



Principali informazioni sull'insegnamento

Denominazione dell'insegnamento	Linguaggi di Programmazione	
Corso di studio	Informatica triennale (L31)(track E-M) – sede di Bari	
Anno Accademico	2025/26	
Crediti formativi universitari (CFU) / European Credit Transfer and Accumulation System (ECTS)	9 CFU	
Settore Scientifico Disciplinare	INFO-01/A	
Lingua di erogazione	Italiano	
Anno di corso	Primo	
Periodo di erogazione	2° semestre, le date esatte sono indicate nel manifesto/regolamento	
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/corsi-di-laurea	

Docente/i	
Nome e cognome	Pasquale Lops
Indirizzo mail	pasquale.lops@uniba.it
Telefono	080-5442276
Sede	Dipartimento di Informatica, Via E. Orabona 4, 70125, Bari. Stanza 759, 7° piano.
Sede virtuale	Piattaforma e-learning UNIBA https://elearning.uniba.it/
Sito web del docente	https://www.uniba.it/it/docenti/lops-pasquale
Ricevimento (giorni, orari e modalità, es. su appuntamento)	Martedì ore 10-12 oppure appuntamento da concordare per e-mail con il docente

Syllabus	
Obiettivi formativi	L'insegnamento si propone di introdurre: ● i fondamenti teorici dei linguaggi di programmazione;



	le tecniche sottostanti la progettazione dei linguaggi di programmazione. Lo studente acquisirà la conoscenza della teoria dei linguaggi formali, sarà in grado di comprendere i meccanismi secondo i quali si implementa un linguaggio di programmazione e saprà associare le operazioni svolte dal compilatore ai fondamenti teorici dei linguaggi formali.
Prerequisiti	Le seguenti conoscenze preliminari facilitano ed accelerano la comprensione degli argomenti dell'insegnamento: - Programmazione: basi della programmazione imperativa, ricorsione, processo di compilazione; - Matematica Discreta: teoria degli insiemi, relazioni, principio di induzione, strutture algebriche elementari (in particolare, concetto di monoide libero), dimostrazioni di tipo induttivo e deduttivo, definizione di funzione iniettiva, suriettiva, biiettiva. - Raccomandazione: al fine di colmare eventuali lacune rispetto ai prerequisiti si suggerisce di consultare il materiale didattico relativo al percorso di "Linguaggi di Programmazione" disponibile sulla piattaforma: https://elearning.uniba.it/
Contenuti di insegnamento (Programma)	La numerazione degli argomenti ne indica l'ordine di presentazione e di studio. Inoltre, l'indice di ogni argomento è utile per ritrovare i relativi contenuti nei testi di riferimento. 1. <u>Introduzione ai linguaggi di programmazione ed ai linguaggi formali</u> Problemi, Macchine di Turing, cenni a calcolabilità e linguaggi di programmazione. Interpretazione e compilazione. Gerarchia di linguaggi di programmazione e di macchine astratte. Aree di ricerca dell'informatica teorica, panoramica su e relazioni tra sintassi e semantica, alberi di derivazione. Regole di produzione, esempi di linguaggi formali. Ore lezione frontale: 9 Ore esercitazione in aula: 1 2. <u>Grammatiche generative</u> Linguaggi formali e monoidi liberi generati da un insieme. Generazione e riconoscimento di linguaggi formali. Grammatiche generative. Esempi di grammatiche generative. Derivazioni. Correttezza di una grammatica. Ore lezione frontale: 9 Ore esercitazione in aula: 1 3. <u>Linguaggi liberi da contesto e linguaggi dipendenti da contesto</u> Definizioni ed esempi di linguaggi liberi da contesto. Definizioni ed esempi di linguaggi dipendenti da contesto. Grammatiche e linguaggi monotoni. Ore lezione frontale: 3 Ore esercitazione in aula: 1 4. <u>Linguaggi liberi da contesto</u> Alberi di derivazione. Principio di sostituzione di sottoalberi. Pumping lemma per i linguaggi liberi da contesto. Ore lezione frontale: 9 Ore esercitazione in aula: 9, comprensive di ore per la preparazione alla prova in itinere 5. <u>Grammatiche e macchine</u> Classificazione delle grammatiche secondo Chomsky. Teorema della Gerarchia di Chomsky. Lemma della stringa vuota. Operazioni sui linguaggi e proprietà di chiusura delle classi di linguaggi rispetto alle operazioni. Ore lezione frontale: 14 Ore esercitazione in aula: 6 6. <u>Automi</u> Automi a stati finiti deterministici e non deterministici. Linguaggi a stati finiti. Ore lezione frontale: 3 Ore esercitazione in aula: 2 7. <u>Linguaggi regolari ed espressioni regolari</u> Definizioni e proprietà delle espressioni regolari. Teorema di Kleene. Pumping Lemma per i linguaggi regolari. Ore lezione frontale: 7



	Ore esercitazione in aula: 10, comprensive di ore per la preparazione alla prova scritta finale. 8. Modello del compilatore Analizzatore lessicale, analizzatore sintattico, analizzatore semantico, generazione e ottimizzazione del codice. Tabella dei simboli. Ore lezione frontale: 2
Testi di riferimento	[Semeraro] G. Semeraro. Elementi di teoria dei linguaggi formali, ilmiolibro.it (2017), http://ilmiolibro.kataweb.it/libro/informatica-e-internet/317883/elementi-di-teoria-dei-linguaggi-formali [Gabbrielli] M. Gabbrielli, S. Martini. Linguaggi di Programmazione, Principi e paradigmi. 2/ed., McGraw-Hill (2011). [Hopcroft] J. E. Hopcroft, R. Motwani, J. D. Ullman. Automi, Linguaggi e calcolabilità. Pearson (2018). Nella sezione seguente è specificata in dettaglio la corrispondenza tra gli argomenti del programma ed i capitoli dei testi di riferimento. Gli studenti che lo desiderano possono ottenere i testi in prestito dalla Biblioteca. Si suggerisce di verificare l'eventuale disponibilità dei testi 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	Si specificano di seguito, per ogni argomento del programma, i capitoli dei testi dai quali studiare. 1. Introduzione ai linguaggi di programmazione ed ai linguaggi formali [Gabbrielli] Capitolo 1; [Semeraro] Capitolo 1. [Hopcroft] Capitolo 8 per approfondimenti sulla Macchina di Turing. 2. Grammatiche generative [Semeraro] Capitoli 1 e 2. 3. Linguaggi liberi da contesto e linguaggi dipendenti da contesto [Semeraro] Capitolo 3. 4. Linguaggi liberi da contesto [Semeraro] Capitolo 4. 5. Grammatiche e macchine [Semeraro] Capitolo 5. 6. Automi [Semeraro] Capitolo 6 [Hopcroft] Capitolo 2 per approfondimenti su equivalenza tra automi deterministici e non deterministici 7. Linguaggi regolari ed espressioni regolari [Semeraro] Capitolo 7 [Hopcroft] Capitolo 3 per approfondimenti su espressioni regolari e per procedura di conversione di automi in espressioni regolari 8. Modello del compilatore [Gabbrielli] Capitoli 2, 3 e 4 per approfondimenti su analisi lessicale e sintattica. Per ogni argomento è inoltre disponibile una dispensa, corrispondente ai contenuti proiettati in aula durante le lezioni, che non è sostitutiva dei testi di riferimento. Elenco del materiale disponibile sulla piattaforma di e-learning ADA https://elearning.uniba.it/: ● dispense in formato pdf degli argomenti del programma; ● esercizi con relativo svolgimento; ● tracce di esempio delle prove d'esame, alcune delle quali con relative soluzioni ● tracce di esempio della prova in itinere.
Organizzazione della didattica	



Ore			
Totali	Didattica frontale	Laboratorio ed esercitazioni	Studio individuale
225 ore	56 ore	30 ore	139 ore
CFU/ETCS			
9 CFU	7 CFU	2 CFU	

Metodi didattici	
	• Lezioni frontali condotte con l'ausilio di dispense proiettate in aula e rese disponibili tramite la piattaforma di e-learning prima delle lezioni; • Svolgimento in aula di due tipologie di esercitazione: (a) esercizi svolti interamente dal docente con indicazione delle soluzioni; (b) esercitazioni guidate in cui gli studenti risolvono insieme al docente quesiti simili a quelli da affrontare nelle prove d'esame. Entrambe le tipologie di esercitazione sono svolte con l'obiettivo di acquisire dimestichezza con i modelli di computazione (grammatiche, automi, ecc.) utili alla classificazione, alla generazione ed al riconoscimento di linguaggi formali.

Risultati di apprendimento previsti	
Conoscenza e capacità di comprensione	• Comprensione della relazione tra problemi, algoritmi, linguaggi formali e linguaggi di programmazione; • Comprensione della Teoria dei Linguaggi Formali; • Conoscenza delle tecniche di analisi e traduzione dei linguaggi di programmazione; • Conoscenza dei fondamenti teorici alla base delle componenti di analisi e traduzione dei linguaggi di programmazione (analizzatore lessicale, analizzatore sintattico).
Conoscenza e capacità di comprensione applicate	• Capacità di classificare un linguaggio formale nella gerarchia di Chomsky; • Capacità di generare, descrivere e riconoscere un linguaggio formale.
Competenze trasversali	Autonomia di giudizio • Capacità di valutare la correttezza della soluzione proposta a problemi relativi alla teoria dei linguaggi formali. Abilità comunicative • Capacità di formalizzare e comunicare la soluzione proposta a problemi relativi alla teoria dei linguaggi formali. Capacità di apprendere in modo autonomo • Capacità di associare un problema pratico ai modelli appresi (ad esempio riconoscimento di stringhe o frasi in linguaggio naturale mediante espressioni regolari).



Valutazione	
Modalità di verifica dell'apprendimento	<u>Prova d'esame</u> L'esame consiste in una prova scritta, nella quale si richiede di rispondere alle seguenti tipologie di quesiti: ● svolgimento di esercizi sulla teoria dei linguaggi formali; ● enunciazione di definizioni sulla teoria dei linguaggi formali; ● dimostrazione di teoremi della teoria dei linguaggi formali; ● quesiti aperti sul modello di compilatore. Regolamentazione della prova d'esame: ● durata della prova: varia tra 60 e 90 minuti in base alla tipologia di quesiti da risolvere; ● materiale consultabile: tavola relativa alle proprietà di chiusura delle classi di linguaggi, elenco delle proprietà delle espressioni regolari; ● la valutazione è espressa in <i>trentesimi</i>; ● comunicazione dei risultati: piattaforma esse3. <u>Prova intermedia</u> È una prova scritta aperta agli studenti in corso, NON obbligatoria, riguardante una specifica parte del programma indicata dal docente in prossimità dell'interruzione delle lezioni stabilita da Regolamento Didattico e Manifesto degli Studi. La prova intermedia prevede: ● esecuzione di esercizi sulla teoria dei linguaggi formali; ● enunciazione di definizioni; ● dimostrazioni di teoremi della teoria dei linguaggi formali. Regolamentazione della prova intermedia: ● durata della prova: come prova d'esame; ● materiale consultabile: nessuno; ● la prova è soggetta ad autovalutazione: il docente fornirà le soluzioni e le indicazioni per assegnare una valutazione ad ogni errore commesso (grave/non grave). La valutazione finale, in base agli errori commessi, potrà essere sufficiente/insufficiente; ● Il superamento della prova intermedia NON esonera lo studente dallo svolgimento di una parte della prova d'esame. L'obiettivo della prova è quello di dare un riscontro allo studente sul proprio livello di comprensione degli argomenti dell'insegnamento. Dopo la prova intermedia, si suggerisce di contattare il docente per un ricevimento in caso di dubbi. <u>Simulazione prova d'esame</u> Al termine delle lezioni si terrà una simulazione della prova d'esame, che consisterà nell'esecuzione di esercizi tratti da prove d'esame precedenti. Gli esercizi saranno svolti in autonomia dagli studenti nel tempo solitamente concesso in sede d'esame in modo da avere un riscontro della propria preparazione <i>prima</i> dell'esame. Al termine della prova il docente mostrerà le relative soluzioni.
Criteri di valutazione	<i>Conoscenza e capacità di comprensione:</i> ○ Capacità di enunciare con rigore definizioni e teoremi della teoria dei linguaggi formali; ○ Capacità di descrivere con precisione e chiarezza le tecniche ed i fondamenti teorici alla base dei meccanismi di analisi e traduzione dei linguaggi di programmazione. <i>Conoscenza e capacità di comprensione applicate:</i> ○ Capacità di applicare i teoremi della teoria dei linguaggi formali alla soluzione di problemi; ○ Capacità di classificare un linguaggio formale nella gerarchia di Chomsky, mediante la costruzione di una dimostrazione formale;



	○ Capacità di generare, descrivere e riconoscere un linguaggio formale, mediante la costruzione di grammatiche, la definizione di espressioni regolari e la progettazione di opportune macchine. <i>Autonomia di giudizio:</i> ○ Modalità adottate per la verifica delle soluzioni proposte (ad esempio, verifica di correttezza di una grammatica). <i>Abilità comunicative:</i> ○ Chiarezza nella descrizione delle soluzioni proposte ai quesiti. <i>Capacità di apprendere:</i> ○ Capacità di astrazione, di ragionamento per analogia e dimostrazione di creatività nella risoluzione dei quesiti.
Criteri di misurazione dell'apprendimento e di attribuzione del voto finale	Ad ogni quesito da svolgere sarà attribuito un punteggio, assicurando che la somma dei punteggi sia pari a 30 (ad esempio, 3 quesiti ad ognuno dei quali sono attribuiti 10 punti). La determinazione del voto attribuito alla risposta data ad ogni quesito tiene conto dei seguenti criteri: 1) la correttezza della soluzione, della dimostrazione formale o della definizione fornita; 2) la completezza della soluzione, della dimostrazione formale o della definizione fornita; 3) la logica seguita dallo studente nel proporre la soluzione o la dimostrazione formale; 4) l'utilizzo di un adeguato formalismo per la descrizione della soluzione, della dimostrazione o della definizione. Per ottenere una valutazione sufficiente, la risposta fornita al quesito deve soddisfare almeno il criterio 1). I voti superiori alla sufficienza sono attribuiti se le risposte fornite soddisfano anche gli altri criteri. Il voto finale è dato dalla somma dei punteggi ottenuti sui singoli quesiti. <u>Prova intermedia</u> La valutazione della prova è espressa in trentesimi. La prova intermedia si intende superata quando il voto conseguito è maggiore o uguale a 16. L'esito della prova intermedia concorre alla valutazione finale come descritto nel riquadro "Modalità di verifica dell'apprendimento". <u>Prova d'esame</u> La valutazione della prova è espressa in trentesimi. La prova d'esame si intende superata quando il voto conseguito è superiore o uguale a 18. La lode è attribuita quando la logica seguita dallo studente nel proporre la soluzione o la dimostrazione formale evidenzia particolari capacità di astrazione, ragionamento per analogia, creatività.
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.uniba.it/ I programmi degli insegnamenti sono disponibili al seguente URL: ● https://elearning.uniba.it/ Le informazioni che tutti gli studenti dovrebbero conoscere sono scritte nei Regolamenti Didattici e Manifesti degli studi disponibili sul sito:



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

Si suggerisce agli studenti di diffidare dalle 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, fissare un incontro con il docente secondo le modalità previste per il ricevimento.

Suggerimenti per una corretta preparazione, in particolare per studenti non frequentanti:

- 1) Studiare gli argomenti teorici nell'ordine suggerito prima di cimentarsi negli esercizi. L'ordine con il quale sono presentati gli argomenti è fondamentale per la comprensione degli stessi. Tentare di risolvere gli esercizi apprendendo solo dalle tracce di esempio e relative soluzioni non consente di acquisire le basi culturali per poter superare l'esame.
Comprendere perché si arriva ad una certa soluzione è molto più importante che svolgere molti esercizi senza comprenderne del tutto le basi teoriche.
- 2) Svolgere gli esercizi, dei quali sono fornite le soluzioni, disponibili su ADA, in modo da confrontare le proprie soluzioni con quelle fornite dal docente.
- 3) Svolgere gli esercizi tratti da prove d'esame per valutare la propria preparazione.
- 4) Contattare il docente per chiedere eventuali chiarimenti e/o verificare le proprie soluzioni agli esercizi svolti in autonomia prima di iscriversi alla prova d'esame.
- 5) In caso di mancato superamento della prova, contattare il docente per la revisione del proprio elaborato e la comprensione degli errori commessi.



Main information on the course

Course name	Programming Languages	
Degree	Computer Science (L31) (track E-M) – Bari	
Academic year	2025/26	
European Credit Transfer and Accumulation System (ECTS), in Italian Crediti Formativi Universitari (CFU)	9 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	INFO-01/A	
Course language	Italian	
Course year	First	
Course period	2nd semester, dates are reported in the Didactic Regulations and Study Prospectus	
Course attendance requirement	Attendance is highly recommended	
Website of the Degree	https://www.uniba.it/it/ricerca/dipartimenti/informatica/didattica/corsi-di-laurea/corsi-di-laurea	

Teacher(s)

Name and Surname	Pasquale Lops
email	pasquale.lops@uniba.it
phone	080-5442276
office	Department of Computer Science, Via E. Orabona 4, 70126, Bari. office n.759, 7th floor.
e-learning platform	e-learning platform https://elearning.uniba.it/
Teacher's homepage	https://www.uniba.it/it/docenti/lops-pasquale
Office hours	Tuesday h 10-12 or students are required to send an email to the instructor to ask for an appointment.

Syllabus

Course goals	The course aims to introduce: - the theoretical foundations of programming languages; - the techniques underlying the design of programming languages. The student will acquire the knowledge of the theory of formal languages, will be able to understand the mechanisms according to which a programming language is implemented and will be able to associate the operations carried out by the compiler to the theoretical foundations of formal languages.
Prerequisites/requirements	The following preliminary knowledge facilitates and accelerates the understanding of the topics of the course: - from "Programming": basics of imperative programming, recursion, compilation process; - from "Discrete Mathematics": set theory, relations, principle of induction, elementary algebraic structures (in particular, concept of monoid and free monoid generated by a set), inductive and deductive proofs, definition of injective, surjective, bijective function. - Recommendation: in order to fill any gaps with respect to prerequisites, it is suggested to consult the teaching materials related to the "Programming Languages" pre-course available on the platform: https://elearning.uniba.it/



Course program	The numbering of topics indicates their order of presentation and study. In addition, the index of each topic is useful for finding the related content in reference texts. <u>1. Introduction to programming languages and formal languages:</u> Problems, Turing machines, computability and programming languages. Interpreters and compilers. Hierarchy of programming languages and abstract machines. Research areas of theoretical computer science, overview of and relationships between syntax and semantics, derivation trees. Production rules, examples of formal languages. Lecture: 9 Hands-on: 1 <u>2. Generative grammars:</u> Formal languages and free monoids generated by a set. Generation and recognition of formal languages. Generative grammars. Examples of generative grammars. Derivations. Correctness of a grammar. Lecture: 9 Hands-on: 1 <u>3. Context-free languages and context-dependent languages:</u> Definitions and examples of context-free languages. Definitions and examples of context-dependent languages. Monotonous grammars and languages. Lecture: 3 Hands-on: 1 <u>4. Context-free languages:</u> Derivation trees. Subtree replacement principle. Pumping lemma for context free languages. Lecture: 9 Hands-on: 9, including hours for preparation for the mid-term test. <u>5. Grammars and machines:</u> Classification of grammars according to Chomsky. Chomsky's Hierarchy Theorem. The empty string lemma. Operations on languages and closure properties of language classes with respect to the operations. Lecture: 14 Hands-on: 6 <u>6. Automata:</u> Deterministic and non-deterministic finite state automata. Finite state languages. Lecture: 3 Hands-on: 2 <u>7. Regular languages and regular expressions:</u> Definitions and properties of regular expressions. Kleene's theorem. Pumping Lemma for regular languages. Lecture: 7 Hands-on: 10, including hours for preparation for the mid-term test. <u>8. Compiler model:</u> Lexical analyzer, syntactic analyzer, semantic analyzer, code generation and optimization. Symbol table. Lecture: 2
Books of reference	[Semeraro] G. Semeraro. Elementi di teoria dei linguaggi formali, ilmiolibro.it (2017), http://ilmiolibro.kataweb.it/libro/informatica-e-internet/317883/elementi-di-teoria-dei-linguaggi-formali



	2. [Gabrielli] M. Gabbrielli, S. Martini. Linguaggi di Programmazione, Principi e paradigmi. 2/ed., McGraw-Hill (2011). 3. [Hopcroft] J. E. Hopcroft, R. Motwani, J. D. Ullman. Automi, Linguaggi e calcolabilità. Pearson (2018). The following section details the correspondence between the program topics and the chapters of the reference books. Students can obtain the books on loan from the Library. It is suggested that students check whether the texts are available through the University Library System https://opac.uniba.it/easyweb/w8018/index.php? and contact the library to arrange borrowing.		
Notes to the books	The chapters of the texts from which to study are specified below for each topic in the program. 1. <u>Introduction to programming languages and formal languages:</u> [Gabbrielli] Chapter 1; [Semeraro] Chapter 1. [Hopcroft] Chapter 8 for further information on the Turing Machine. 2. <u>Generative grammars:</u> [Semeraro] Chapter 1 and 2. 3. <u>Context-free languages and context-dependent languages:</u> [Semeraro] Chapter 3. 4. <u>Context-free languages:</u> [Semeraro] Chapter 4. 5. <u>Grammars and machines:</u> [Semeraro] Chapter 5. 6. <u>Automata:</u> [Semeraro] Chapter 6. [Hopcroft] Chapter 2 for further information on equivalence between deterministic and non-deterministic automata. 7. <u>Regular languages and regular expressions:</u> [Semeraro] Chapter 7. [Hopcroft] Chapter 3 for further information on regular expressions and for the procedure for converting automata into regular expressions. 8. <u>Compiler model:</u> [Gabrielli] Chapters 2, 3 and 4 for further information on lexical and syntactic analysis. A handout is also available for each topic, corresponding to the content projected in the classroom during lectures, which is not a substitute for the reference books. List of materials available on the UNIBA e-learning platform https://elearning.uniba.it: - pdf handouts of the program topics; - exercises with solutions; - samples of the exam tests, some of them with solutions - samples of the mid-term tests.		
Organization of the didactic activities			
Hours			
Total	Lectures	Hands-on	Individual study
225 hours	56 hours	30 hours	139 hours
CFU/ETCS			
9 CFU	7 CFU	2 CFU	
Teaching methods			
	- Lectures given with the help of slides made available through the e-learning platform before lessons; - Two types of exercises carried out in the classroom:		



	(a) exercises carried out by the professor; (b) guided exercises in which students solve, with the professor, problems to be addressed in the exam test. Both types of exercises are carried out with the aim of acquiring familiarity with computation models (grammars, automata, etc.) useful for the classification, generation and recognition of formal languages.
--	--

Expected learning outcomes	
Knowledge and understanding	• Understanding of the relationship between problems, algorithms, formal languages and programming languages; • Understanding of the Theory of Formal Languages; • Knowledge of analysis and translation techniques of programming languages; • Knowledge of the theoretical foundations underlying analysis and translation components of programming languages (scanner, parser).
Applying knowledge and understanding	• Ability to classify a formal language in the Chomsky hierarchy; • Ability to generate, describe and recognize a formal language.
Other skills	Autonomy of judgment • Ability to evaluate the correctness of the proposed solution to problems related to the theory of formal languages. Communication skills • Ability to formalize and communicate the proposed solution to problems related to the theory of formal languages. Ability to learn independently • Ability to associate a practical problem with learned models (e.g. recognition of strings or sentences in natural language using regular expressions).

Assessment	
Assessment methods	<u>Exam test</u> The exam consists of a written test, in which it is required to answer the following types of questions: • carrying out exercises on the theory of formal languages; • enunciation of definitions on the theory of formal languages; • proof of theorems on the theory of formal languages; • open questions about the compiler model. Regulation of the exam: • duration of the test: it varies between 60 and 90 minutes, depending on the type of questions to be solved • material that can be consulted during the test: table relating to the closure properties of languages, list of properties of regular expressions; • the evaluation is expressed in thirtieths; • communication of results: Esse3 platform. <u>Mid-term test</u> It is a written test open to students in progress, not mandatory, concerning a specific part of the program indicated by the teacher near the interruption of lessons



	established by the Didactic Regulations and Study Prospectus. The mid-term test includes: ● carrying out exercises on the theory of formal languages; ● enunciation of definitions; ● proofs of theorems on the theory of formal languages Regulation of the mid-term test: ● duration of the test: same as the exam test; ● material that can be consulted during the test: nothing; ● passing the mid-term test exempts the student from carrying out a part of the exam test (i.e., the topics of the mid-term test), provided that the exam test is attempted in the first call; ● the final mark will be obtained as the average of the mark achieved in the midterm test and the mark achieved in the first call exam test; ● the aim of the mid-term test is to give feedback to the student on the level of ● understanding of the topics of the course. <u>Exam test simulation</u> At the end of the lessons there will be a simulation of the exam, which will consist in the execution of exercises taken from previous exam tests. The exercises will be carried out independently by the students in the time usually allowed during the exam in order to have a feedback of their preparation before the exam. At the end of the test the instructor will show the solutions.
Evaluation criteria	● <i>Knowledge and understanding:</i> ○ Ability to rigorously enunciate definitions and theorems of the theory of formal languages; ○ Ability to describe with accuracy and clarity the techniques and theoretical foundations underlying the mechanisms of analysis and translation of programming languages. ● <i>Applied knowledge and understanding:</i> ○ Ability to apply the theorems of the theory of formal languages to problem solving; ○ Capacità di classificare un linguaggio formale nella gerarchia di Chomsky, mediante la costruzione di una dimostrazione formale ○ Ability to classify a formal language in the Chomsky hierarchy, by constructing a formal proof; ○ Ability to generate, describe and recognize a formal language, through the construction of grammars, the definition of regular expressions, and the design of appropriate machines. ● <i>Autonomy of judgment:</i> ○ Methods adopted for the verification of the proposed solutions (for example, verification of correctness of a grammar). ● <i>Communication skills:</i> ○ Clarity of the proposed solutions to the questions. ● <i>Learning skills:</i> ○ Ability to abstract, reasoning by analogy and demonstration of creativity in solving questions.
Measurements and final grade	Each question to be carried out will be assigned a mark, ensuring that the sum of the marks is equal to 30 (for example, 3 questions to each of which 10 points are attributed). The determination of the mark assigned to the answer given to each question takes into account the following criteria: 1) the correctness of the solution, formal proof or definition provided; 2) the completeness of the solution, formal proof or definition provided; 3) the logic followed by the student in proposing the solution or formal proof; 4) the use of an appropriate formalism for the description of the solution, proof or definition. To pass the exam test or the mid-term test, the student must be able to propose a



	solution that satisfies at least criterion 1). Marks above the minimum are awarded to students who propose solutions that also satisfy the other criteria. The final mark is given by the sum of the marks obtained on the individual questions. Mid-term test The evaluation of the mid-term test is expressed over a 30-point scale. The mid-term test is passed when the mark obtained is greater than or equal to 16. The outcome of the mid-term test contributes to the final evaluation as described in the box "Assessment methods". Exam test The evaluation of the exam test is expressed over a 30-point scale. The exam test is passed when the mark obtained is greater than or equal to 18. Attribution of honors: Honors are awarded when the logic followed by the student in proposing the solution or formal proof highlights particular capacities of abstraction, reasoning by analogy, creativity.
Further information	It is suggested that students rely exclusively on the information/communication 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 faculty members of the relevant subjects: • 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 programs of the courses are available at the following URL: • https://elearning.uniba.it/ Information that all students need to know is available in the Didactic Regulations and Study Prospectus available on the website: • https://www.uniba.it/it/ricerca/dipartimenti/informatica/didattica/corsi-di-laurea/corsi-di-laurea <i>Students are suggested to be wary of information and materials circulating on unofficial sites or social groups, as they are often found to be unreliable, incorrect or incomplete. For any doubts, arrange a meeting with the teachers in accordance with the reception arrangements.</i> Link to the course on the department's ADA e-learning platform: https://elearning.uniba.it/ <u>Tips for proper preparation, especially for non-attending students:</u> 1) Study the theoretical topics in the suggested order before engaging in the exercises. The order in which the topics are presented is fundamental for understanding them. Trying to solve the exercises by learning only from the example exam tests and related solutions does not allow the student to acquire the cultural bases to be able to pass the exam. Understanding the reasons that led to a certain solution is much more important than doing many exercises without fully understanding the theoretical bases. 2) Carry out the exercises, whose solutions are provided by the instructor on the e-learning platform, in order to compare your own solutions with those provided by the teacher. 3) Carry out the exercises taken from exam tests to evaluate your preparation. 4) Contact the instructor for any clarifications and / or verify your solutions to the exercises carried out independently before registering for the exam. 5) In case of failure to pass a test, contact the instructor for the revision of your test and the understanding of the errors made.