Principali informazioni sull'insegnamento	
Titolo insegnamento	Programmazione
Corso di studio	Informatica
Crediti formativi	9+3
Denominazione inglese	Computer Programming
Obbligo di frequenza	No
Lingua di erogazione	Italiano

Docente responsabile	Nome Cognome	Indirizzo Mail
	Fabio Abbattista	Fabio.abbattista@uniba.it
Luogo e orario di ricevimento	Dipartimento Informatica	Martedì dalle 16:00 – 17:00

Dettaglio credi formativi	Ambito disciplinare	SSD	Crediti
	Informatico, fisico, matematico, economico, linguistico	INF/01-Informatica	9+3

Modalità di erogazione	
Periodo di erogazione	I° semestre
Anno di corso	I°
Modalità di erogazione	Lezioni frontali, esercitazioni, laboratorio

Organizzazione della didattica	
Ore totali	117 (corso) + 183 (studio individuale)
Ore di corso	72+45
Ore di studio individuale	153+30

Calendario	
Inizio attività didattiche	24 settembre 2018
Fine attività didattiche	11 gennaio 2019

Syllabus	
Prerequisiti	
Risultati di apprendimento previsti	• *Conoscenza e capacità di comprensione* Lo studente dovrà essere in grado di analizzare e risolvere semplici problemi, progettando e sviluppando programmi nel linguaggio C. • *Conoscenza e capacità di comprensione applicate* Lo studente dovrà acquisire competenze relative a: - Traduzione di semplici algoritmi in programmi correttamente funzionanti e ben documentati; - Capacita' di individuazione di malfunzionamenti attraverso il debugging; - Capacità di problem-solving attraverso l'applicazione di nozioni apprese nelle discipline informatiche di base nella pratica della programmazione. • *Autonomia di giudizio* Lo studente deve dimostrare di aver acquisito autonomia di giudizio e capacità di valutazione degli algoritmi sviluppati da lui o

	da terzi. • *Abilità comunicative* Lo studente deve essere in grado di illustrare in modo appropriato le caratteristiche tecniche degli strumenti e delle metodologie informatiche relative agli algoritmi e alla programmazione in un determinato linguaggio di programmazione. • *Capacità di apprendere* Lo studente dovrà mostrare di essere in grado di orientarsi agevolmente nelle problematiche relative alla comprensione e all'utilizzo delle tecnologie e dei metodi di competenza per lo sviluppo di algoritmi e per la loro traduzione in programmi per computer.
Contenuti di insegnamento	1.Introduzione Problem solving: algoritmi e programmi. Programmazione: una definizione Linguaggi assemblativi e di alto livello. Linguaggi imperativi. Cenni sulla loro evoluzione. Linguaggi e grammatiche. Sintassi dei linguaggi di programmazione. Diagrammi sintattici. Forma di Backus-Naur. Cenni sui compilatori. 2. Problem solving Definizione e proprieta' degli algoritmi. Alcuni semplici esempi. Progettazione di un algoritmo per raffinamenti successivi. Astrazione 3. Rappresentazione di algoritmi Specifica di un algoritmo: diagrammi di flusso, albero di decomposizione, linguaggio naturale, pseudocodice. Costrutti base: la sequenza. Esempi. Costrutti base: la selezione. Esempi. Costrutti base: la iterazione. Esempi. Programmazione strutturata. Teorema di Bohem-Jacopini (enunciato). Algoritmi elementari: conteggio, sommatoria di un insieme di numeri, calcolo del fattoriale, conversione da caratteri a numeri in base 10 e da numero in base 10 a caratteri, massimo comun divisore 4.Progettazione del software Cenni su programmazione in grande e programmazione in piccolo e sulle metodologie di progetto top-down e bottom-up. Analisi dei requisiti. Esempi.

	Progetto del software. Esempi. Codifica e debug. Correttezza, classificazione degli errori. Esempi. Test di un programma: metodi basati sulle specifiche. Esempi. 5. Linguaggi di programmazione: dati predefiniti Tipi di dato. Tipi semplici. Compatibilità ed equivalenza tra tipi di dato. Variabili e costanti. Istruzione di assegnazione. Dati strutturati: array. Dati strutturati : stringhe. Esempi Algoritmi su array e stringhe: ricerca del massimo e minimo, calcolo del valore medio, costruzione di istogrammi mediante array, inversione degli elementi e rimozione valori duplicati. Algoritmi su matrici: somma e prodotto di 2 matrici, trasposta di una matrice quadrata. 6. Linguaggi di programmazione: controllo Strutture di controllo di base. Astrazione funzionale mediante sottoprogrammi (procedure e funzioni). Identificatori e scope/campo di visibilità di un identificatore. Valore di ritorno delle funzioni. Esempi Parametri formali ed effettivi. Esempi. Tecniche di legame dei parametri: per valore. Esempi. Tecniche di legame dei parametri: per indirizzo. Puntatori. Esempi. Effetti collaterali in procedure e funzioni. Gestione delle attivazioni dei sottoprogrammi. 7. Tipi di dato strutturato e dati definiti dall'utente Tipi di dato strutturato: Union. Esempi. Tipi di dato strutturato: Struct. Esempi. Tipi di dato utente: typedef. Esempi. 8. I file Definizione di file. Tipi di file. I file in C Gestione dei file di testo. Esempi. Gestione del file binari. Esempi. 9.Algoritmi fondamentali Algoritmo di ricerca binaria. Algoritmi di ordinamento: ordinamento per selezione, per inserzione e per scambi, ricerca lineare e binaria.

Programma	
Testi di riferimento	P. Deitel e H. Deitel, Il Linguaggio C - Fondamenti e tecniche di programmazione, Pearson, 2013
Note ai testi di riferimento	Testi integrativi: B.W. Kernighan e R. Pike, Programmazione nella pratica, Addison-Wesley, 1999. J.R. Hanly, E.B. Koffman, Problem solving e programmazione in C, Apogeo, 2013
Metodi didattici	Lezioni frontali ed esercitazioni pratiche in laboratorio
Metodi di valutazione	Alcune prove pratiche da svolgere in itinere, non obbligatorie. Il superamento delle prove in itinere e/o i risultati delle esercitazioni pratiche attribuiscono una premialità sul voto finale. Prova di laboratorio e orale.
Criteri di valutazione	Lo studente dovrà dimostrare di aver acquisito la capacità di progettare l'algoritmo ottimale per la soluzione di problemi con caratteristiche diverse. Inoltre deve aver sviluppato buone competenze nell'utilizzo del linguaggio C.
Altro	