LICENCIATURA: Engenharia Informática	**ÁREA CIENTÍFICA:** Engenharia Informática
UNIDADE CURRICULAR/CURRICULAR UNIT: Algoritmos e Estruturas de Dados / Algorithms and data structures	**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. Saber desenvolver algoritmos no contexto da resolução de problemas utilizando programas informáticos;
2. Compreender, saber implementar e manipular estruturas de dados do tipo array;
3. Compreender, saber implementar e manipular pilhas e filas de espera;
4. Compreender e saber implementar técnicas de programação recursivas;
5. Compreender, saber implementar e manipular linked lists (listas encadeadas);
6. Compreender, saber implementar e manipular árvores;
7. Saber utilizar e aplicar as estruturas de dados dos tipos indicados em aplicações informáticas, para a resolução de problemas;
8. Realizar projetos de trabalho em grupo que permitam desenvolver capacidades e atitudes de responsabilização, de solidariedade e de cooperação;
9. Utilizar, de forma criativa e crítica, conhecimentos, capacidades e atitudes na criação de algoritmos e utilização de estruturas de dados, para a resolução de problemas aplicados a exemplos concretos, integrando os conhecimentos adquiridos;
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;
11. Manifestar abertura e curiosidade face à ciência da computação, numa atitude de constante aprendizagem com integração de capacidades de saber fazer, atitudes e conhecimento.

(English)

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

1. Know how to develop algorithms in the context of problem-solving using computer programs;

2. Understand, know how to implement and manipulate array-type data structures;
3. Understand, know how to implement and manipulate stacks and queues;
4. Understand and know how to implement recursive programming techniques;
5. Understand, know how to implement and manipulate linked lists;
6. Understand, know how to implement and manipulate trees;
7. Know how to use and apply data structures of the types indicated in computer applications, to solve problems;
8. Carry out group work projects that allow the development of capacities and attitudes of responsibility, solidarity and cooperation;
9. Use, in a creative and critical way, knowledge, skills and attitudes in the creation of algorithms and use of data structures, to solve problems applied to concrete examples, integrating the acquired knowledge;
10. Develop essential personal and social skills for integration into the business world, namely: teamwork, problem solving, interpersonal relationships, initiative, assertive communication, creativity, critical thinking, good time management, adaptability and resilience;
11. Express openness and curiosity towards computer science, in an attitude of constant learning with the integration of know-how, attitudes and knowledge.

CONTEÚDOS PROGRAMÁTICOS / SYLLABUS

1. Introdução à Algoritmia e Estruturas de Dados
2. Vetores – Array
   - 2.1. Introdução aos vetores
   - 2.2. Implementação de um vetor
   - 2.3. Adicionar, alterar e remover elementos de um vetor
   - 2.4. Ordenação de um vetor
   - 2.5. Pesquisa sequencial e pesquisa binária ou dicotómica
3. Pilhas e Filas de Espera
   - 3.1. O que são Pilhas
   - 3.2. Implementar uma Pilha
   - 3.3. Adicionar, alterar e remover elementos a uma Pilha
   - 3.4. O que são Filas de Espera
   - 3.5. Implementar uma Fila de Espera
   - 3.6. Adicionar, alterar e remover itens a uma fila de espera
4. Recursividade – Conceitos e técnicas
   - 4.1. Recursividade – Conceitos base
   - 4.2. Função Fatorial
   - 4.3. Recursão excessiva - Fibonacci Numbers
   - 4.4. Tail Recursion
   - 4.5. Problemas na utilização de funções recursivas

5. Listas encadeadas
   5.1. Listas simplesmente encadeadas
   5.2. Criar uma lista simplesmente encadeada
   5.3. Percurso e localização de um item
   5.4. Adicionar, alterar e remover itens em qualquer ponto de uma lista simplesmente encadeada
   5.5. Listas duplamente encadeadas
   5.6. Implementar uma lista duplamente encadeada
   5.7. Adicionar, alterar e remover itens de uma lista duplamente encadeada
   5.8. Listas encadeadas circulares
   5.9. Implementar uma lista encadeada circular
   5.10. Adicionar, alterar ou remover um conjunto de itens a uma lista encadeada circular
6. Árvores
   6.1. Conceitos base
   6.2. Representação de árvores
   6.3. Árvores binárias
   6.4. Árvores de ordenação binária
   6.5. Atravessamento de uma árvore
   6.6. Localizar e inserir itens em árvores de ordenação binária
   6.7. Implementar uma árvore de ordenação binária
   6.8. Inserir um nó numa árvore de ordenação binária
   6.9. Verificar se existe um dado nó numa árvore de ordenação binária
   6.10. Calcular o número de nós da árvore de ordenação binária

(English)

1. Introduction to Algorithms and Data Structures
2. Vectors - Arrays
   2.1. Introduction to vectors
   2.2. Implementing a vector
   2.3. Adding, Changing, and Removing Elements from a Vector
   2.4. Sorting a vector
   2.5. Sequential search and binary or dichotomous search
3. Stacks and Queues
   3.1. What are Stacks
   3.2. Implement a Stack
   3.3. Adding, Changing, and Removing Elements to a Stack
   3.4. What are Queues?

3.5. Implement a Queue

3.6. Adding, Changing, and Removing Items from a Queue

4. Recursion - Concepts and techniques

4.1. Recursion - Basic Concepts

4.2. Factorial Function

4.3. Excessive recursion - Fibonacci Numbers

4.4. Tail recursion

4.5. Problems using recursive functions

5. Linked lists

5.1. Simply linked lists

5.2. Create a simply linked list

5.3. Path and location of an item

5.4. Add, change and remove items at any point in a simply linked list

5.5. Doubly linked lists

5.6. Implement a doubly linked list

5.7. Add, change and remove items from a doubly linked list

5.8. Circular linked lists

5.9. Implement a circular linked list

5.10. Add, change, or remove a set of items to a circular linked list

6. Trees

6.1. Basic concepts

6.2. Tree representation

6.3. Binary trees

6.4. Binary Sort Trees

6.5. Crossing a tree

6.6. Find and insert items into binary sort trees

6.7. Implement a binary sort tree

6.8. Insert a node in a binary tree

6.9. Check if a given node exists in a binary tree

6.10. Calculate the number of nodes in the binary tree

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

Nesta unidade curricular os estudantes deverão adquirir conhecimentos teórico-práticos em Algoritmos e Estruturas de Dados, com ênfase em estruturas do tipo array, pilhas, listas e árvores que lhes permitam aplicar estes conceitos-chave da ciência da computação no desenvolvimento de aplicações informáticas. Também deverão aprender os princípios básicos inerentes à recursividade e técnicas de programação recursiva, como aplicar este conceito na elaboração de algoritmos com ciclos de pesquisas

e / ou computação, para que sejam capazes de avaliar e saber optar quando e onde devem utilizar técnicas de programação recursiva ou ciclos de repetição iterativa, no contexto do desenvolvimento de programas de computação.

Verifica-se, assim, a coerência entre os conteúdos e os objetivos de aprendizagem na medida em que: o objetivo 1 é atingido através do conteúdo 1; os conteúdos dos pontos 2.1 a 6 concretizam os objetivos 2, 3, 4, 5, e 6; os objetivos 7, 8, 9, 10 e 11 serão concretizados com a realização de trabalhos práticos na aula e a realizar pelos estudantes durante o semestre.

(English)

In this curricular unit, students should acquire theoretical and practical knowledge in Algorithms and Data Structures, with emphasis on array-like structures, stacks, lists and trees that allow them to apply these key concepts of computer science in the development of computer applications. They should also learn the basic principles inherent to recursion and recursive programming techniques, how to apply this concept in the development of algorithms with research cycles and / or computation, so that they are able to evaluate and know how to choose when and where to use recursive programming techniques. or iteratively repeating cycles, in the context of developing computer programs. Point 1 allows the first objective to be achieved. The following points up to and including 6 achieve the objectives with the same numbers. The remaining objectives will be achieved with the completion of practical work in class and to be carried out by students during the semester.

Thus, the coherence between the contents and the learning objectives is verified insofar as: objective 1 is reached through content 1; the contents of points 2.1 to 6 fulfill objectives 2, 3, 4, 5, and 6; objectives 7, 8, 9, 10 and 11 will be achieved with practical work in class and to be carried out by students during the semester.

METODOLOGIAS DE ENSINO E AVALIAÇÃO / TEACHING METHODOLOGIES INCLUDING EVALUATION

As aulas desta unidade curricular são de natureza teórico-prática. Em todas as aulas, exercícios de aplicação prática dos algoritmos e das estruturas de dados complementam a exposição teórica dos conceitos apresentados. Esta metodologia permite que os alunos adquiram, não apenas os conhecimentos teóricos, mas também as necessárias competências para aplicar as estruturas de dados a situações práticas simuladas.

De acordo com o Regulamento de Funcionamento do ISTEC-Porto a avaliação é efetuada através de um exame escrito individual e 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 em recursos de aprendizagem proporcionados por sistemas de e-learning.

(English)

The classes of this curricular unit are of theoretical and practical nature. Are foreseen 60 contact hours. In all classes, practical application exercises of algorithms and data structures complement the theoretical exposition of those presented. This

methodology allows students to acquire, not only the theoretical knowledge, but also the necessary skills to apply the data structures to simulated practical situations.

According to the ISTEC-Porto Regulation, the assessment is made through an individual and compulsory written examination. In the final classification, elements of continuous assessment may be considered, such as tests, individual or group work, as well as the participation in the presential classes and in learning resources provided by e-learning systems.

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 usada na unidade curricular visa contribuir para que o estudante aprofunde os seus conhecimentos e capacidades

no domínio dos princípios básicos inerentes à recursividade e técnicas de programação recursiva, como aplicar este conceito na elaboração de algoritmos com ciclos de pesquisas e / ou computação, para que sejam capazes de avaliar e saber optar quando e onde devem utilizar técnicas de programação, no contexto do desenvolvimento de programas de computação.

A obtenção dos objetivos da unidade curricular é assegurada pela natureza teórico-prática das aulas da unidade curricular que são planeadas para permitir a compreensão teórica e a aplicação prática dos conceitos, partindo das estruturas de dados mais simples para as construções mais complexas.

(English)

The methodology used in the course unit aims to contribute to the student deepen their knowledge and skills

in the domain of the basic principles inherent to recursion and recursive programming techniques, how to apply this concept in the development of algorithms with search and/or computing cycles, so that they are able to evaluate and to know how to choose when and where to use programming techniques, in the context of the development of computing programs.

The attainment of the objectives of the course unit is assured by the theoretical-practical nature of the classes of the curricular unit that are planned to allow the theoretical and practical understanding of the concepts, starting from the simplest data structures to the most complex constructions.

BIBLIOGRAFIA / BIBLIOGRAPHY

FUNDAMENTAL / ESSENTIAL:

Rocha, A. A. (2014). *Estruturas de Dados e Algoritmos em C*, 3a. Edição. FCA.

Shaffer, C. A. (2011). *Data Structures & Algorithm Analysis in Java*. Courier Corporation.

COMPLEMENTAR/ COMPLEMENTARY:

Goodrich, M. T., Tamassia, R. & Mount, D.M. (2011). *Data Structures and Algorithms in C++*, Second Edition. John Wiley & Sons.

Neto, J. P. (2004*). Programação. Algoritmos e Estruturas de Dados*. Escolar Editora.

INTERNET:

Acesso a publicações da especialidade, gratuitamente, através da rede SPRINGER:

https://link.springer.com/