UNIVERSITÀ DEGLI STUDI DI PALERMO

DIPARTIMENTO	Matematica e Informatica
ANNO ACCADEMICO OFFERTA	2015/2016
ANNO ACCADEMICO EROGAZIONE	2017/2018
CORSO DILAUREA	INFORMATICA
INSEGNAMENTO	COMPILATORI
TIPO DI ATTIVITA'	B
AMBITO	50166-Discipline Informatiche
CODICE INSEGNAMENTO	14049
SETTORI SCIENTIFICO-DISCIPLINARI	INF/01
DOCENTE RESPONSABILE	MANTACI SABRINA Professore Associato Univ. di PALERMO
ALTRI DOCENTI	
CFU	6
NUMERO DI ORE RISERVATE ALLO STUDIO PERSONALE	102
NUMERO DI ORE RISERVATE ALLA DIDATTICA ASSISTITA	48
PROPEDEUTICITA'	05880 - PROGRAMMAZIONE E LABORATORIO C.I. 16670 - ALGORITMI E STRUTTURE DATI 16784 - SISTEMI OPERATIVI 16450 - ARCHITETTURE DEGLI ELABORATORI 16448 - METODI MATEMATICI PER L'INFORMATICA 16671 - INFORMATICA TEORICA
MUTUAZIONI	
ANNO DI CORSO	3
PERIODO DELLE LEZIONI	2° semestre
MODALITA' DI FREQUENZA	Facoltativa
TIPO DI VALUTAZIONE	Voto in trentesimi
ORARIO DI RICEVIMENTO DEGLI STUDENTI	**MANTACI SABRINA** Giovedì 15:00 18:00 Room 217 - second Floor. DMI - Via Archirafi 34

DOCENTE: Prof.ssa SABRINA MANTACI

PREREQUISITI	
RISULTATI DI APPRENDIMENTO ATTESI	Conoscenza e capacità di comprensione Il corso intende fornire agli studenti le nozioni necessarie per comprendere ed affrontare le diverse problematiche relative alle fasi della compilazione con particolare attenzione all'analisi lessicale, sintattica e semantica ma che trovano applicazione anche in altri contesti (traduzioni di linguaggi, parser, scanner). Il corso si prefigge anche di trasmettere la conoscenza di importanti strumenti di generazione automatica di parser e scanner. Capacità di applicare conoscenza e comprensione Il corso si pone come obiettivo principale la comprensione, da parte dello studente, dei modelli computazionali e delle tecniche che gestiscono il funzionamento degli analizzatori lessicali e sintattici in un moderno compilatore, al fine di applicare tali metodologie da un lato per la generazione automatica o manuale di tali analizzatori, e dall'altro per la trasformazione e l'analisi dei testi guidata dalla sintassi. Gli studenti avranno modo di applicare le conoscenze acquisite anche attraverso un elaborato realizzato in gruppo facendo uso degli strumenti di generazione automatica di parser e scanner. Autonomia di giudizio Gli studenti saranno guidati ad apprendere in maniera critica e responsabile gli argomenti che verranno loro proposti in aula e in laboratorio. Ciascuno studente avrà inoltre occasione di arricchire la propria autonomia di giudizio attraverso la realizzazione di un progetto o di un elaborato che sarà parte integrante della prova di valutazione. Abilità comunicative Attraverso le attività di laboratorio previste, il corso tenderà a sviluppare negli studenti l'interazione e la capacità di saper lavorare in gruppo, di confrontarsi sulle problematiche al fine di individuare le soluzioni in base alle conoscenze acquisite durante il corso. L'acquisizione delle abilità comunicative sarà realizzata tramite la partecipazione attiva dello studente alle attività di laboratorio nonché l'esposizione dei risultati del lavoro individuale o di gruppo su argomenti o problematiche proposti dal docente. Capacità d'apprendimento Il materiale proposto in aula e le esercitazioni in laboratorio svilupperanno le capacità di apprendimento degli studenti che saranno in grado di "interrogare" in modo integrato le proprie conoscenze-competenze a fronte delle problematiche affrontate. Gli studenti saranno stimolati inoltre ad una conoscenza più approfondita e critica dei linguaggi di programmazione a loro già noti, tramite lo studio di come avviene il processo di compilazione di tali linguaggi.
VALUTAZIONE DELL'APPRENDIMENTO	Prove in itinere. Prova orale con discussione di un elaborato realizzato singolarmente o in gruppo.
OBIETTIVI FORMATIVI	L'insegnamento ha due obiettivi. Il primo è di tipo culturale. Si vuole cioè introdurre una teoria, quella alla base della realizzazione dei compilatori per i linguaggi di programmazione, che rappresenta probabilmente il più importante contributo scientifico allo sviluppo e alla diffusione delle tecnologie informatiche. Il secondo obiettivo è più applicativo. Infatti, anche qualora lo studente non venga mai coinvolto (nella propria carriera lavorativa) nel progetto di un nuovo compilatore, è comunque altamente probabile che egli/ella si trovi spesso ad utilizzare metodologie e tecniche che di tale teoria fanno parte: dagli automi e le grammatiche come formalismi per definire il comportamento di un sistema, alle tecniche per realizzare traduttori eventualmente più semplici di un compilatore vero e proprio (ad esempio, l'interprete di un file di configurazione). Verrà studiata la struttura generale di un compilatore, ponendo poi particolare enfasi agli aspetti di analisi lessicale e di parsing. Verranno studiati parser di tipo top-down (a discesa ricorsiva, predittivi e LL(1)) e di tipo bottom-up (LR e LALR). Infine, saranno esaminati esempi di traduttori per semplici linguaggi.
ORGANIZZAZIONE DELLA DIDATTICA	Lezioni frontali in aula e in laboratorio
TESTI CONSIGLIATI	A. Aho, M. Lam, R. Sethi, J. Ullman Compilers, Principles, Techniques & Tools Addison Wesley A. Aho, M. Lam, R. Sethi, J. Ullman COMPILATORI. Principi, Tecniche e strumenti Addison Wesley Stefano Crespi Reghizzi Linguaggi Formali e Compilazione Pitagora Editrice

PROGRAMMA

ORE	Lezioni
4	Introduzione: Linguaggi di Programmazione e Processori di Linguaggi di Programmazione. Compilatori e Linguaggi. Linguaggi Macchina, Linguaggi Assembly ed evoluzione dei linguaggi di programmazione. Compilatori e Interpreti. Macchina Virtuale. Struttura di un compilatore: Preprocessore. Linker e Loader. Fasi della compilazione. Front End e Back End. Passate di un compilatore.
6	Analisi lessicale: Operazioni preliminari. Token e Lessemi. Errori lessicali. Token e espressioni regolari. Definizioni regolari. Eliminazioni di ambiguità. Automi a stati finiti. Implementazioni di DFA. Simulazioni di NFA. Generazione automatica di scanner. Uso di Flex.
6	Risoluzione di esercizi con l'ausilio di Flex.
14	Analisi sintattica: Grammatiche context-free. Alberi di derivazione. Grammatiche ambigue. Automi a pila deterministici e non deterministici. Complessità di calcolo di un automa a pila. Algoritmo di Earley. Errori sintattici e metodi di gestione degli errori. Parser discendenti o top-down: Parser a discesa ricorsiva. Parser LL(1). Eliminazione della ricorsione sinistra. Fattorizzazione sinistra. Insiemi First e Follow. Parser ascendenti o bottom-up: Parser shift-reduce. Parser LR(0). Parser SLR. Parser LR(1). Parser LALR(1). Proprietà dei linguaggi e delle grammatiche LR(k). Confronto tra le grammatiche LL(k) e LR(k). Generatori automatici di parser. Uso di Bison.
6	Analisi semantica: Semantica statica e dinamica. Grammatiche con attributi. Semantica guidata dalla sintassi. Albero sintattico decorato. Calcolo degli attributi. Grafo delle dipendenze. Grammatiche con S- attributi. Grammatiche con L-attributi. Ordinamento topologico del grafo delle dipendenze. Tabella dei simboli. Vari tipi di implementazioni tramite array, liste concatenate e ABR. Implementazione tramite hash table with chaining.Attributi di visibilità e metodi di realizzazione. Type checking. Equivalenza d tipi. Type coercion.
10	Risoluzione di esercizi con l’ausilio di Bison. Uso della tabella dei simboli
2	Generazione del codice: Codice intermedio. Codice a tre indirizzi. Strutture dati per l’implementazione del 3AC. Codice per macchina virtuale. P-code. Ottimizzazione del codice. Esempi di ottimizzazioni indipendenti dalla macchina. Generatori di codice oggetto. Esempi di ottimizzazioni dipendenti dalla macchina.