Principali informazioni sull'insegnamento	
Titolo insegnamento	Programmazione 2 (corso B)
Corso di studio	Informatica e Tecnologie per la Produzione del Software
Crediti formativi	7+2
Denominazione inglese	Computer Programming 2 (B)
Obbligo di frequenza	No
Lingua di erogazione	Italiano

Docente responsabile	Nome Cognome	Indirizzo Mail
	Nicola Boffoli	nicola.boffoli@uniba.it

Dettaglio credi formativi	Ambito disciplinare	SSD	Crediti
	Informatico	ING INF 05	7+2

Modalità di erogazione	
Periodo di erogazione	Semestre 1
Anno di corso	2
Modalità di erogazione	Lezioni frontali 86 ore (56 Teoria 30 Esercitazione e/o Laboratorio)

Organizzazione della didattica	
Ore totali	225 ore
Ore di corso	86
Ore di studio individuale	139

Calendario	
Inizio attività didattiche	6 Ottobre 2020
Fine attività didattiche	13 gennaio 2021

Syllabus	
Prerequisiti	*Conoscenze di programmazione imperativa, architettura dei calcolatori, algoritmi e strutture dati, basi di dati*
Risultati di apprendimento previsti (declinare rispetto ai Descrittori di Dublino) (si raccomanda che siano coerenti con i risultati di apprendimento del CdS, compreso i risultati di apprendimento trasversali)	• *Conoscenza e capacità di comprensione* Lo studente dovrà acquisire una conoscenza di base della modellazione e programmazione Orientata agli Oggetti (OO). • *Conoscenza e capacità di comprensione applicate* Attraverso l'introduzione del linguaggio di programmazione Java e lo sviluppo guidato di un progetto completo lo studente approfondirà la progettazione orientata a oggetti, la composizione di classi, l'uso di gerarchie di classi ed alcune strutture dati fondamentali, lo sviluppo di applicazioni clientserver, lo sviluppo di interfacce grafiche. • *Autonomia di giudizio*

	Lo studente dovrà dimostrare di aver acquisito un'elevata autonomia di giudizio nella realizzazione di servizi di data mining di interesse per la comunità scientifica e di business progettati e realizzato coerentemente con i principi del paradigma OO • *Abilità comunicative* Lo studente sarà in grado di relazionare in maniera appropriate in riferimento alla modellazione e programmazione orientata ad oggetti • *Capacità di apprendere* Lo studente dovrà mostrare di aver sviluppato capacità di apprendere e di orientarsi agilmente nelle problematiche che si presentano durante lo sviluppo di software progettato e realizzato coerentemente con i principi del paradigma OO.
Contenuti di insegnamento	Introduzione ai paradigmi di programmazione. I tre approcci alla programmazione: operazionale, definizionale e dimostrazionale. L'astrazione nella programmazione Fondamenti: Introduzione all'astrazione. Astrazione di funzione, di procedura, di controllo, e di selettore. Astrazione di tipo e tipi astratti di dato. Specifiche algebriche e assiomatiche per i tipi astratti di dato. I moduli per l'incapsulamento dell'informazione e l'information hiding. Oggetti e classi di oggetti. Astrazione di dati: Tipo astratto di dato vs. classe di oggetti. Astrazione generica. Ambienti e linguaggi di programmazione. La programmazione orientata agli oggetti. Fondamenti: 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 e 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. Approfondimenti su Java: il trattamento delle eccezioni; identificazione di tipo al run-time; programmazione generica in Java; il sistema I/O di Java. Connessione con le Basi di Dati: JDBC. Progettazione e creazione di interfacce per applicazioni: il package SWING, JavaFX Programmazione in rete: socket. Il multithreading: creazione di classi attive; sincronizzazione nell'accesso dei metodi. Servlet. Cenni su SPRING. Esercitazioni guidate su: L'ambiente Eclipse. Progetto di applicazioni con singole classi; progetto di applicazioni con più classi organizzate gerarchicamente e in package; progetto di applicazioni con classi astratte e uso del polimorfismo; progetto di applicazioni con contenitori e trattamento delle eccezioni; progetto di applicazioni con I/O da file; progetto di connessione a database; progetto di applicazioni con GUI mediante con SWING; progetto di applicazioni client-server e multithreading.

Programma	
Testi di riferimento	A.L. Ambler, M.H. Burnett, & B.A. Zimmerman Operational Versus Definitional: A Perspective on Programming Paradigms IEEE Computer, 25(9): 28-43, September 1992. M. Shaw Abstraction Techniques in Modern Programming Languages IEEE Software, 10-26, October 1984. G. Masini, A. Napoli, D. Colnet, D. Léonard, & K. Tombre Linguaggi per la Programmazione a Oggetti (cap. 2-3, 6) Gruppo Editoriale Jackson, 1991 D. A. Watt Programming Language Concepts and Paradigms (cap. 5-6) Prentice Hall, 1990. Bruce Eckel Thinking in Java, 4th Edition (cap. 1-11, 13-14, 16-17, 19-20, 23-24) Prentice-Hall, 2006 Walter Savitch. Programmazione di base e avanzata con Java 2/ed Pearson Education, 2014
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 (indicare almeno la tipologia scritto, orale, altro)	Prova scritta (o esoneri) Prova di laboratorio

Criteri di valutazione (per ogni risultato di apprendimento atteso su indicato, descrivere cosa ci si aspetta lo studente conosca o sia in grado di fare e a quale livello al fine di dimostrare che un risultato di apprendimento è stato raggiunto e a quale livello)	In considerazione della natura teorico-pratica del corso, la verifica dell'apprendimento avverrà durante la prova scritta e già in itinere durante le lezioni di laboratorio. *Conoscenza e capacità di comprensione* Durante la prova scritta lo studente dovrà rispondere a quesiti che verificheranno la sua padronanza dei concetti modellazione e programmazione OO. *Conoscenza e capacità di comprensione applicate* Durante la prova di laboratorio lo studente dovrà realizzare i programmi Java richiesti dal docente che verificheranno la padronanza del linguaggio JAVA Durante le lezioni di laboratorio, lo studente dovrà scrivere, in autonomia, parti di un software dimostrando la padronanza del linguaggio JAVA *Autonomia di giudizio* Lo studente dovrà utilizzare gli strumenti di documentazione e testing al fine di confermare la robustezza del software realizzato *Abilità comunicative* Lo studente dovrà dimostrare durante la prova scritta proprietà di linguaggio e padronanza dei contenuti del corso *Capacità di apprendere* Lo studente dovrà dimostrare la sua capacità di apprendere tramite lo sviluppo, in autonomia, delle funzionalità richieste durante le esercitazioni.
Altro	