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

1. Caracterização da Unidade Curricular.

1.1. Designação da unidade curricular ***(1.000 carateres).***
Programação em Sistemas Computacionais / Computer Systems Programming

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 h

1.5. Horas de contacto[3] ***(100 carateres).***
TP - 67,5 h

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).***
Ezequiel Augusto Cachão Conde

3. Outros docentes e respetivas cargas letivas na unidade curricular ***(1.000 carateres).***

4. Objetivos de aprendizagem (conhecimentos, aptidões e competências a desenvolver pelos estudantes). ***(1000 carateres)***.
Os estudantes que terminam com sucesso esta unidade curricular serão capazes de:

1. Compreender o modelo de execução de programas escritos em linguagens de alto nível nos sistemas computacionais reais;
2. Conceber e implementar programas de complexidade moderada em linguagem C;
3. Escrever módulos de *software* em *assembly* para integrar em programas desenvolvidos em linguagem C;
4. Compreender a arquitetura básica das caches de acesso à memória RAM;
5. Desenvolver software usando o modelo de compilação separada;
6. Compreender as diferenças entre a ligação estática e dinâmica de módulos;
7. Produzir, depurar e testar *software* modular para ambientes nativos.

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. Understand the execution model of programs written in high-level languages in real computer systems;
2. Design and implement programs of moderate complexity in C language;
3. Write software modules in assembly to integrate into programs developed in C language;
4. Understand the basic architecture of RAM caches;
5. Develop software using the separate build model;
6. Understand the differences between the static and dynamic connection of modules;
7. Produce, debug and test modular software for native environments.

5. Conteúdos programáticos ***(1.000 carateres).***

I. Linguagem C: *arrays*, estruturas, ponteiros e sua aritmética.

II. Escrita de programas em *assembly*. Convenção de chamada a funções: convenções de chamada da linguagem C. Programas envolvendo código em C e *assembly*. Percurso e manipulação de *stack frames*.
III. Hierarquia de memória. Noção de *cache*. Organização de cache de acesso a RAM e impacto no desempenho dos programas.
IV. Construção modular de programas: pré-processador; compilação separada; ficheiros cabeçalho e objeto; ligação estática; bibliotecas estáticas.
V. Implementação de sistema de alocação dinâmica de memória sem reciclagem automática.
VI. Ligação dinâmica de código. Construção e utilização de bibliotecas de ligação dinâmica. Ligação dinâmica em tempo de carregamento e em tempo de execução.

5. Syllabus ***(1.000 characters).***

I. Language C: arrays, structures, pointers and their arithmetic.
II. Write programs in assembly. Function-calling convention: C-language conventions. Programs involving code in C and assembly. Path and manipulation of stack frames.
III. Memory hierarchy. Cache notion. Organization of RAM access cache and impact on program performance.
IV. Modular construction of programs: pre-processor; separate compilation; header and object files; static connection; static libraries.
V. Implementation of dynamic memory allocation system without automatic recycling.
VI. Dynamic code binding. Building and using dynamic link libraries. Dynamic binding at load time and at run time.

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

Nesta unidade curricular, os estudantes compreendem o modelo de execução de programas escritos em linguagens de alto nível no sistema computacional real (pontos II, III, V e VI dos conteúdos programáticos) e adquirem prática de desenvolvimento de aplicações modulares em linguagem C (pontos I e IV). Tais capacidades são fundamentais para a progressão nas áreas de sistemas operativos e de sistemas embebidos. Na componente prática relativa aos pontos IV e VI, os alunos desenvolvem aplicações, em linguagem C, utilizando bibliotecas *open-source standard*. Finalmente, a introdução às *caches* de acesso a RAM (ponto III) suporta estudos posteriores na área da programação concorrente.

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

In this curricular unit, students understand the execution model of programs written in high-level languages in the real computational system (points II, III, V and VI of the programmatic contents) and acquire the practice of developing modular applications in C language (points I and IV). Such capabilities are critical to progression in the areas of operating systems and embedded systems. In the practical component on points IV and VI, students develop applications, in C language, using standard open-source libraries. Finally, the introduction to RAM access caches (point III) supports further studies in the area of concurrent programming.

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

Ensino teórico-prático, estando previstas 30 aulas durante o semestre a que correspondem 67,5 horas de contacto (15 aulas de 3 horas e 15 de 1,5 horas). O tempo total de trabalho do estudante é de 162 horas. As aulas destinam-se à apresentação dos temas e de exemplos práticos de aplicação. Os tópicos principais são ainda explorados através da realização de trabalhos práticos em grupo.
Os resultados da aprendizagem são avaliados individualmente através do teste escrito e na discussão final dos trabalhos de grupo.
A nota final da UC é atribuída individualmente, sendo o resultado da soma da nota da avaliação escrita com a nota atribuída na discussão dos trabalhos práticos (na gama -3 a +3 valores).

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

Theoretical-practical teaching, with 30 classes planned during the semester corresponding to 67.5 contact hours (15 lessons of 3 hours and 15 of 1,5 hours). The total work time of the student is 162 hours. The classes are designed to present topics and practical examples of application. The main topics are further explored through practical group work.
Learning outcomes are assessed individually through written test and final discussion of group work.
The final grade of the UC is assigned individually, being the result of adding the grade of the written evaluation with the grade assigned in the discussion of the practical assignments (in the range -3 to +3 points).

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

Os conteúdos programáticos são expostos nas aulas teórico-práticas, complementando uma apresentação interativa das matérias com a realização, pelos estudantes, de pequenos exercícios de consolidação das mesmas.

As competências indicadas nos pontos 2, 3, 5, 6 e 7 dos objetivos de aprendizagem são desenvolvidas na realização dos trabalhos de grupo.

São dedicadas aulas teórico-práticas para apoio ao desenvolvimento dos trabalhos de grupo, nomeadamente no que se refere à utilização das bibliotecas *open-source* utilizadas na UC.

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

The syllabus contents are presented in the theoretical-practical classes, complementing an interactive presentation of the subjects with the accomplishment, by the students, of small exercises of consolidation of the same ones.

The competences indicated in points 2, 3, 5, 6 and 7 of the learning objectives are developed in carrying out the group work.

Theoretical-practical classes are given to support the development of group work, especially regarding the use of open-source libraries used in UC.

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

B. Kernighan, D. Ritchie, *The C Programming Language*, 2nd edition, Prentice Hall, 1988. ISBN 9780131103627

M. Barr, A. Massa, Programming Embedded Systems: With C and GNU Development Tools, 2nd Edition, Sebastopol, CA: O'Reilly Media, Inc., 2009. ISBN 9780596009830

W. Wolf, Computers as Components: Principles of Embedded Computer Systems Design, 3rd edition, San Francisco, CA: Morgan Kaufmann Publishers Inc., 2016. ISBN 9780123884367

R. Bryant, D. O'Hallaron, *Computer Systems: A Programmer's Perspective*, 3rd edition, Pearson, 2016. ISBN 9780134092669

S. Fuber, Arm System On-Chip-Architecture, 2nd edition, Boston, MA: Addison-Wesley Longman Publishing Co., Inc., 2000. ISBN 9780201675191

---

[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.