LICENCIATURA: Engenharia Informática	**ÁREA CIENTÍFICA:** Engenharia Informática
UNIDADE CURRICULAR/CURRICULAR UNIT: Introdução à Ciência dos Computadores **/** Introduction to Computer Science	**ECTS:** 6
DURAÇÃO: Semestral	**HORAS DE CONTACTO TEÓRICO PRÁTICAS**: 60 (48 TP+12 OT)

OBJETIVOS DE APRENDIZAGEM / LEARNING OUTCOMES OF THE CURRICULAR UNIT

Para concluir com sucesso esta unidade curricular, os estudantes deverão demonstrar possuir os seguintes conhecimentos e capacidades:

1. Compreender a importância teórica da ciência dos computadores.
2. Conhecer as principais linguagens de programação, distinguindo se são compiladas ou interpretadas.
3. Conhecer as características de algumas linguagens de programação de alto nível: imperativas, estruturadas, funcionais, programação em lógica, orientadas a objetos.
4. Conhecer o conceito de algoritmo e fluxograma para a resolução de problemas, sendo capaz de criar algoritmos e fluxogramas para exemplos simples.
5. Ser capaz de utilizar a linguagem Python para a execução de programas muito simples.
6. Saber o que é um Sistema Operativo, quais os seus componentes fundamentais, e distinguir S.O. proprietários e de acesso livre (open-source).
7. Conhecer e dominar a lógica computacional e lógica, operações lógicas e tabelas de verdade, assim com as leis de De Morgan.
8. Conhecer os princípios fundamentais dos sistemas de numeração binários e da sua implementação num computador.
9. Investigar os conceitos de computação avançada e do futuro da computação, tais como Inteligência Artificial, Computação Quântica, processamento na Cloud.
10. Desenvolver competências pessoais e sociais essenciais para a integração no mundo empresarial, nomeadamente: trabalho em equipa, resolução de problemas, relacionamento interpessoal, capacidade de iniciativa, comunicação assertiva, criatividade, espírito crítico, boa gestão de tempo, adaptabilidade e resiliência.

(English)

To successfully complete this curricular unit, students must demonstrate the following knowledge and skills:

1. Understand the theoretical importance of computer science.
2. Account the main programming languages, distinguishing if they are compiled or interpreted.
3. Identify the characteristics of some high-level programming languages: imperative, structured, functional, logic programming, object-oriented.

4. Know the concept of algorithm and flowchart for solving problems, being able to create algorithms and flowcharts for simple examples.
5. Be able to use the Python language to run very simple programs.
6. Understand what an Operating System is, what its fundamental components are, and distinguish proprietary and open-source OS.
7. Know and master computational logic and logic, logical operations, and truth tables, as well as De Morgan's laws.
8. Know the fundamental principles of binary numbering systems and their implementation in a computer.
9. Become familiar with advanced computing concepts and the future of computing, such as Artificial Intelligence, Quantum Computing, Cloud processing.
10. Develop essential personal and social skills for integration in the business world, namely: teamwork, problem solving, interpersonal skills, initiative, assertive communication, creativity, critical thinking, good time management, adaptability and resilience.

CONTEÚDOS PROGRAMÁTICOS / SYLLABUS

1. LINGUAGENS DE PROGRAMAÇÃO
   1.1. Código máquina e assembly
   1.2. Linguagens compiladas (exemplos)
   1.3. Linguagens interpretadas (exemplos)
   1.4. Tipos de linguagens: imperativas, funcionais, estruturadas, em lógica, orientadas a objetos
2. ALGORITMOS E FLUXOGRAMAS
   2.1. Conceito de algoritmo
   2.2. Exemplo de algoritmo
   2.3. Estruturas de controlo de fluxo em pseudo-linguagem
      2.3.1. Estruturas de sequência
      2.3.2. Estruturas de decisão
      2.3.3. Estruturas de repetição ou loop
   2.4. Fluxogramas
   2.5. Erros mais comuns e validação de entrada de dados
   2.6. Debugging e traçagens de programa
3. INTRODUÇÃO LINGUAGEM DE PROGRAMAÇÃO: PYTHON
   3.1. Caraterísticas da linguagem
   3.2. IDEs para Python
   3.3. Instalação do Wingware 101 (ou Anaconda e Spyder)
   3.4. Execução de código em modo imperativo
   3.5. Exemplos
4. CONCEITOS BÁSICOS ESTRUTURA COMPUTADORES E SISTEMAS OPERATIVOS
   4.1. Evolução do hardware dos computadores (várias gerações)

4.2. Sistemas Operativos

4.2.1. Por lotes e partilha de tempo (time-sharing)

4.2.2. Multiprogramação, Interativos

4.2.3. Memória Virtual

4.2.4. Monotarefa, Monoutilizador, Multitarefa e Multiutilizador

4.2.5. Sistemas distribuídos, Cloud

4.3. Componentes Sistemas Operativos

4.3.1. Núcleo (Kernel)

4.3.2. Processos (em Windows e Unix/Linux)

4.3.3. Gestão de Memória (virtual, paginação, primária e secundária)

4.3.4. Sistema de ficheiros

4.3.5. Gestão de periféricos - Entrada/Saída (E/S ou I/O)

4.4. Mecanismos de Segurança e controlo de acessos

5. INTRODUÇÃO À LÓGICA COMPUTACIONAL

5.1. Operações lógicas (AND, OR, NOT, XOR)

5.2. Exemplo de algoritmo

5.3. Estruturas de controlo de fluxo em pseudo-linguagem

6. SISTEMAS DE NUMERAÇÃO BINÁRIOS

6.1. Conversão de números base 2, 8, 10 e 16

6.2. Técnicas para representação de números binários em computador

6.3. Aritmética binária de inteiros e números flutuantes

7. FUTURO DA COMPUTAÇÃO

7.1. Inteligência Artificial

7.2. Processamento e servidores na Cloud

7.3. Computação Quântica

7.4. IoT

7.5. Cibersegurança e proteção de dados (RPGD)

(English)

1. PROGRAMMING LANGUAGES

1.1. Assembly and Machine code

1.2. Compiled languages (examples)

1.3. Interpreted languages (examples)

1.4. Types of languages: imperative, functional, structured, logical, object-oriented

2. ALGORITHMS AND FLOWCHARTS

2.1. Algorithm Concept

2.2. Algorithm example

2.3. Flow control structures in pseudo-language

2.3.1. Sequence Structures

2.3.2. Decision structures

2.3.3. Repeating or looping structures

2.4. Flowcharts

2.5. Most common errors and data entry validation

2.6. Debugging and program plots

3. INTRODUCTION TO PROGRAMMING: PYTHON

3.1. Language features

3.2. Python IDEs

3.3. Wingware 101 (or Anaconda and Spyder)

3.4. Code execution in imperative mode

3.5. Examples

4. COMPUTERS AND OPERATING SYSTEMS

4.1. Evolution of computer hardware (several generations)

4.2. Operating Systems

4.2.1. By batching and time-sharing

4.2.2. Multiprogramming, Interactive

4.2.3. Virtual Memory

4.2.4. Single-task, Single-user, Multi-task and Multi-user

4.2.5. Distributed Systems, Cloud

4.3. Operating Systems Components

4.3.1. Core (Kernel)

4.3.2. Processes (on Windows and Unix/Linux)

4.3.3. Memory Management (virtual, paging, primary and secondary)

4.3.4. file system

4.3.5. Peripheral Management - Input/Output (I/O or I/O)

4.4. Security Mechanisms and Access Control

5. INTRODUCTION TO COMPUTATIONAL LOGIC

5.1. Logical operations (AND, OR, NOT, XOR)

5.2. Algorithm example

5.3. Flow control structures in pseudo-language

6. BINARY NUMBER SYSTEMS

6.1. Conversion of base numbers 2, 8, 10 and 16

6.2. Techniques for computer representation of binary numbers

6.3. Binary Integer Arithmetic and Floating Numbers

7. FUTURE OF COMPUTING
   7.1. Artificial intelligence
   7.2. Cloud Processing and Servers
   7.3. Quantum Computation
   7.4. IoT
   7.5. Cybersecurity and Data Protection (RPGD)

DEMONSTRAÇÃO DA COERÊNCIA DOS CONTEÚDOS PROGRAMÁTICOS COM OS OBJETIVOS DA UNIDADE CURRICULAR/ DEMONSTRATION OF THE SYLLABUS COHERENCE WITH THE CURRICULAR UNIT'S OBJECTIVES

Esta unidade curricular introduz conceitos fundamentais da Ciência dos Computadores. Dá uma visão geral das linguagens de programação, conceitos básicos de algoritmia e fluxogramas, utilizando a linguagem Python como primeira abordagem à programação. Também é abordada a lógica computacional e os sistemas de numeração binários utilizados pelos computadores.

Nas primeiras aulas serão apresentados conceitos de programação e as principais linguagens existentes, a sua evolução histórica, desde o código máquina e assembler, até à programação orientada por objetos. Seguir-se-á a introdução aos sistemas operativos, com descrição da sua função, evolução histórica, principais componentes. Depois haverá uma introdução a algoritmos e fluxogramas, como metodologias para desenvolvimento de programas. Continua com uma breve introdução à lógica computacional, e aos sistemas de numeração decimal, binário, octal e hexadecimal. Concluí com uma abordagem ao futuro da computação, da Inteligência Artificial à Computação Quântica. O objetivo principal consiste em dar uma perspetiva das metodologias a aplicar para a resolução de problemas através do uso de computadores, no passado, presente e futuro.

(English)

This unity provides the fundamental concepts of Computer Science. It gives an overview of programming languages, basic algorithmic concepts, and flowcharts, using the Python language as a first approach to programming. Computational logic and binary numbering systems used by computers are also discussed.

Starts by introducing the basic programming concepts and listing the main existing languages as well as their historical evolution, from machine code and assembly to object-oriented. An introduction to operating systems will provide a description of its role, historical evolution, and main components. Introduction to using algorithms and flowcharts as methodologies for program will follow. A brief introduction to computational logic, and decimal, binary, octal and hexadecimal numbering systems is the following topic. Concludes with an approach to the future of computing, from Artificial Intelligence to Quantum Computing. The main goal is to give an overview of the methodologies to be applied for solving problems using computers, in the past, present and future.

METODOLOGIAS DE ENSINO E AVALIAÇÃO / TEACHING METHODOLOGIES INCLUDING EVALUATION

As aulas assumirão um carácter teórico-prático, sendo lecionadas num contexto baseado em apresentações teóricas, com exposição de conteúdos suportados em material didático que é disponibilizado aos estudantes na plataforma de LMS.

De acordo com o Regulamento de Funcionamento do ISTEC Porto a avaliação é efetuada através de um exame final obrigatório. Na classificação final, poderão ser considerados elementos de avaliação contínua, tais como testes, trabalhos individuais ou em grupo, assim como a participação nas aulas presenciais e com recursos de aprendizagem proporcionados por sistemas de e-learning. O estudante que realize os trabalhos práticos propostos nas aulas e nas condições aprovadas, poderá prescindir da realização da Prova prática final.

(English)

Classes will assume a theoretical-practical nature, being taught in a context based on theoretical presentations, with exposure of content supported by didactic material that is made available to students on the LMS platform.

According to the ISTEC Porto Operating Regulations, the evaluation is carried out through a mandatory final examination. In the final classification, elements of continuous evaluation may be considered, such as tests, individual or group work, as well as participation in face-to-face classes and learning resources provided by e-learning systems. The student who performs the practical work proposed in the classes and in the approved conditions, may dispense with the completion of the final practical test.

DEMONSTRAÇÃO DA COERÊNCIA DAS METODOLOGIAS DE ENSINO COM OS OBJETIVOS DA UNIDADE CURRICULAR / DEMONSTRATION OF THE COHERENCE BETWEEN THE TEACHING METHODOLOGIES AND THE LEARNING OUTCOMES

A metodologia a utilizar na unidade curricular para as aulas teórico-práticas, tendo por base o conteúdo programático descrito, permite atingir os objetivos enunciados. Verifica-se, assim, a coerência entre os conteúdos e os objetivos de aprendizagem na medida em que: os objetivos dos pontos 1 a 3 são atingidos através do conteúdo 1; o objetivo do ponto 4 é concretizado pelo conteúdo 2; os objetivos 5 a 9 são atingidos pelos conteúdos 3 a 7 respetivamente; o objetivo 10 será concretizado com a realização de trabalhos práticos na aula e a realizar pelos estudantes durante o semestre.

(English)

The methodology to be used in the curricular unit for the theoretical-practical classes, based on the syllabus described, allows achieving the stated objectives. Thus, the coherence between the contents and the learning objectives is verified insofar as: the objectives of points 1 to 3 are achieved through content 1; the objective of point 4 is achieved by content 2; objectives 5 to 9 are achieved by contents 3 to 7 respectively; objective 10 will be achieved with practical work in class and to be carried out by students during the semester.

BIBLIOGRAFIA / BIBLIOGRAPHY

FUNDAMENTAL / ESSENTIAL:

Carriço, A. J. et al. (2008). *Arquitetura e Funcionamento dos Computadores*. Edições Chambel Lda.

Costa, E. (2015). *Programação em Python*. FCA.

Marques, J. A., Ferreira, P., Ribeiro, C., Veiga, L., & Rodrigues, R. (2012). *Sistemas Operativos*. (2ª ed.). FCA.

COMPLEMENTAR / COMPLEMENTARY:

Vasconcelos, J. (2015). *PYTHON – Algoritmia e Programação Web*. FCA.

INTERNET:

Acesso a publicações da especialidade, através da rede SPRINGER:

https://link.springer.com/

Os slides das aulas serão disponibilizados pelo docente no Moodle. / The notes given during the class will be made available on Moodle by the lecturer.