UNIVERSIDADE DO ALGARVE

English version at the end of this document

Ano Letivo	2019-20
Unidade Curricular	ALGORITMOS E ESTRUTURAS DE DADOS
Cursos	ENGENHARIA INFORMÁTICA (1.º ciclo)
Unidade Orgânica	Faculdade de Ciências e Tecnologia
Código da Unidade Curricular	14781051
Área Científica	CIÊNCIA DE COMPUTADORES
Sigla	
Línguas de Aprendizagem	Português
Modalidade de ensino	Ensino presencial
Docente Responsável	Pedro João Valente Dias Guerreiro

DOCENTE	TIPO DE AULA	TURMAS	TOTAL HORAS DE CONTACTO (*)
Pedro João Valente Dias Guerreiro	PL; T	T1; PL1	30T; 30PL
Maria Margarida da Cruz Silva Andrade Madeira e Carvalho de Moura	PL	PL3	30PL
Noélia Susana Costa Correia	PL	PL2	30PL

* Para turmas lecionadas conjuntamente, apenas é contabilizada a carga horária de uma delas.

ANO	PERÍODO DE FUNCIONAMENTO*	HORAS DE CONTACTO	HORAS TOTAIS DE TRABALHO	ECTS
2º	S2,S1	30T; 30PL	168	6

* A-Anual;S-Semestral;Q-Quadrimestral;T-Trimestral

Precedências

Sem precedências

Conhecimentos Prévios recomendados

Idealmente, os alunos terão realizado com êxito as cadeiras de programação antecedentes: Programação Imperativa e Laboratório de Programação.

Objetivos de aprendizagem (conhecimentos, aptidões e competências)

No fim desta cadeira, todos os alunos deverão ser capazes de:

- Compreender as estruturas de dados fundamentais: vetores, listas, filas, pilhas, árvores, tabelas de dispersão, ?union-find?.
- Compreender os principais algoritmos de ordenação.
- Compreender os principais algoritmos sobre grafos.
- Compreender os principais algoritmos sobre cadeias de caracteres e vetores de cadeias de caracteres.
- Saber utilizar bibliotecas que implementam essas estruturas de dados e esses algoritmos.
- Programar essas estruturas de dados, adaptando-as a necessidades supervenientes e acrescentando novas funcionalidades.
- Distinguir as características de complexidade dos principais algoritmos.
- Compreender e saber aplicar as ideias fundamentais da programação dinâmica.
- Apreciar melhor a problemática do desenvolvimento de software e a importância da qualidade do software.
- Programar em Java, tirando partido do mecanismos das classes.

Além disto, os alunos terão reforçado as suas competências gerais de programação.

Conteúdos programáticos

- Preliminares
  - Programação em Java
  - Programação com classes
  - Modelo de programação
- Conceitos fundamentais
  - Sacos, pilhas e filas
  - *Union-Find*
  - Análise de algoritmos
- Ordenação
  - Algoritmos elementares
  - Mergesort, quicksort
  - Filas com prioridade
- Busca
  - Árvores binárias de busca
  - Árvores equilibradas
  - Tabelas de dispersão
- Grafos
  - Busca em profundidade
  - Busca em largura
  - Árvores de cobertura
  - Caminho mais curto
- Estratégias programativas
  - Divisão-conquista
  - Algoritmos vorazes ("gananciosos")
  - Programação dinâmica
- Outros assuntos
  - Algoritmos sobre cadeias de caracteres

Demonstração da coerência dos conteúdos programáticos com os objetivos de aprendizagem da unidade curricular

A cadeira Algoritmos e Estruturas de Dados existe em todas as licenciaturas em Engenharia Informática, com este nome ou com um nome equivalentes. Os conteúdos programáticos são os conteúdos consagrados: algoritmos básicos, algoritmos de ordenação, árvores binárias, tabelas de dispersão, algoritmos sobre grafos, complexidade, algoritmos sobre cadeias de caracteres e sobre coleções de cadeias de caracteres, programação dinâmica. Cada um destes tópicos terá relativamente mais ou menos ênfase, consoante a receptividade demonstrada pelos alunos, em cada edição.

Metodologias de ensino (avaliação incluída)

Nas aulas teóricas o professor faz a exposição da matéria usando o quadro e o computador para apresentar os conteúdos e fazer demonstrações.

Nas aulas práticas, os alunos resolvem problemas de programação ou realizam trabalhos mais longos.

A avaliação usa a modalidade de *avaliação por frequência* . O exame assume a forma de uma prova escrita, com suporte computacional.

A parte da avaliação feita ao longo do funcionamento da unidade curricular é realizada por meio de exercícios, trabalhos práticos e problemas de programação.

São admitidos ao exame os alunos apenas os alunos com nota superior ou igual a 7.5 na avaliação ao longo do funcionamento.

A nota do exame tem peso 70% na nota final se for maior ou igual a 8.5 e tem peso 100% se não.

Todas as ações que fazem parte da avaliação são realizadas individualmente, ao abrigo do código de honra da cadeira.

Demonstração da coerência das metodologias de ensino com os objetivos de aprendizagem da unidade curricular

Nas aulas teóricas, em anfiteatro, para a turma inteira, o professor discute os sucessivos tópicos de Algoritmos e Estruturas de Dados, pela ordem mais profícua, enfatizando as boas práticas e chamando a atenção para as ratoeiras. Isso é feito com auxílio de apresentações PowerPoint e com demonstrações ao vivo de programas que incorporam os assuntos em análise.

Nas aulas práticas, para turmas mais pequenas, os alunos resolvem problemas que exercitam os conceitos concomitantemente discutidos nas aulas práticas, para reforçar a interiorização desses conceitos e para desenvolver as competências gerais de programação.

Os professores observam o desempenho dos alunos, orientam e intervêm para assegurar que os programas escritos pelos alunos têm as características pretendidas, em função do avanço da cadeira.

Bibliografia principal

Algorithms, quarta edição, Robert Sedgewick e Kevin Wayne, 2011
Introduction to Algorithms, 3.ª edição, Thomas Cormen, Charles Leiserson, Ronald Rivest, Clifford Stein, 2009.
Slides usados nas aulas teóricas.

Academic Year	2019-20
Course unit	ALGORITHMS AND DATA STRUCTURES
Courses	INFORMATICS (COMPUTER SCIENCE) (1st Cycle)
Faculty / School	FACULTY OF SCIENCES AND TECHNOLOGY
Main Scientific Area	CIÊNCIA DE COMPUTADORES
Acronym	
Language of instruction	Portuguese
Teaching/Learning modality	face to face learning
Coordinating teacher	Pedro João Valente Dias Guerreiro

Teaching staff	Type	Classes	Hours (*)
Pedro João Valente Dias Guerreiro	PL; T	T1; PL1	30T; 30PL
Maria Margarida da Cruz Silva Andrade Madeira e Carvalho de Moura	PL	PL3	30PL
Noélia Susana Costa Correia	PL	PL2	30PL

* For classes taught jointly, it is only accounted the workload of one.

Contact hours

T	TP	PL	TC	S	E	OT	O	Total
30	0	30	0	0	0	0	0	168

T - Theoretical; TP - Theoretical and practical ; PL - Practical and laboratorial; TC - Field Work; S - Seminar; E - Training; OT - Tutorial; O - Other

Pre-requisites

no pre-requisites

Prior knowledge and skills

Ideally, students will have successfully completed the previous programming courses in the curriculum: Imperative Programming and Programming Lab.

The students intended learning outcomes (knowledge, skills and competences)

At the end of this course, students should be able to:

- Understand the fundamental data structures: vectors, lists, queues, stacks, trees, graphs, union-find.
- Understand the main sorting algorithms.
- Understand the main algorithms on graphs.
- Understand the main algorithms on strings and on arrays of strings.
- Know how to use libraries that implement these data structures and these algorithms.
- Program these data structures, adapting them to future needs and adding new functionalities.
- Distinguish the complexity characteristics of the main algorithms.
- Understand and apply the fundamental ideas of dynamic programming.
- Appreciate the software development life cycle and the importance of software quality.
- Program in Java, using classes.

In addition, students will have strengthened their overall programming skills.

Syllabus

- Introduction
  - Programming in Java
  - Programming with classes
  - Programming model in the course
- Fundamental concepts
  - Bags, stacks and queues
  - Union-Find
  - Analysis of algorithms
- Sorting
  - Elementary algorithms
  - Mergesort, quicksort
  - Priority queues
- Searching
  - Binary search trees
  - Balanced trees
  - Hash tables
- Graphs
  - Depth-first search
  - Breadth-first search
  - Spanning trees
  - Shortest paths
- Programming strategies
  - Divide and conquer
  - Greedy Algorithms
  - Dynamic programming
- Other topics
  - Algorithms for strings

Demonstration of the syllabus coherence with the curricular unit's learning objectives

All Computer Science programmes have a course unit devoted to Algorithms and Data Structures, with this name or with an equivalent one. The proposed syllabus is a standard one: it covers basic algorithms, sorting algorithms, binary trees, hash tables, binary trees, graphs, complexity, algorithms on strings and on collections of strings, dynamic programming. Each of these topics will be relatively more or less emphasized, depending on the receptivity shown by the students, in each edition.

Teaching methodologies (including evaluation)

In the lectures, the teacher presents the topics of the course, using the projector, the whiteboard, and his computer for demonstrations.

In practical classes, students solve programming problems or perform longer assignments.

The evaluation uses the "frequency evaluation" modality. The exam takes the form of a written test, with computational support.

The part of the evaluation made throughout the course unit is carried out through exercises, and programming problems.

Students with a mark of 7.5 or higher in the evaluation during the course of the course are admitted to the exam.

The exam mark has a weight of 70% on the final grade if it is greater than or equal to 8.5 and has a weight of 100% if not.

All actions that are part of the evaluation are performed individually, under the honor code of the course unit.

Demonstration of the coherence between the teaching methodologies and the learning outcomes

In the lectures, which happen in lecture hall or in large classrooms, for the whole class, the teacher discusses the successive topics of Algorithms and Data Structures, incrementally, emphasizing good practices and drawing attention to the traps. This is done with the aid of PowerPoint presentations and with live demonstrations of programs that incorporate the concepts under analysis.

In the labs, for smaller classes, students solve problems that exercise the concepts concomitantly discussed in the practical classes, to reinforce the understanding of these concepts and to develop general programming skills.

Teachers monitor students' performance, guide and intervene to ensure that the written programs of the students have the desired characteristics.

Main Bibliography

Algorithms, 4th ed., Robert Sedgewick e Kevin Wayne, 2011.
Introduction to Algorithms, 3rd ed., Thomas Cormen, Charles Leiserson, Ronald Rivest, Clifford Stein, 2009.
Lecture slides, provided by the course staff.