Principali informazioni sull'insegnamento			
Titolo insegnamento	Programmazione II		
Corso di studio	Informatica e Tecnologie per la Produzione del Software		
Crediti formativi	7 (Teoria) + 2 (Esercitazione/Laboratorio)		
Denominazione inglese	Computer Programming II		
Obbligo di frequenza	NO		
Lingua di erogazione	Italiano		
Docente responsabile	Nome Cognome	Indirizzo e-mail	
	Pasquale Ardimento	pasquale.ardimento@uniba.it	
Dettaglio credi formativi	Ambito disciplinare	*SSD* ING-INF/05	*Crediti* 7+2
Modalità di erogazione			
Periodo di erogazione	Primo semestre		
Anno di corso	Secondo		
Modalità di erogazione	Lezioni frontali Esercitazioni in aula Esercitazioni di laboratorio		

Organizzazione della didattica	
Ore totali	225 (175 Teoria + 50 Laboratorio)
Ore di corso	86 (56 Teoria + 30 Laboratorio)
Ore di studio individuale	139 (119 Teoria + 20 Laboratorio)

Calendario	
Inizio attività didattiche	26 settembre 2022
Fine attività didattiche	13 gennaio 2023

Syllabus	
Prerequisiti	Competenze di base relative alla programmazione (in C) ed ai linguaggi di programmazione.
Risultati di apprendimento previsti	• *Conoscenza e capacità di comprensione* Lo studente dovrà acquisire: le conoscenze di base ed alcuni aspetti avanzati della programmazione orientata agli oggetti nel linguaggio JAVA; il concetto di astrazione, così come i concetti di classe ed oggetto, di ereditarietà e polimorfismo. Lo studente dovrà, inoltre, approfondire gli aspetti relativi al trattamento delle eccezioni, all'identificazione di tipo al run-time (RTTI), alla programmazione generica, al sistema I/O. • *Conoscenza e capacità di comprensione applicate* Lo studente dovrà acquisire le competenze necessarie per lo sviluppo e la realizzazione di semplici applicazioni di cui si forniscono le specifiche di massima. • *Autonomia di giudizio* Lo studente dovrà dimostrare di aver acquisito una notevole autonomia di giudizio in quanto egli deve essere in grado di saper decidere in autonomia quali scelte progettuali dover applicare per raggiungere la soluzione in modo efficace ed efficiente e quali algoritmi applicare in base al problema da risolvere. • *Abilità comunicative* Lo studente deve essere in grado di illustrare in modo appropriato e talvolta dettagliato gli aspetti teorici acquisiti durante il corso e dimostrare di saperli correlare agli aspetti pratici della programmazione Object-Oriented in Java. • *Capacità di apprendere* Lo studente dovrà mostrare di aver sviluppato capacità di apprendere e di orientarsi

	agilmente nelle problematiche che richiedono apprendimento autonomo. Ad esempio, lo studente deve essere in grado di apprendere autonomamente il funzionamento di una libreria Java non presentata a lezione, ritrovando autonomamente tutte le informazioni dettagliate di cui ha bisogno mediante la J*avadoc* disponibile online.
Contenuti di insegnamento	La programmazione orientata agli oggetti. **Fondamenti**: l'astrazione nella progettazione, l'astrazione nella programmazione, oggetti, classi concrete, classi astratte, metaclassi, ereditarietà singola ed ereditarietà multipla, polimorfismo, gerarchia di classi e gerarchia di interfacce. Composizione di classi. Confronto tra ereditarietà e composizione nel riuso del software. Ambienti e linguaggi di programmazione. **Java**: caratteristiche generali del linguaggio; Java ed Internet; Java vs. C++. Ambienti di sviluppo Java. **Oggetti in Java**: costruttori; distruttori; metodi, argomenti e valori di ritorno. **Controllare il flusso di esecuzione**: uso degli operatori Java; il controllo di esecuzione; l'inizializzazione. **Nascondere le implementazioni**: i package; i modificatori di accesso; le interfacce. **Il riuso delle classi in Java**: ereditarietà, derivazione protetta; polimorfismo. I contenitori: array; collezioni; le nuove collezioni. **Trattamento delle eccezioni in Java**: Sollevamento e gestione delle eccezioni; eccezioni standard e personalizzate; **Programmazione generica in Java**: *Generics* e *container*, *Generics* ed interfacce, metodi generici, il problema dell'*erasure*; **Identificazione di tipo al run-time**: RTTI (Run-Time Type Identification) "tradizionale"; il meccanismo della *reflection*; **Il sistema I/O di Java**: librerie di Input; librerie di Output; reindirizzamento e compressione dei dati; connessione con le Basi di Dati: JDBC; **Creazione di interfacce grafiche in Java:** Progettazione e creazione di interfacce per applicazioni: il package SWING **Esercitazioni sull'ambiente di sviluppo *Eclipse***: progetto di applicazioni con più classi organizzate gerarchicamente e in package; progetto di applicazioni con classi astratte ed uso del polimorfismo; progetto di applicazioni con contenitori e trattamento delle eccezioni; progetto di applicazioni con I/O da file.

Programma	
Testi di riferimento	• *Object-Oriented Analysis and Design with Applications* (cap. 2 e 3) di Grady Booch, second edition, Addison-Wesley Professional, ISBN-13**:** 978-0321774941; • *Thinking in Java* di Bruce Eckel (cap. 1-11, 13-14, 16-17, 19-20, 23-24), 4th Edition Prentice-Hall, 2006; • *The Java Language Specification, Java SE 10 Edition* di J. Gosling, B. Joy, G. Steele, G. Bracha, A. Buckley: https://docs.oracle.com/javase/specs/jls/se10/jls10.pdf
Note ai testi di riferimento	I testi di riferimento saranno integrati con slide e materiale didattico messo a disposizione dal docente sulla piattaforma ADA
Metodi didattici	Lezioni frontali ed esercitazioni pratiche di programmazione in Java
Metodi di valutazione	L'esame consta di una prova di laboratorio e di un caso di studio. caso di studio. • Da realizzare individualmente (singolo studente). • I software di sviluppo da utilizzare dovranno essere quelli indicati dal docente; • La consegna del caso di studio è possibile in una qualunque data degli appelli di Programmazione II. • Qualora la valutazione del caso di studio fosse positiva (almeno 18/30), non si potrà più ripetere tale prova. Maggiori informazioni sul caso di studio saranno disponibili sulla piattaforma di e-

	learning. prova di laboratorio (90 minuti). • Realizzazione di un programma scritto in JAVA rispondente al problema riportato nella traccia data. La realizzazione del programma richiederà l'applicazione dei concetti appresi durante le lezioni. Qualora il voto ottenuto dallo studente fosse almeno pari a 18/30 per ambedue le prove, lo studente potrà procedere con la verbalizzazione. Il voto finale sarà determinato come la media aritmetica delle valutazioni ottenute per le due prove suddette.
Criteri di valutazione	• Conoscenza e capacità di comprensione applicate. Durante lo svolgimento della prova di laboratorio lo studente dovrà dimostrare la padronanza dei costrutti, dei concetti e dei principi di base dell'Object Orientation applicandoli opportunamente per la traccia data.
Altro	• Propedeuticità "*Gli insegnamenti di Programmazione, Architettura degli elaboratori e sistemi Operativi, e Laboratorio di Informatica sono propedeutici agli insegnamenti nei settori INF/01 e INGINF/05 del secondo anno.*"