Le regole di visibilità	4
11	La ricorsione	2
12	I vettori La dichiarazione dei vettori Esempi sui vettori	3
13	Esercitazione e studio individuale di materiale video prodotto dal Docente, a supporto delle lezioni teoriche. Il video descrive lo sviluppo di un flowchart prendendo spunto dall'esercizio 4.40 'Crescita della popolazione mondiale' del testo di riferimento. Il metodo di progettazione rispetta i principi della programmazione strutturata e utilizza strutture di controllo di base del linguaggio di programmazione C - discussione con il docente.	4
14	Ulteriori esempi sui vettori	3
15	Vettori statici ed automatici Passare i vettori alle funzioni, passaggio per riferimento (indirizzo) e per valore, qualificatore 'const' Algoritmi di ordinamento: bubble sort	4
16	Studio di un caso: calcolare la media, mediana e la moda usando i vettori La ricerca nei vettori (lineare), algoritmo e programma per la ricerca lineare in un vettore	4
17	La ricerca nei vettori (binaria) Vettori multidimensionali.	4
18	Manipolazione delle matrici esercitazione sulle matrici bidimensionali (fig.6.22 del testo 4^ed)	3
19	Esercitazione e studio individuale di materiale video prodotto dal Docente, a supporto delle lezioni teoriche. Il video descrive lo sviluppo di un flowchart dell'esercizio 'dado a 20 facce'. Il metodo di progettazione rispetta i principi della programmazione strutturata - discussione con il docente.	3
20	Esercitazione: Individuazione dei numeri primi. Crivello di Eratostene.	3
21	Esercitazione e studio individuale di materiale video prodotto dal docente, a supporto delle lezioni teoriche. Il video descrive il passaggio da un algoritmo descritto con flowchart al codice in Linguaggio C, prendendo spunto dall'esercizio 'dado a 20 facce'. Il metodo di progettazione rispetta i principi della programmazione strutturata e utilizza strutture di controllo di base del linguaggio di programmazione C - discussione con il docente.	3
22	Esercitazione e studio individuale di materiale video prodotto dal Docente, a supporto delle lezioni teoriche. Il video descrive lo sviluppo di un flowchart prendendo spunto dall'esercizio 'Craps - un gioco d'azzardo' del testo di riferimento. Il metodo di progettazione rispetta i principi della programmazione strutturata - discussione con il docente.	3
23	preparazione all'esonero.	4
24	simulazione autonoma in preparazione del 1^ esonero, assistita dal docente	3
25	Auto correzione esonero assistita dal docente	4
26	La gerarchia dei dati	4

	I file e gli stream Creare un file ad accesso sequenziale Leggere i dati da un file ad accesso sequenziale	
27	Leggere i dati da un file ad accesso sequenziale (approfondimenti) Programma per l'interrogazione del credito.	3
28	I file ad accesso casuale Creare un file ad accesso casuale	4
29	Scrivere i dati in modo casuale in un file ad accesso casuale Leggere i dati in modo casuale da un file ad accesso casuale	3
30	Studio di un caso: un programma per l'elaborazione delle transazioni	3
	esercitazione autonoma, assistita dal docente; Correzione collettiva esercitazione svolta. Preparazione al 2^ esonero.	3
	Esercitazione e Simulazione 2^ esonero	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)

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 ripercorrono i contenuti del libro, pertanto non verranno fornite. Il testo di riferimento contiene tutti gli argomenti del corso, 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.

Sulla piattaforma e-learning di Uniba (v. sopra 'sede virtuale') sono disponibili:
- materiale video di supporto 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) e l'uso di una rappresentazione grafica (es. flow chart) 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. Xcode, Visual Studio) 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 video 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	Una prima prova di valutazione intermedia, con valore esonerante, si tiene in prossimità della settimana di interruzione delle lezioni, normalmente collocata intorno alla metà di novembre. La prova consiste 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 trentesimi. Una seconda prova di valutazione intermedia, con valore esonerante, si tiene immediatamente prima o dopo le vacanze natalizie, in base all'andamento del corso. La seconda prova consiste in un test a risposta multipla chiusa, in particolare saranno somministrati circa 50 domande ed il tempo previsto è di norma 30 minuti. Ogni risposta errata verrà penalizzata con 0,25/50esimi di punto. Il risultato ottenuto viene normalizzato a 30. Il voto risultante dalla media dei voti ottenuti nelle due prove di esonero viene proposto dal docente per essere verbalizzato. I voti dell'esonero possono essere utilizzati esclusivamente nella prima sessione di esami (gennaio/febbraio). Lo studente può, se preferisce, integrare l'esame con una prova orale. 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 prova di teoria, se sostenuta e superata nella valutazione intermedia, può essere utilizzata dallo studente come prova di teoria dell'esame regolare. La modalità della prova d'esame regolare è analoga a quella descritta sopra per le due prove intermedie (prova scritta e test). Materiali permessi per sostenere la prima prova di valutazione intermedia e la prova scritta d'esame: testo di riferimento in formato esclusivamente cartaceo. I risultati di tutte le prove vengono comunicati con lista pubblica (numero di matricola e voto conseguito) per email o social. 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, in flowchart strutturati e nel linguaggio di programmazione C;
  - Lo studente dovrà essere in grado di utilizzare un ambiente di sviluppo a sua scelta 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 flowchart 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.

Criteri di misurazione dell'apprendimento e di attribuzione del voto finale

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

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 dalle informazioni circolanti su siti o gruppi social non ufficiali, poichè spesso sono risultate non affidabili, non corrette o incomplete.

Link al corso sulla piattaforma e-learning di Uniba: https://elearning.uniba.it/

Main information on the course

Course name	**Programmazione (track MM)**
Degree	Informatica e Comunicazione Digitale
Academic year	2024/25
European Credit Transfer and Accumulation System (ECTS), in Italian Crediti Formativi Universitari (CFU)	12 CFU
Settore Scientifico Disciplinare	INF/01
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	https://www.uniba.it/it/corsi/cdl-informatica-comunicazione-digitale-taranto/corso-di-laurea-triennale-in-informatica-e-comunicazione-digitale

Teacher(s)

Name and Surname	Giovanni Dimauro
email	giovanni.dimauro@uniba.it
phone	080-5443294
office	Dipartimento di Informatica, Via Orabona 4, 70125, Bari. Stanza n.617, 6^ piano.
e-learning platform	https://elearning.di.uniba.it/
Teacher's homepage	http://www.di.uniba.it/~dimauro/
Office hours	appointment with 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. Individual reading by the student of the 1st chapter of the reference book (see below): "Introduction to computers, the Internet and the web"

Course program

Mod	Argomenti	Ore
1	Presentation of the course, contents, exam methods, exemptions, attendance at lessons, timetables, methods of classroom practice, etc. Introduzione alla programmazione Un semplice programma C: visualizzare una riga di testo. Un altro semplice programma C: sommare due interi.	3
2	Uso del compilatore ambiente di sviluppo Xcode cenni al debugging esercitazioni e approfondimenti degli argomenti: Un semplice programma C: visualizzare una riga di testo.	3
3	Nozioni sulla memoria L'aritmetica del C Prendere delle decisioni: gli operatori di uguaglianza e relazionali.	3

		Gli algoritmi Lo pseudocodice. Le strutture di controllo Il comando di selezione if	
	4	Il comando di selezione if else. Il comando di iterazione while. Formulazione degli algoritmi: studio di un caso: l'iterazione controllata da un contatore. Formulazione degli algoritmi con processo top down per raffinamenti successivi: studio di un caso: iterazione controllata da un valore sentinella. Conversione di tipo, precisione.	4
	5	Strutture di controllo nidificate esercitazione con uso di pseudocodice Formulazione degli algoritmi con processo top down per raffinamenti successivi: studio di un caso: strutture di controllo nidificate	3
	6	Gli operatori di incremento e di decremento Gli elementi dell'iterazione Iterazione controllata da un contatore il comando di iterazione FOR esempi di utilizzo del comando FOR Il comando di selezione multipla switch introduzione al comando break, studio di un caso. Il comando di iterazione do...while Le istruzioni break e continue	4
	7	Gli operatori logici Operatori di uguaglianza (= =) e di assegnamento (=); Riassunto della programmazione strutturata. I moduli di programma in C	3
	8	Le funzioni della libreria matematica Le funzioni Le definizioni di funzione I prototipi di funzione Lo stack delle chiamate di funzione e i record di attivazione I file di intestazione Invocare le funzioni: chiamata per valore e per riferimento Generazione di numeri casuali	3
	9	Exercise/self-assessment: flowchart development, activity carried out with the assistance of the teacher	4
	10	Esempio: un gioco d'azzardo Le classi di memoria Le regole di visibilità	4
	11	La ricorsione	2
	12	I vettori La dichiarazione dei vettori Esempi sui vettori	3
	13	Exercise and individual study of video material produced by the teacher, to support the theoretical lessons. The video describes the development of a flowchart inspired by exercise 4.40 ' Crescita della popolazione mondiale' of the reference text. The design method respects the principles of structured programming and uses basic control structures of the C programming language - discussion with the teacher.	4
	14	Ulteriori esempi sui vettori	3
	15	Vettori statici ed automatici Passare i vettori alle funzioni, passaggio per riferimento (indirizzo) e per valore, qualificatore 'const' Algoritmi di ordinamento: bubble sort	4
	16	Studio di un caso: calcolare la media, mediana e la moda usando i vettori	4

		La ricerca nei vettori (lineare), algoritmo e programma per la ricerca lineare in un vettore	
	17	La ricerca nei vettori (binaria) Vettori multidimensionali.	4
	18	Manipolazione delle matrici esercitazione sulle matrici bidimensionali (fig.6.22 del testo 4^ed)	3
	19	Exercise and individual study of video material produced by the teacher, to support the theoretical lessons. The video describes the development of a 20-sided die exercise flowchart. The design method respects the principles of structured programming - discussion with the teacher.	3
	20	Esercitazione: Individuazione dei numeri primi. Crivello di Eratostene.	3
	21	Exercise and individual study of video material produced by the teacher, to support the theoretical lessons. The video describes the transition from an algorithm described with a flowchart to the C language code, inspired by the '20-sided die' exercise. The design method respects the principles of structured programming and uses basic control structures of the C programming language - discussion with the teacher.	3
	22	Exercise and individual study of video material produced by the teacher, to support the theoretical lessons. The video describes the development of a flowchart inspired by the exercise 'Craps - a game of chance' in the reference text. The design method respects the principles of structured programming - discussion with the teacher.	3
	23	preparation for exemption	4
	24	autonomous simulation in preparation for the 1st exemption, assisted by the teacher	3
	25	Teacher-assisted exemption self-correction	4
	26	La gerarchia dei dati I file e gli stream Creare un file ad accesso sequenziale Leggere i dati da un file ad accesso sequenziale	4
	27	Leggere i dati da un file ad accesso sequenziale (approfondimenti) Programma per l'interrogazione del credito.	3
	28	I file ad accesso casuale Creare un file ad accesso casuale	4
	29	Scrivere i dati in modo casuale in un file ad accesso casuale Leggere i dati in modo casuale da un file ad accesso casuale	3
	30	Studio di un caso: un programma per l'elaborazione delle transazioni	3
		personal practice, assisted by the teacher; Collective correction exercise carried out. Preparation for the 2nd exemption.	3
		Exercise and Simulation 2nd exemption	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 (vanno bene anche le edizioni successive e precedenti dalla 4^ in poi) Additional text, optional: Kim N. King, Programmazione in C, Maggioli Editore, ISBN 8838785821		

	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 department's e-learning 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			
9 CFU	6 CFU	2 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) and the use of a graphical representation (eg flow chart) 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 Xcode, Visual Studio) 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

A first mid-term evaluation test, with an exempt value, is held close to the week of interruption of lessons, normally located around mid-November. 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.
A second midterm evaluation test, with exempt value, is held immediately before or after the Christmas holidays, depending on the progress of the course. The second test consists of a closed multiple choice test, in particular about 50 questions will be administered and the expected time is normally 30 minutes. Each incorrect answer will be penalized with 0.25/50th of a point. The result obtained is normalized to 30.

The mark resulting from the average of the marks obtained in the two exemption tests is proposed by the teacher to be recorded. The grades from the exemption can only be used in the first exam session (January/February).
The student can, if he prefers, integrate the exam with an oral exam.
The vote is out of thirty.

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 theory test, if taken and passed in the mid-term evaluation, can be used by the student as a theory test in the regular exam.

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	descriptors
< 18 insufficiente	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-di-laurea/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-di-laurea/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.

Link to the course on the department e-learning platform:
https://elearning.di.uniba.it/