<table>
<tr><td colspan="2">Classe delle lauree in:
Ingegneria dell'Informazione (classe 09)</td><td colspan="2">Corso di laurea magistrale in:
Ingegneria Informatica</td><td colspan="2">Anno accademico:
2011 - 2012</td></tr>
<tr><td>Tipo di attività formativa:
Caratterizzante</td><td>Ambito disciplinare:
Ingegneria Informatica</td><td colspan="2">Settore scientifico disciplinare: Sistemi di elaborazione dell'informazione (ING-INF/05)</td><td colspan="2">CFU:
9</td></tr>
<tr><td>Titolo dell'insegnamento:
Linguaggi Formali e Compilatori</td><td>Codice dell'insegnamento:</td><td>Tipo di insegnamento:
obbligatorio</td><td>Anno:
I</td><td>Semestre:
I</td></tr>
<tr><td colspan="6">DOCENTE:
Prof. Giacomo Piscitelli</td></tr>
<tr><td colspan="6">ARTICOLAZIONE IN TIPOLOGIE DIDATTICHE:
Il corso comprende 76 ore di lezioni teoriche e di esercitazioni, 2 ore di introduzione al progetto.</td></tr>
<tr><td colspan="6">CONOSCENZE PRELIMINARI:
Linguaggi di programmazione, Ingegneria del software.</td></tr>
<tr><td colspan="6">OBIETTIVI FORMATIVI:
Il corso si propone di introdurre ai principali concetti della definizione sintattica dei linguaggi e di fornire conoscenza e pratica delle tecniche di traduzione per i moderni linguaggi di programmazione sia di tipo general purpose che per applicazioni specifiche.</td></tr>
<tr><td colspan="6">PROGRAMMA:
1. Linguaggi di programmazione (1CFU: 8l): Tecniche di implementazione dei linguaggi; Linguaggi e macchine astratte; Traduttori, compilatori e interpreti; Alfabeto, stringhe, vocabolario; Linguaggio, approcci alla definizione.
2. Grammatiche e classificazione (1.25 CFU: 8l-4e): Grammatiche regolari e context-free; Backus-Naur Form (BNF); Automi e Macchine di Turing
3. Struttura di un compilatore (1.5 CFU: 10l-4e): Analisi lessicale (scanning), implementazione di scanner.
4. Struttura di un compilatore (1.75 CFU: 12l-4e): Analisi sintattica (parsing), top-down parsing, LL parsing, bottom-up parsing, LR parsing, gestione errori di analisi e implementazione di parser.
5. Struttura di un compilatore (1.75 CFU: 12l-4e): Analisi Semantica, syntax-directed translation, attribute definitions.
6. Rappresentazione intermedia (0.5 CFU: 4l)
7. Generazione del codice (0.5 CFU: 4l)
8. Tecniche di ottimizzazione del codice (0.25 CFU: 2l)
9. Progetto (1 CFU): avvio alla realizzazione di un "lavoro d'anno".</td></tr>
<tr><td colspan="6">METODI DI INSEGNAMENTO:
Lezioni ed esercitazioni in aula supportate da PC portatile e proiettore; chiarimenti a richiesta forniti dal docente; discussione circa le modalità di svolgimento del "lavoro d'anno" (progetto).</td></tr>
<tr><td colspan="6">CONOSCENZE E ABILITÀ ATTESE:
Al termine del corso gli allievi conosceranno la struttura di un traduttore e saranno in grado di realizzare compilatori di linguaggi di programmazione sia di tipo general purpose che per applicazioni specifiche.</td></tr>
<tr><td colspan="6">SUPPORTI ALLA DIDATTICA:
PC portatile e proiettore; libro di testo e appunti in formato elettronico (.pdf) approntati dal docente; tool open-source; progetti d'anno distribuiti attraverso lo "scaffale virtuale" del sito didattico del docente.</td></tr>
<tr><td colspan="6">CONTROLLO DELL'APPRENDIMENTO E MODALITÀ D'ESAME:
Realizzazione del prototipo di semplici traduttori/compilatori. Esame orale.</td></tr>
<tr><td colspan="6">TESTI DI RIFERIMENTO PRINCIPALI:
- Trasparenze del corso
- Aho, Lam, Sethi, Ullman, Compilers: principles, techniques & tools, Pearson Int.
- Hopcroft, Motwani, Ullmann, Introduction to Automata Theory, Languages and Computation, Addison Wesley.</td></tr>
<tr><td colspan="6">ULTERIORI TESTI SUGGERITI:
David Gries, Principi di progettazione dei compilatori, Franco Angeli Editore.</td></tr>
</table>

<table>
<tr><td colspan="2">Degree class:
Information Engineering</td><td>First level (three year) degree:
Computer System Engineering</td><td colspan="2">Academic year:
2011 - 2012</td></tr>
<tr><td>Type of course
Characterizing</td><td>Disciplinary area:
Computer System Engineering</td><td>Scientific Discipline Sector:
Information Processing Systems (ING-INF/05)</td><td colspan="2">ECTS Credits:
9</td></tr>
<tr><td>Title of the course:
Formal Languages and Compilers</td><td>University Code:</td><td>Type of course:
compulsory</td><td>Year:
I</td><td>Semester:
I</td></tr>
<tr><td colspan="5">LECTURER:
Prof. Giacomo Piscitelli</td></tr>
<tr><td colspan="5">HOURS OF INSTRUCTION
76 hours of in-class lectures and applications; 2 hours of introduction to the project-work.</td></tr>
<tr><td colspan="5">PREREQUISITES:
Programming languages; software engineering.</td></tr>
<tr><td colspan="5">AIMS:
This course aims at endowing future computer systems engineers with the topics of syntactic definition of programming languages and with the translation techniques of modern programming languages both general purpose and application oriented.</td></tr>
<tr><td colspan="5">PROGRAMME:
1. Theory of formal languages (1CFU: 8l): Languages implementation techniques; Languages and abstract machines; Translators, compilers and interpreters; alphabet, strings, vocabulary; Language definition approaches.
2. Grammars and classifications (1.25 CFU: 8l-4e): Regular and context-free grammars; Backus-Naur Form (BNF); Automata and Turing Machines.
3. Structure of a compiler (1.5 CFU: 10l-4e): Lexical analysis (scanning), scanner implementation.
4. Structure of a compiler (1.75 CFU: 12l-4e): Syntax analysis (parsing), top-down parsing, LL parsing, bottom-up parsing, LR parsing, errors management and parser implementation.
5. Structure of a compiler (1.75 CFU: 12l-4e): Semantic analysis, syntax-directed translation, attribute definitions.
6. Intermediate program representation (0.5 CFU: 4l)
7. Code generation (0.5 CFU: 4l)
8. Code optimization techniques (0.25 CFU: 2l)
9. Project (1 CFU): starting the implementation “project work”.</td></tr>
<tr><td colspan="5">TEACHING METHODS:
In class lectures and applications; tool testing.</td></tr>
<tr><td colspan="5">EXPECTED OUTCOME AND SKILL:
A successful student should know, understand, design and implement a simple translator/compiler/interpreter.</td></tr>
<tr><td colspan="5">TEACHING AIDS:
Lectures and notes, teacher’s foils, guides and exercises, previous assignments available through the “virtual shelf” in the didactic section of the personal url.</td></tr>
<tr><td colspan="5">EXAMINATION METHOD:
The exam consists of the discussion of the project work and of an oral examination.</td></tr>
<tr><td colspan="5">BIBLIOGRAPHY:
- Course foils
- Aho, Lam, Sethi, Ullman, Compilers: principles, techniques & tools, Pearson Int.
- Hopcroft, Motwani, Ullmann, Introduction to Automata Theory, Languages and Computation, Addison Wesley.</td></tr>
<tr><td colspan="5">FURTHER BIBLIOGRAPHY:
David Gries, Principles of compilers design, Franco Angeli Editore.</td></tr>
</table>