Principali informazioni sull'insegnamento

Denominazione dell'insegnamento	**Calcolabilità e Complessità (M-Z)**	
Corso di studio	Corso di Laurea Triennale in Informatica	
Anno Accademico	2023/24	
Crediti formativi universitari (CFU) / European Credit Transfer and Accumulation System (ECTS)	6 CFU	
Settore Scientifico Disciplinare	INF/01	
Lingua di erogazione	Italiano	
Anno di corso	Secondo	
Periodo di erogazione	2^ semestre; le date esatte sono riportate 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/informatica-270/laurea-triennale-in-informatica-d.m.-270-1	

Docente/i

Nome e cognome	Emanuele Covino
Indirizzo mail	emanuele.covino@uniba.it
Telefono	0805442142
Sede	Dipartimento di Informatica, Via Orabona 4, 70125, Bari. Stanza n.561, 5^ piano.
Sede virtuale	Piattaforma ADA – https://elearning.di.uniba.it/ Pagina del corso – http://www.di.uniba.it/~covino → corsi → Calcolabilità e complessità
Sito web del docente	http://www.di.uniba.it/~covino
Ricevimento (giorni, orari e modalità, es. su appuntamento)	Martedì dalle 11 alle 12; in coda a ogni lezione.

Syllabus	
Obiettivi formativi	Il corso si propone di analizzare la definizione del concetto di calcolo, dei relativi modelli, dei concetti di calcolabile e non calcolabile, anche in funzione della complessità dei problemi. Inoltre, di fornire la conoscenza (e la capacità di definizione) di grammatiche che individuano linguaggi di programmazione, e la capacità di definire le opportune macchine astratte per la soluzione efficiente di problemi di calcolo. Si analizzeranno le caratteristiche di problemi e linguaggi decidibili e i limiti alla decidibilità. Lo studente dovrà acquisire competenze relative alla progettazione, implementazione, funzionamento e valutazione dei modelli di calcolo astratti.
Prerequisiti	Le seguenti conoscenze preliminari facilitano ed accelerano la comprensione degli argomenti dell'insegnamento: • da Programmazione: programmazione imperativa; calcolo ricorsivo e iterativo; • da Matematica Discreta: teoria degli insiemi, funzioni e relazioni, principio di induzione, strutture algebriche, dimostrazioni di tipo induttivo; • da Linguaggi di Programmazione: sintassi e semantica dei linguaggi di programmazione. •
Contenuti di insegnamento (Programma)	**Introduzione.** (2 ore) Automi, computabilità e complessità. Terminologia e notazione matematica. Definizioni, teoremi dimostrazioni. Tipi di dimostrazioni. **I. Automi e linguaggi** (10 ore) <u>Linguaggi regolari.</u> Definizione di automa finito deterministico. Definizione di computazione. Progettare automi finiti. Le operazioni regolari. Definizione di automa finito non deterministico. Equivalenza tra NFA e DFA. Chiusura rispetto alle operazioni regolari. Definizione di espressione regolare. Equivalenza tra automi finiti, espressioni regolari, grammatiche di tipo 3. Linguaggi non regolari: il pumping lemma per i linguaggi regolari. Esercizi. <u>Linguaggi context-free.</u> Definizione di grammatiche context-free. Grammatiche ambigue. Forma normale di Chomsky. Automi a pila. Equivalenza fra automi a pila e grammatiche context-free. Linguaggi non context-free: il pumping lemma per i linguaggi context-free. Esercizi. **II. Teoria della calcolabilità** (20 ore) <u>La tesi di Church-Turing.</u> Definizione di macchina di Turing. Varianti: macchine multinastro, multitraccia, non deterministiche. Equivalenza fra i modelli. Definizione di algoritmo. <u>Decidibilità</u>. Problemi e linguaggi decidibili. Limiti della decidibilità. Diagonalizzazione. Problemi semidecidibili. Un linguaggio non Turing-riconoscibile. **III. Teoria della complessità** (20 ore) <u>Complessità temporale.</u>

	Misure di complessità. Notazioni, analisi degli algoritmi, relazioni di complessità fra modelli. Complessità temporale: le classi P e NP, il problema P=NP. NP-completezza. Il teorema di Cook. Problemi NP-completi. Complessità spaziale. Il teorema di Savitch. Le classi PSPACE e NPSPACE. PSPACE-completezza. **IV. Esercitazioni** (10 ore) Utilizzando il software JFLAP (http://www.jflap.org/), definire automi finiti deterministici e non deterministici, automi push-down deterministici e non deterministici, macchine di Turing deterministiche e non deterministiche, macchine di Turing multinastro; progettare e implementare una macchina universale.
Testi di riferimento	Testo principale: (1) M. Sipser - Introduzione alla teoria della computazione. Eventuali approfondimenti: (2) G. Ausiello, F. D'Amore, G. Gambosi - Linguaggi, modelli, complessità. (3) Aho, Lam, Sethi, Ullman – Compilatori: principi, tecniche e strumenti. 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	Le slides utilizzate a lezione fungono da supporto didattico: sintetizzano i contenuti del testo di riferimento e riportano esercizi e attività svolte in aula. Tutto il materiale utilizzato durante le lezioni sarà reso disponibile (e aggiornato in tempi ragionevoli) sulla piattaforma ADA del dipartimento e sulla pagina del corso (vedi sopra 'sede virtuale'). La fonte principale di studio è comunque costituita dal testo di riferimento.

Organizzazione della didattica

Ore

Totali	Didattica frontale	Pratica (laboratorio, progetto, esercitazione, altro)	Studio individuale
150 ore	32 ore	30 ore	88 ore

CFU/ETCS

6 CFU	4 CFU	2 CFU	

Metodi didattici

	Lezioni frontali di teoria supportate da slide, esercitazioni guidate in aula, esercitazioni e/o ricerche da svolgere singolarmente o in piccoli gruppi di studio.

Risultati di apprendimento previsti	
Conoscenza e capacità di comprensione	Lo studente acquisirà una conoscenza di base del concetto di calcolo, dei relativi modelli, della misura di complessità del calcolo; dei concetti di decidibilità, semi decidibilità e indecidibilità.
Conoscenza e capacità di comprensione applicate	Lo studente acquisirà competenze relative alla progettazione, implementazione, funzionamento e valutazione dei modelli di calcolo astratti.
Competenze trasversali	**Autonomia di giudizio** Lo studente dovrà dimostrare di aver acquisito autonomia di giudizio e capacità di valutazione nell'ambito dei temi della calcolabilità e della complessità computazionale dei problemi. **Capacità di apprendere in modo autonomo** Lo studente svilupperà capacità di apprendimento per orientarsi nelle problematiche che si presentano durante lo sviluppo e l'analisi di modelli di calcolo e della relativa complessità. **Abilità comunicative** Lo studente relazionerà in maniera appropriata in riferimento ai principi della calcolabilità e della complessità dei problemi, mostrando di aver sviluppato capacità di intraprendere in autonomia ulteriori approfondimenti su tali argomenti.

Valutazione	
Modalità di verifica dell'apprendimento	Prova scritta in itinere: non obbligatoria, da tenersi durante l'interruzione delle lezioni prevista, che verte sugli argomenti trattati nella prima parte del corso e consiste in tre/quattro esercizi. Il superamento della prova in itinere e/o i risultati delle esercitazioni pratiche attribuiscono una premialità sul voto finale (da 0 a 4 punti). Prova di laboratorio: si svolge nei laboratori didattici del dipartimento e ha una durata di un'ora. Durante la prova di laboratorio lo studente dovrà utilizzare il software Jflap per definire un automa che riconosca un dato linguaggio o calcoli una data funzione. Contribuisce a formare il voto finale (da 0 a 6 punti). Prova scritta: si svolge nelle aule del dipartimento e ha una durata di due ore. Consiste in sei domande e/o esercizi riguardanti il programma svolto. Contribuisce a formare il voto finale (da 0 a 24 punti, si ritiene superata con 14 punti). Per sostenere la prova scritta occorre aver superato la prova di laboratorio. Lo studente potrà decidere di sostenere la prova di laboratorio e quella scritta in due appelli differenti. In caso di mancato superamento della prova scritta, la prova di laboratorio resta valida per l'intero anno accademico.
Criteri di valutazione	Conoscenza e capacità di comprensione: Capacità di comprendere le domande formulate per la prova scritta e rispondere in maniera pertinente ed esaustiva.

	Capacità di comprendere le linee guida per lo svolgimento delle esercitazioni. Conoscenza e capacità di comprensione applicate: Conoscenza esaustiva degli argomenti oggetto del corso e loro utilizzo nello svolgimento di esercizi oggetto della prova scritta. Autonomia di giudizio: Lo studente dovrà dimostrare di aver acquisito autonomia di giudizio e capacità di valutazione nell'ambito dei temi della calcolabilità e della complessità computazionale dei problemi di calcolo. Abilità comunicative: Capacità di rispondere ai quesiti formulati per la prova scritta in maniera corretta, esaustiva e utilizzando correttamente il linguaggio tecnico-matematico. Capacità di apprendere: Comprensione dei contenuti del corso e capacità utilizzare i concetti appresi nello svolgimento di esercizi e analisi dei problemi proposti.
Criteri di misurazione dell'apprendimento e di attribuzione del voto finale	Il voto finale è la somma dei tre voti conseguiti nella prova in itinere, laboratorio e scritto.
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.

Main information on the course

Course name	**Calcolabilità e Complessità (M-Z)**
Degree	Informatica (first-level degree in Computer Science)
Academic year	2023-2024
European Credit Transfer and Accumulation System (ECTS), in Italian Crediti Formativi Universitari (CFU)	6 CFU
Settore Scientifico Disciplinare	INF/01
Course language	English
Year	Second
Semester	Second Semester
Attendance's rules	It is highly recommended to attend classes
Web site	https://www.uniba.it/it/ricerca/dipartimenti/informatica/didattica/corsi-di-laurea/informatica-270/laurea-triennale-in-informatica-d.m.-270-1

Teacher(s)

Name and Surname	Emanuele Covino
Email	emanuele.covino @uniba.it
Phone	0805442142
Office	Dipartimento di Informatica, Via Orabona 4, 70125, Bari. Stanza n.561, 5th floor
e-learning platform	Piattaforma ADA - https://elearning.di.uniba.it/
Teacher's homepage	http://www.di.uniba.it/~covino → corsi → Calcolabilità e complessità
Office hours	Tuesday from 11 to 12; at the end of each lecture.

Syllabus

Course goals	The course will analyze the definition of computation, of the related models, of what is computable and what is not, also depending on the complexity of problems. Moreover, the course will teach how to define grammars for programming languages and how to define abstract machines for the feasible solution of computational problems. Decidable and semi-decidable problems will be introduced, together with bounds to decidability. The student will learn how to design, to implement and to evaluate computational models.
Prerequisites/requirements	The following preliminary knowledges will help and speed-up the comprehension of the topics of each lecture:

	• from Programmazione: imperative programming; recursive and iterative computation; • from Matematica Discreta: set theory, functions and relations, induction principle, algebraic structures, inductive proofs; • from Linguaggi di Programmazione: syntax and semantic of programming languages.
Course program	**Introduction** (2 hours) Automata, computability, and complexity. Mathematical notions and terminology. Definitions, theorems, and proofs. Types of proof. **I. Automata and languages** (10 hours) Regular Languages. Formal definition of a finite automaton. Formal definition of computation. Designing finite automata. The regular operations. Formal definition of a nondeterministic finite automaton. Equivalence of NFAs and DFAs. Closure under the regular operations. Formal definition of a regular expression. Equivalence with finite automata. Nonregular Languages. The pumping lemma for regular languages. Exercises. Context-Free Languages. Formal definition of a context-free grammar. Examples of context-free grammars. Ambiguity. Chomsky normal form. Pushdown Automata. Examples of pushdown automata. Equivalence with context-free grammars. Non-Context-Free Languages. The pumping lemma for context-free languages. Deterministic Context-Free Languages. Properties of DCFLs. Relationship of DPDAs and DCFGs. Exercises. **II. Computability Theory** (20 hours) The Church–Turing Thesis Formal definition of a Turing machine. Variants of Turing Machines: multitape Turing machines, nondeterministic Turing machines. Equivalence within and with other models. The Definition of Algorithm. Decidability Decidable Languages and problems. Undecidability. The diagonalization method. An undecidable language. A Turing-unrecognizable language. Reducibility. **III. Complexity Theory** (20 hours) Time complexity Measuring Complexity. Big-O and small-o notation. Analyzing algorithms. Complexity relationships among models. The Class P: polynomial time. The Class NP. The P versus NP question. NP-completeness. The Cook–Levin theorem. Space Complexity Savitch's Theorem. The class PSPACE and NPSPACE. PSPACE-completeness. Exercises. **IV. Practice** (10 hours) With the JFLAP software (http://www.jflap.org/), the student will define deterministic and non-deterministic finite automata's, push-down automata's, deterministic and non-deterministic Turing machines, multitape Turing machines; design and realize a universal Turing machine.
Books of reference	Main book: (1) M. Sipser - Introduzione alla teoria della computazione.

	More books: (2) G. Ausiello, F. D'Amore, G. Gambosi - Linguaggi, modelli, complessità. (3) Aho, Lam, Sethi, Ullman – Compilatori: principi, tecniche e strumenti. The students can borrow the previous texts from the Library, using https://opac.uniba.it/easyweb/w8018/index.php?
Notes to the books	The slides used during each lecture summarize the content of the book of reference, and show exercises and classroom activities. Everything is uploaded in the ADA platform and in the course website (see above, "teacher's homepage"). The main reference is anyhow the book (1).

Organization of the didactic activities

Hours			
Total	Lectures	Practice sessions	Individual study
150 hours	32 hours	30 hours	88 hours
CFU/ETCS			
4 CFU	4 CFU	2 CFU	

Teaching methods

Lectures supported by slides that also show examples to better illustrate the discussed topics. Exercises and discussions. Group researches about the topics of the course.

Expected learning outcomes

Knowledge and understanding	The student will have a basic knowledge of the definition of computation, of the related models, and of the measures of complexity; he will have knowledge of the idea of decidability, semi-decidability, and undecibability.
Applying knowledge and understanding	The student will be able to design, implement, and evaluate abstract computational models.
Other skills	**Making judgements** The student will achieve the ability to integrate knowledge, handle complexity and make decisions about the topics of computability and complexity. **Learning skills** The student will develop the ability of autonomous learning in order to navigate into the design and analysis of computational model and the related complexity. **Communication** Students are encouraged to work in groups and to illustrate the outcome of exercises, with the goal of developing their communication and collaboration skills. Students are also required to develop projects/case studies that can be presented in class; this allows the student to demonstrate his/her communicative abilities.

Assessment

Assessment methods	Partial written test: non mandatory, it takes place during the lectures' break; it's about the topics of the first part of the course, and it consists in four exercises. The evaluation is from 0 to 4 points. Lab test: the student will write (within one hour, usign JFLAP) an automata that decides a language or computes a given function. The evaluation is from 0 to 6 points. Written test: the student has two hours to provide the solution of six questions and/or exercises. The evaluation is from 0 to 24 points, with a minimun threshold of 14. The student can take the written test only if he has already an evaluation for the la test. The student can take the lab and the written test in two different session.
Evaluation criteria	In order to assess the knowledge acquired by the student, and also the acquired abilities to make judgments and to communicate as well as their learning skills, the written test (individual) is evaluated on the basis of the correctness of the answers provided and, considering the open answers, his/her ability to synthesise, the clarity of the presentation, the examples provided to better illustrate the written text, the ability to make comparisons and provide their own critical views.
Measurements and final grade	The final evaluation is the sum of partial, lab, and written test.
Further information	The students are required to trust news and communications that can be found in the official website of the Dipartimento di Informatica only, as well as on social groups founded and handled by the lecturers of each course: ● 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/ Topics of each course con be found here: ● https://programmi.di.uniba.it/ Rules and regulations can be found here: ● https://www.uniba.it/it/ricerca/dipartimenti/informatica/didattica/corsi-di-laurea/corsi-di-laurea We suggest to avoid any non-official information and study material; they are always untrustable, wrong, incomplete. Ask your lecturer.