Ficha de Unidade Curricular – (Versão A3ES 2018-2023)

1. Caracterização da Unidade Curricular.

1.1. Designação da unidade curricular *(1.000 carateres).*
Programação II

1.2. Sigla da área científica em que se insere *(100 carateres).*
IC

1.3. Duração[1] *(100 carateres).*
Semestral

1.4. Horas de trabalho[2] *(100 carateres).*
162

1.5. Horas de contacto[3] *(100 carateres).*
67,5 (T: 45; PL: 22,5)

1.6. ECTS *(100 carateres).*
6

1.7. Observações[4] *(1.000 carateres).*

1.7. Remarks *(1.000 carateres).*

2. Docente responsável e respetiva carga letiva na Unidade Curricular *(preencher o nome completo) (1.000 carateres).*
Mário Henrique Carrasqueira Simões - 67,5 horas de contacto

3. Outros docentes e respetivas cargas letivas na unidade curricular *(1.000 carateres).*
Miguel Pinto Campilho Gomes - 67,5 horas de contacto
Mário Justiniano Morais Pinheiro - 67,5 horas de contacto

4. Objetivos de aprendizagem (conhecimentos, aptidões e competências a desenvolver pelos estudantes). *(1.000 carateres)*.
Os estudantes que terminam com sucesso esta unidade curricular serão capazes de:
1. Implementar, testar e corrigir programas de complexidade média, contruídos em vários módulos escritos em linguagem C, podendo envolver o uso bibliotecas;
2. Relacionar os mecanismos da linguagem C com o modelo computacional que suporta a execução;
3. Desenvolver algoritmos e estruturas de dados dinâmicas, recorrendo à definição de tipos e ao alojamento dinâmico de memória;
4. Desenvolver e utilizar implementações genéricas de algoritmos, aplicáveis a diversos tipos de dados, com recurso à parametrização das operações a realizar;
5. Identificar e explorar as funcionalidades das bibliotecas normalizadas e criar novas bibliotecas;
6. Descrever como os programas são compilados, ligados, carregados e executados.

4. Intended learning outcomes (knowledge, skills and competences to be developed by the students). *(1.000 characters).*
Students who successfully complete this course unit will be able to:
1. Implement, test and correct programs of medium complexity, built in several modules written in C language, and that may involve the use of libraries;
2. Relate the mechanisms of the C language to the computational model that supports execution;
3. Develop algorithms and dynamic data structures, using type definition and dynamic memory accommodation;

4. Develop and use generic implementations of algorithms, applicable to several types of data, using the parameterization of the operations to be performed;
5. Identify and exploit the functionalities of standard libraries and create new libraries;
6. Describe how programs are compiled, linked, loaded, and executed.

5. Conteúdos programáticos *(1.000 carateres).*

I. Gestão da memória nos programas escritos em linguagem C (alojamento permanente, variáveis automáticas e alojamento dinâmico); Revisões de ponteiros e sua aritmética; Programação genérica; Programação recursiva.
II. Organização dos programas em módulos; Compilação, ligação e execução de programas; Conceito de pré-compilação; Criação e utilização de macros e de ficheiros para inclusão; Automatização do processo de compilação e ligação.
III. Utilização da biblioteca normalizada, famílias de funções e suas funcionalidades; Ligação estática ou dinâmica; Utilização de outras bibliotecas; criação de bibliotecas.
IV. Algoritmia e estruturas de dados; Criação dinâmica de estruturas de dados – arrays com alojamento dinâmico, listas ligadas, árvores binárias e hash-table. Algoritmos de ordenação; inserção e remoção ordenadas; balanceamento das árvores binárias.

5. Syllabus *(1.000 characters).*

I. Memory management in programs written in C language (permanent housing, automatic variables and dynamic housing); Revisions of pointers and their arithmetic; Generic programming; Recursive programming.
II. Organization of programs in modules; Compilation, linkage and execution of programs; Pre-compilation concept; Creating and using macros and inclusion header files; Automation of the compilation and linkage process.
III. Use of the standard library, function families and their functionalities; Static or dynamic linkage; Use of other libraries; creation of libraries.
IV. Algorithms and data structures; Dynamic creation of data structures - arrays with dynamic allocation, linked lists, binary trees and hash-table. Sorting algorithms; insertion and removal; binary tree balancing.

6. Demonstração da coerência dos conteúdos programáticos com os objetivos de aprendizagem da unidade curricular *(1.000 carateres).*

Os temas são introduzidos por ordem crescente de complexidade; Os objetivos 1 e 2, de maior relevância, são suportados pelo somatório dos conhecimentos adquiridos em todos os temas.
O objetivo 3 é suportado pelo tema IV. O objetivo 4 é suportado pelos temas I e III. O objetivo 5 é suportado pelos temas II e III. O objetivo 6 é suportado pelo tema II.

6. Evidence of the syllabus coherence with the curricular unit's intended learning outcomes *(1.000 characters).*

Topics are introduced in increasing order of complexity; Objectives 1 and 2, of greater relevance, are supported by the sum of the knowledge acquired in all themes. Objective 3 is supported by the theme IV. Objective 4 is supported by the themes I and III. Objective 5 is supported by the themes II and III. Objective 6 is supported by the theme II.

7. Metodologias de ensino (avaliação incluída) *(1.000 carateres).*

O ensino é teórico-prático, estando previstas 30 aulas durante o semestre (67,5 horas de contacto - 15 aulas de 1,5 horas e 15 aulas de 3 horas, das quais são usadas 5 aulas de 3 horas para o apoio aos trabalhos práticos).
As aulas destinam-se à apresentação dos temas e de exemplos práticos de aplicação. A aprendizagem é complementada por trabalhos práticos, sobre os temas principais, realizados em grupo com o máximo de 3 alunos, maioritariamente fora das horas de contacto.
A avaliação é formada por um teste escrito (T), por três mini-testes relativos aos temas dos trabalhos práticos, de que se apura a média dos dois melhores (MT), e pela discussão oral dos trabalhos práticos (P). A discussão é realizada em grupo, sendo atribuídas classificações individuais em função do domínio demonstrado. A ponderação da classificação final (CF) é: CF = 0,4 x T + 0,4 P + 0,2 MT;
Os mínimos para aprovação são, na escala de 20 valores: CF >= 10; T >= 9,5; MT >= 8,0; P >= 9,5.

7. Teaching methodologies (including assessment) *(1.000 characters).*

Teaching is theoretical and practical, with 30 classes scheduled during the semester (67.5 contact hours - 15 lessons of 1.5 hours and 15 lessons of 3 hours, of which 5 lessons of 3 hours are used to support the practical work).
The classes are designed to present topics and practical examples of application. The learning is complemented by practical work, on the main themes, carried out in groups with a maximum of 3 students, mostly outside the contact hours.

The evaluation consists of a written test (T), three mini-tests related to the subjects of the practical assignments, from witch is used the average of the two best (MT), and the oral discussion of the practical work (P). The discussion is carried out in a group, being assigned individual classifications according to the demonstrated domain. The final classification (CF) is: CF = 0.4 x T + 0.4 P + 0.2 MT;
The minimums for approval are, in the scale of 20 values: CF> = 10; T> = 9.5; MT> = 8.0; P = 9.5.

8. Demonstração da coerência das metodologias de ensino com os objetivos de aprendizagem da unidade curricular *(3.000 carateres).*

Os alunos obtiveram em unidades curriculares anteriores os conceitos gerais de programação, usando a linguagem C, e de arquitetura de computadores.
Nas aulas são apresentados os temas, de I a IV, com vista ao domínio dos objetivos referidos. A exposição é complementada, em interação com os alunos, pela realização, teste e correção de programas exemplificativos dos temas expostos. Estes programas, além de concretizarem os conceitos de programação lecionados são usados para explorar as ferramentas de programação, com ênfase na metodologia de depuração de erros.
Os trabalhos práticos, a realizar pelos alunos, têm complexidade crescente e são coordenados com os temas das aulas, que fornecem conceitos de base para a sua realização. Nos trabalhos, os alunos confrontam-se com problemas e dúvidas que, ao elaborar soluções, fazem desenvolver as suas competências.
O plano de trabalhos práticos está organizado em séries de exercícios (SE) organizadas por grupos de temas: SE1, temas I e II; SE2, temas III; SE3, tema IV, usando recursos dos temas II e III. O agrupamento dos temas nos trabalhos promove os objetivos de aprendizagem de acordo com a relação indicada em 6.

8. Evidence of the teaching methodologies coherence with the curricular unit's intended learning outcomes *(3.000 characters).*

The students obtained in previous curricular units the general concepts of programming, using the C language, and of computer architecture.
In the classes are presented the themes, from I to IV, with a view to the mastery of the referred objectives. The exhibition is complemented, in interaction with the students, by the accomplishment, test and correction of programs exemplifying the exposed themes. These programs, in addition to realizing the programming concepts taught, are used to explore the programming tools, with an emphasis on the methodology of debugging.
The practical work, to be carried out by the students, has increasing complexity and is coordinated with the themes of the classes, which provide basic concepts for its accomplishment. In the works, the students develop their competences though the contact with the problems raised by the exercises.
The practical work plan is organized in series of exercises (SE) organized by groups of themes: SE1, themes I and II; SE2, themes III; SE3, theme IV, using resources from themes II and III. The grouping of the themes in the works promotes the learning objectives according to the relation indicated in 6.

9. Bibliografia de consulta/existência obrigatória *(1.000 carateres).*

The C Programming Language, Second Edition; Dennis M. Ritchie, Brian W. Kernighan; Prentice Hall, 1988; ISBN: 9780133086249

Manuais das ferramentas de programação em linguagem C – gcc, ld, as, ar, insight, GNU

---

[1] Anual, semestral, trimestral, …

[2] Número total de horas de trabalho.

[3] Discriminadas por tipo de metodologia adotado (T - Ensino teórico; TP - Ensino teórico-prático; PL - Ensino prático e laboratorial; TC - Trabalho de campo; S - Seminário; E - Estágio; OT - Orientação tutorial; O - Outro).

[4] Assinalar sempre que a unidade curricular seja optativa.