Ano Lectivo	2016/17
Curso	Engenharia Informática
Unidade Curricular	Teoria da Computação
Língua de ensino	Português
ECTS/tempo de trabalho (horas)	<table><tr><th>ECTS</th><th>Total</th><th colspan="7">Horas de contacto semestral</th></tr><tr><td rowspan="2">5</td><td rowspan="2">65</td><td>T</td><td>TP</td><td>PL</td><td>S</td><td>TC</td><td>O</td><td>OT</td></tr><tr><td>15</td><td>15</td><td></td><td></td><td></td><td></td><td></td></tr></table> T - Teóricas; TP - Teórico-práticas; PL - Prática-laboratorial; S - Seminário; OT - Orientação tutorial; TC - Trabalho de campo; E - Estágio; O* - Outras horas caraterizadas como Ensino Clínico ao abrigo da Diretiva nº 77/453/CEE de 27 Junho adaptada pela Diretiva 2005/36/CE;
Docente Responsável/Carga letiva [nome completo e e-mail]	Jorge Miguel Calha Rainho Machado / jmachado@estgp.pt
Outros Docentes e respetivas cargas letivas [nome completo e e-mail]	Jorge Miguel Calha Rainho Machado / jmachado@estgp.pt
Pré-requisitos [competências à entrada; pré-requisitos; precedências]	Não
Objetivos da aprendizagem (conhecimentos, aptidões e competências) a desenvolver pelos estudantes, operacionalização dos objetivos e medição do seu grau de cumprimento)	Em primeiro lugar os alunos devem aprender a trabalhar com modelos computacionais comuns: os autómatos finitos, as gramáticas e as expressões regulares. Pretende-se que os alunos dominem a teoria das linguagens regulares e que consigam aplica-la às problemáticas associadas às máquinas de estados determinísticas e não determinísticas. Com isto espera-se que os alunos fiquem aptos a definir gramáticas e expressões passíveis de ser transformadas em autómatos e vice-versa com o intuito de dominar o parsing de linguagens protocolares representadas em texto. Em segundo lugar pretende-se que os alunos sejam capazes de aplicar os conhecimentos do objetivo número um para criar compiladores/interpretadores de linguagens regulares com o objetivo de automatizar máquinas de estados e criar ferramentas de parsing. Pretende-se que estas ferramentas sejam capazes de receber um input textual e criar uma árvore de estruturas passível de ser computada por um sistema automático.
Conteúdos Programáticos [estrutura de conteúdos a desenvolver para o total de horas previsto]	1. Automação e Linguagens Regulares a) Autómatos finitos deterministas. b) Propriedades de fecho de linguagens regulares. c) Equivalência e minimização de autómatos finitos deterministas. d) Autómatos finitos não deterministas. e) Equivalência de autómatos finitos deterministas e não deterministas relativamente às linguagens reconhecidas. f) Gramática, gramática livre de contexto e gramática regular. g) Expressões regulares. h) Conversão entre as diferentes representações de linguagens regulares: da gramática regular para a expressão regular e da expressão regular para a gramática regular; da expressão regular para o autómato finito não determinista com movimentos épsilon; do autómato finito determinista para a gramática regular; da gramática regular para o autómato finito não determinista. 2. Compiladores a) Estrutura de um compilador. b) Análise Lexical, Sintáctica e Semântica. c) Ferramentas Lex e Yacc.
Demonstração da coerência dos conteúdos programáticos com os objectivos da unidade curricular	O ponto um tem base teórica e servem para introduzir os conceitos de automação necessários para formar o aluno segundo o primeiro objetivo da unidade. Com isto os alunos ficarão aptos a desenvolver autómatos, linguagens regulares e expressões regulares passíveis de ser transformadas entre si segundo técnicas automáticas também elas lecionadas na alínea h) dos conteúdos.

	O ponto dois é de cariz prático e serve para formar o aluno na tecnologia necessária para construir um compilador operacional capaz de ser parametrizado com uma gramática. Com estas ferramentas os alunos ficam aptos a responder ao objetivo dois da unidade e aprendem a desenvolver um parser real.
Metodologias de ensino (avaliação incluída) [indicar os produtos, critérios e pesos de avaliação] (máx1000 carateres)	**1 - Metodologias de ensino** Juntamente com a teoria serão apresentados aos alunos exercícios semanais que serão sempre iniciados nas aulas e terminados em casa. Durante a parte dois do programa serão fornecidos exemplos que irão evoluir até se chegar a um compilador exemplo completo. Estes exemplos serão sempre testados pelos alunos nos últimos 30 minutos de aula. O projeto será desenvolvido em grupos de três ou quatro elementos pois é um trabalho moroso e carece de várias fases de implementação que desta forma são distribuídas pelo grupo. **2 - Avaliação por frequência** Exercícios sobre automação e linguagens regulares 10% Teste escrito 70% Exercícios práticos de aula sobre código de 3 endereços 20% **3 - Avaliação por Exame** Teste Escrito sobre toda a matéria 100%
Demonstração da coerência das metodologias de ensino com os objectivos da aprendizagem da unidade curricular	Pretende-se que os alunos estejam aptos a desenvolver interpretadores de linguagens regulares, como tal é fundamental que dominem as técnicas de desenho e operação de linguagens regulares. A primeira parte dos conteúdos serve para formar os alunos no sentido de perceberem os modelos e conseguirem executar manualmente os passos dos algoritmos de interpretação. Para cumprir esse objetivo os alunos fazem exercícios práticos de execução manual dos passos dos interpretadores e treinam em casa com mais exercícios fornecidos nas aulas. Estes exercícios são feitos em papel para não distrair os alunos para problemas tecnológicos. Pretende-se em segundo lugar que os alunos consigam desenvolver um interpretador/compilador de uma linguagem e por isso são lecionados os componentes clássicos de um compilador e pede-se aos alunos que implementem e testem programas automáticos que executam essas fases. Esta fase consiste ainda em exercícios da parte de compiladores os quais são iniciados durante as horas de contacto e terminados em casa com validação nas horas de contacto seguintes. Este conjunto de exercícios já envolve programação. Pretende-se por fim que os alunos consigam usar os compiladores em casos reais. Para isso são ensinadas ferramentas aceites pela comunidade, o Lex e o Yacc, para que os alunos desenvolvam um projeto semelhante ao que seria um caso profissional, no qual as ferramentas de parsing são conhecidas e o aluno tem de usa-las para criar o compilador da linguagem específica do projeto.
Bibliografia Principal	1. Sernada, C., ” Introdução à Teoria da Computação”, Editorial Presença, 1993. 004.92 SRN 2. Hopcroft, J., Motwani. R. e Ullman. J.. “Introduction to Automata Theory, Languages, and Computation Addison Wesley”, 2001. 3. Sipser, M., “Introduction to the Theory of Computation”, PWS Publishing Company, 1997. 4. Aho, A., Sethi, R. e Ullman, J., “Compilers: Principles, Techniques and Tools”, Adison-Wesley, 1986. 5, Jorge Machado. Manual de Processador Virtual de código de 3 endereços, Instituto Politécnico de Portalegre, 2012.
Bibliografia Complementar	Tibault Langois. Compiladores. IST. 2000
Situações especiais [estudantes com estatuto especial]	**1 - Avaliação por frequência** **2 - Avaliação por Exame**