Principali informazioni sull'insegnamento

Denominazione dell'insegnamento	**Programmazione II (A-L)**
Corso di studio	Informatica e Tecnologie per la Produzione del Software
Anno Accademico	2024/25
Crediti formativi universitari (CFU) / European Credit Transfer and Accumulation System (ECTS)	09 CFU
Settore Scientifico Disciplinare	IINF-05/A - Sistemi di elaborazione delle informazioni
Lingua di erogazione	Italiano
Anno di corso	Secondo
Periodo di erogazione	1^ 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/corsi-di-laurea

Docente

Nome e cognome	Donato Malerba
Indirizzo mail	donato.malerba@uniba.it
Telefono	080 5443269
Sede	Dipartimento di Informatica, Via Orabona 4, 70125, Bari. Stanza n.509, V piano.
Sede virtuale	Piattaforma ADA - https://elearning.uniba.it/course/view.php?id=2803
Sito web del docente	https://www.uniba.it/it/docenti/malerba-donato
Ricevimento (giorni, orari e modalità, es. su appuntamento)	Mercoledì 11.00-13.00 o su appuntamento

Syllabus

Obiettivi formativi	Acquisizione di conoscenze e competenze sulla programmazione Orientata agli Oggetti (Object-Oriented) nel linguaggio JAVA. Partendo dal paradigma di

	programmazione imperativo appreso nel primo anno del corso di laurea, vengono introdotti i concetti di classe e oggetto ed i concetti di ereditarietà e polimorfismo, e si approfondiscono gli aspetti relativi al trattamento delle eccezioni, all'identificazione di tipo al run-time (RTTI), alla programmazione generica, al sistema I/O. Sono altresì analizzate le collezioni messe a disposizione dal linguaggio JAVA, analizzando peculiarità e differenze (anche in termini di efficienza) delle possibili diverse implementazioni. Attraverso le attività di laboratorio si consolidano le competenze di programmazione già acquisite nel primo anno del corso di laurea.
Prerequisiti	Dall'insegnamento di Programmazione: le basi della programmazione imperativa, ricorsione, compilazione e interpretazione; Dall'insegnamento di Architetture degli Elaboratori e SO: lo stack delle chiamate, la gestione della memoria, processi e thread; Dall'insegnamento di Laboratorio di Informatica: sviluppo di programmi, programmazione modulare, testing, capacità di debugging.
Contenuti di insegnamento (Programma)	1. Riepilogo sui concetti di programmazione di base (7 ore) - Introduzione ai computer e a JAVA (cap. 1) - Concetti e nozioni di base di programmazione (cap. 1 e 2) - Flussi di controllo: la selezione e i cicli (cap. 3 e 4) 2. I metodi: concetti di base (cap. 5) (2 ore) - Definizione e invocazione di metodi - Linee guida sulla scrittura dei metodi 3. Gli array (cap. 6) (5 ore) - Concetti di base sugli array - Utilizzare gli array nei metodi - Ordinamento e ricerca con gli array - Array multidimensionali 4. Ricorsione (cap. 7) (2 ore) - Le basi della ricorsione - Programmare utilizzando la ricorsione 5. La programmazione a oggetti (27 ore) - Definizione di classi (Cap. 8) - Informazion Hiding e incapsulamento (Cap. 8) - Oggetti e riferimenti (Cap. 8) - Costruttori (Cap. 9) - Variabili e metodi statici (Cap. 9) - Overloading (Cap. 9) - Enumerazione come classi (Cap. 9) - La strutturazione in package (Cap. 9) - Rappresentazione in UML delle classi e delle loro relazioni (Cap. 9) - Concetti di base sull'ereditarietà (Cap. 10) - Incapsulamento ed ereditarietà (Cap. 10) - Polimorfismo (Cap. 11) - Classi astratte (Cap. 11) - Interfacce (Cap. 11) - Rappresentazione in UML (Cap. 11) - Programmazione generica (Cap. 12) 6. La gestione delle eccezioni (Cap. 13) (3 ore) - Concetti di base sulle eccezioni - Definizione di nuove eccezioni - Approfondimenti sulle classi di eccezioni 7. Stream e I/O da file (cap. 14) (4 ore)

	- Introduzione ai flussi dati e all'I/O su file - I/O con file di testo - Tecniche generiche per la gestione dei file - I/O su file binari 8. Strutture dati dinamiche (cap. 15) (3 ore) - Liste concatenate, pile e code - Tabelle Hash - Insiemi - Alberi 9. Collezioni, mappe e iteratori (3 ore) - Le collezioni in JAVA (Cap. 12 e 16) - Iteratori (Cap. 16) Esercitazioni guidate e laboratorio (30 ore) • L'ambiente Eclipse. • Semplici esempi di programmi Java; • Programmi con array in Java: • Classe OrdinaArray. Algoritmi di ordinamento bubblesort, insertionsort, selectsort, shellsort, quicksort, mergesort; La classe Arrays; • Esempi di classi specifiche di applicazioni; ordinamento di array di oggetti; ricerca in array di oggetti; • Realizzazione di un dizionario; • Composizione di classi; • Ereditarietà; • Classi astratte e interfacce; • Interfaccia del Dizionario; • Dizionario generico con eccezioni; • I/O e serializzazione.
Testi di riferimento	• Programmazione di base e avanzata con Java, terza edizione, di Walter Savitch e Daniela Micucci, editore Pearson, ISBN: 9788891916020 (capitoli 1-16, Appendici). Gli studenti che lo desiderano possono ottenere il testo 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	Il testo di riferimento è corredato dal materiale proiettato dal docente durante le lezioni e le attività di laboratorio che è reso disponibile disponibile sulla piattaforma e-learning di UNIBA. Sulla piattaforma elearning sono rese disponibili anche alcune tracce di prove scritte di esami, con esempi di tracce svolte.

Organizzazione della didattica

Ore

Totali	Didattica frontale	Laboratorio/esercitazione	Progetto	Studio individuale
225 ore	56 ore	30 ore	0 ore	139 ore

CFU/ETCS

9 CFU	7 CFU	2 CFU	0 CFU	

Metodi didattici	
	Le 86 ore previste per l'insegnamento sono ripartite come segue: 56 ore di didattica frontale (in presenza) con lezioni di teoria; 30 ore di laboratorio (in presenza) con attività pratiche sotto supervisione del docente.

Risultati di apprendimento previsti	
Conoscenza e capacità di comprensione	○ Acquisizione di conoscenze relative al paradigma orientato agli oggetti. ○ Comprensione delle scelte di modellazione object-oriented.
Conoscenza e capacità di comprensione applicate	○ Consolidamento delle capacità di programmazione; ○ Capacità di realizzazione di un semplice programma in un linguaggio orientato agli oggetti mediante utilizzo di un ambiente di sviluppo professionale.
Competenze trasversali	**Autonomia di giudizio** ○ Gli studenti sono in grado di apprezzare la realizzazione di software basati sui principi del paradigma orientato agli oggetti, nonché le potenzialità offerte dal linguaggio Java nella programmazione a oggetti. ○ L'autonomia di giudizio viene acquista attraverso lo studio e le attività laboratoriali. ○ Il raggiungimento dell'adeguata autonomia è verificato attraverso l'esame finale di profitto sia per la teoria che per la pratica. **Abilità comunicative** ○ Gli studenti sono in grado di esporre le tematiche incluse nel programma del corso mediante il lessico specifico della disciplina, e sono in grado di spiegare in modo chiaro e privo di ambiguità a interlocutori specialisti le loro scelte nello sviluppo di semplici programmi in Java. **Capacità di apprendere in modo autonomo** ○ Lo studente svilupperà capacità di apprendere e di orientarsi agilmente nelle problematiche che si presentano durante lo sviluppo di software realizzato in Java coerentemente con i principi del paradigma Object-Oriented.

Valutazione	
Modalità di verifica dell'apprendimento	**Prova d'esame**: L'esame prevede un'unica prova organizzata in due parti: domande di teoria e prova pratica di laboratorio. Le domande di teoria (aperte e/o chiuse) mirano a verificare le capacità di presentazione di concetti di programmazione a oggetti e sono relative a argomenti

	illustrati a lezione e presenti nel sillabo. La prova pratica di laboratorio mira a verificare le capacità di programmazione in Java di semplici applicazioni di cui si forniscono specifiche di massima. I programmi richiedono conoscenze di Java illustrate a lezione e utilizzate nelle attività pratiche di laboratorio. L'esame si svolge in laboratorio nell'arco di tre ore: si stima che due ore debbano essere dedicate alla prova pratica, mentre un'ora debba essere dedicata alle domande di teoria. È responsabilità dello/a studente/studentessa gestire al meglio il tempo assegnato. L'elaborato della prova pratica è dato da un file jar eseguibile, mentre l'elaborato della prova scritta è un file in formato MS Word o pdf. A ognuna delle due parti assegnate sono assegnati 33 punti. La sufficienza (18/33) dev'essere raggiunta su entrambe le parti della prova e il voto finale è il risultato della media dei punteggi conseguiti a ognuna delle due parti. Un punteggio superiore a 30 corrisponde a un trenta con lode. A metà corso è prevista la verifica in itinere a carattere non esonerante e per valutare l'andamento del corso.
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 attività di laboratorio e del progetto **Conoscenza e capacità di comprensione applicate:** ○ Conoscenza esaustiva degli argomenti oggetto del corso e loro utilizzo nello svolgimento di esercizi oggetto della prova scritta, trattazione delle questioni teoriche e realizzazione del progetto **Autonomia di giudizio:** ○ Capacità dello studente di correggere e validare il corretto funzionamento dei programmi sviluppati. **Abilità comunicative:** ○ Capacità di rispondere ai quesiti formulati per la prova scritta in maniera corretta, esaustiva e utilizzando appropriatamente il linguaggio tecnico **Capacità di apprendere:** ○ Comprensione dei contenuti del corso e capacità utilizzare i concetti appresi nello svolgimento di esercizi e sviluppo del progetto di software in Java
Criteri di misurazione dell'apprendimento e di attribuzione del voto finale	Il voto finale sarà attribuito sulla base della valutazione comparativa di tutti i criteri di valutazione sopra citati
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/course/view.php?id=2803 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, poiché spesso sono risultate non affidabili, non corrette o incomplete.

Link al corso sulla piattaforma e-learning ADA:
https://elearning.uniba.it/course/view.php?id=2803

Main information on the course

Course name	**Programmazione II (A-L)**	
Degree	Informatica e Tecnologie per la Produzione del Software	
Academic year	2024/25	
European Credit Transfer and Accumulation System (ECTS), in Italian Crediti Formativi Universitari (CFU)		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	IINF-05/A - Sistemi di elaborazione delle informazioni	
Course language	Italian	
Course year	Second	
Course period	First Semester - exact dates can be found in the didactic regulations	
Course attendance requirement	None, but it is highly recommended to attend classes	
Website of the Degree	https://www.uniba.it/it/ricerca/dipartimenti/informatica/didattica/corsi-di-laurea/corsi-di-laurea	

Teacher(s)

Name and Surname	Donato Malerba
email	donato.malerba@uniba.it
phone	080 5443269
office	Dipartimento di Informatica, Via Orabona 4, 70125, Bari. Room n.508, 5th floor
e-learning platform	Platform ADA - https://elearning.uniba.it/
Teacher's homepage	https://www.uniba.it/it/docenti/malerba-donato
Office hours	Wednesday 11:00-13:00 or on appointment

Syllabus

Course goals	Acquisition of knowledge and skills in Object-Oriented programming in the JAVA language. Building on the imperative programming paradigm learned in the first year of the degree course, the concepts of classes and objects are introduced, along with the concepts of inheritance and polymorphism. The course delves into aspects related to exception handling, runtime type identification (RTTI), generic programming, and the I/O system. The collections provided by the JAVA language are also analyzed, focusing on their peculiarities and differences (including efficiency considerations) among the various possible implementations. Through practice sessions, the programming skills acquired during the first year of the degree course are further consolidated.
Prerequisites/requirements	The following preliminary knowledge facilitates and accelerates the understanding of the course topics: From the course on COMPUTER PROGRAMMING: the fundamentals of imperative programming, recursion, compilation, and interpretation; From the course on COMPUTER ARCHITECTURE AND OPERATING SYSTEMS: the call stack, memory management, processes, and threads; From the course on COMPUTER SCIENCE LABORATORY: program development, modular programming, testing, and debugging skills.
Course program	

1. **Review of Basic Programming Concepts (7 hours)**
   - Introduction to computers and JAVA (Chapter 1)
   - Basic programming concepts and notions (Chapters 1 and 2)
   - Control flow: selection and loops (Chapters 3 and 4)
2. **Methods: Basic Concepts (Chapter 5) (2 hours)**
   - Definition and invocation of methods
   - Guidelines for writing methods
3. **Arrays (Chapter 6) (5 hours)**
   - Basic concepts of arrays
   - Using arrays in methods
   - Sorting and searching with arrays
   - Multidimensional arrays
4. **Recursion (Chapter 7) (2 hours)**
   - Basics of recursion
   - Programming using recursion
5. **Object-Oriented Programming (27 hours)**
   - Class definition (Chapter 8)
   - Information hiding and encapsulation (Chapter 8)
   - Objects and references (Chapter 8)
   - Constructors (Chapter 9)
   - Static variables and methods (Chapter 9)
   - Method overloading (Chapter 9)
   - Enumerations as classes (Chapter 9)
   - Package structuring (Chapter 9)
   - UML representation of classes and their relationships (Chapter 9)
   - Basic concepts of inheritance (Chapter 10)
   - Encapsulation and inheritance (Chapter 10)
   - Polymorphism (Chapter 11)
   - Abstract classes (Chapter 11)
   - Interfaces (Chapter 11)
   - UML representation (Chapter 11)
   - Generic programming (Chapter 12)
6. **Exception Handling (Chapter 13) (3 hours)**
   - Basic concepts of exceptions
   - Defining new exceptions
   - In-depth study of exception classes
7. **Streams and File I/O (Chapter 14) (4 hours)**
   - Introduction to data streams and file I/O
   - I/O with text files
   - General techniques for file management
   - I/O with binary files
8. **Dynamic Data Structures (Chapter 15) (3 hours)**
   - Linked lists, stacks, and queues
   - Hash tables
   - Sets
   - Trees
9. **Collections, Maps, and Iterators (3 hours)**
   - Collections in JAVA (Chapters 12 and 16)
   - Iterators (Chapter 16)

Practice sessions (30 hours)

- The Eclipse environment
- Simple Java program examples
- Programs with arrays in Java
- OrdinaArray class: sorting algorithms (bubblesort, insertionsort, selectsort, shellsort, quicksort, mergesort); the Arrays class
- Examples of application-specific classes; sorting arrays of objects; searching in arrays of objects
- Creating a dictionary

	• Class composition • Inheritance • Abstract classes and interfaces • Dictionary interface • Generic dictionary with exceptions • I/O and serialization.
Books of reference	Programmazione di base e avanzata con Java, terza edizione, di Walter Savitch e Daniela Micucci, editore Pearson, ISBN: 9788891916020 (capitoli 1-16, Appendici). Students who wish to can borrow the texts from the Library. It may be advisable to check their availability through the University Library System at https://opac.uniba.it/easyweb/w8018/index.php? and contact the library to arrange the loan.
Notes to the books	The reference text is supplemented by the slides presented during classes, which are made available on the UNIBA e-learning platform. On the e-learning platform, several written exam papers are also provided, including examples of completed exam papers.

Organization of the didactic activities

Hours

Total	Lectures	Practice sessions	Project work	Individual study
225 hours	56 hours	30 hours	0 hours	139 hours
CFU/ETCS				
9 CFU	7 CFU	2 CFU	0 CFU	

Teaching methods

The 86 hours allocated for the course are divided as follows:
56 hours of in-person lectures covering theoretical lessons;
30 hours of in-person laboratory sessions with practical activities under the supervision of the instructor.

Expected learning outcomes

Knowledge and understanding	○ Acquisition of knowledge related to the object-oriented paradigm. ○ Understanding the choices of object-oriented modeling.
Applying knowledge and understanding	○ Consolidation of programming skills; ○ Ability to develop a simple program in an object-oriented language using a professional development environment.
Other skills	*Making judgements* ○ Students are able to appreciate the creation of software based on the principles of the object-oriented paradigm, as well as the potential offered by the Java language in object-oriented programming. ○ Judgment autonomy is developed through study and practice sessions. ○ The achievement of adequate autonomy is assessed through the final examination, covering both theoretical and practical aspects. *Communication* ○ Students are able to present the topics covered in the course using the specific terminology of the discipline and can clearly and unambiguously explain their

	choices in developing simple Java programs to specialist audiences. *Learning skills* ○ Students are able present the course topics using the specific terminology of the discipline and can clearly and unambiguously explain their choices in developing simple Java programs to specialized audiences.

Assessment	
Assessment methods	**Exam Overview:** The exam consists of a single test divided into two parts: theoretical questions and a practical laboratory test. • **Theoretical Questions:** These may be open-ended and/or multiple-choice questions designed to assess the ability to present object-oriented programming concepts. The questions will cover topics discussed in lectures and included in the syllabus. • **Practical Laboratory Test:** This part evaluates programming skills in Java by requiring students to develop simple applications based on provided specifications. The programs should reflect Java knowledge covered in lectures and applied in laboratory activities. **Exam Details:** • The exam takes place in the laboratory over three hours: approximately two hours are allocated for the practical test, and one hour for the theoretical questions. It is the student's responsibility to manage his/her time effectively. • The practical test deliverable is an executable JAR file, while the theoretical test deliverable is a file in MS Word or PDF format. • Each part of the exam is worth 33 points. A minimum score of 18/33 is required for each part. The final grade is the average of the scores from both parts. A score above 30 corresponds to a grade of 30 with honors. • An interim assessment is scheduled for the mid-point of the course. This assessment is non-exemptive and serves to evaluate progress in the course.
Evaluation criteria	**Knowledge and Understanding:** ○ Ability to understand the questions posed in the written exam and respond in a relevant and thorough manner. ○ Ability to comprehend the guidelines for performing laboratory activities and completing the project. **Applied Knowledge and Understanding:** ○ Comprehensive knowledge of the course topics and their application in completing written exam exercises, addressing theoretical issues, and developing the project. **Autonomy of Judgment:** ○ Ability to debug and validate the correct functioning of the developed programs. **Communication Skills:** ○ Use of specific vocabulary in the field of computer science. **Ability to Learn:** ○ Ability to answer the written exam questions correctly, comprehensively, and with appropriate use of technical language.
Measurements and final grade	The final grade will be awarded based on the comparative evaluation of all the

	assessment criteria mentioned above.
Further information	It is recommended that students rely exclusively on information and communications provided on the official websites of the Department of Computer Science or on social groups only if they are established and managed solely by the instructors of the respective 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.uniba.it/ The course syllabi are available here: • https://elearning.uniba.it/course/view.php?id=2803 The information that all students should know is detailed in the educational regulations and course handbooks available on the website: • https://www.uniba.it/it/ricerca/dipartimenti/informatica/didattica/corsi-di-laurea/corsi-di-laurea Students are advised to be cautious regarding information and materials circulating on unofficial websites or social groups, as they can often be unreliable, incorrect, or incomplete. For any doubts, students should arrange a meeting with the instructor following the specified office hours. Link to the course on the ADA e-learning platform: https://elearning.uniba.it/course/view.php?id=2803