
1. Designação da unidade curricular

[4170] Técnicas de Desenvolvimento de Software / Software Development Techniques

2. Sigla da área científica em que se insere

IC

3. Duração

Unidade Curricular Semestral

4. Horas de trabalho

162h 00m

5. Horas de contacto

Total: 67h 30m das quais T: 45h 00m | TP: 6h 00m | P: 16h 30m

6. % Horas de contacto a distância

Sem horas de contacto à distância

7. ECTS

6

8. Docente responsável e respetiva carga letiva na Unidade Curricular

[889] Pedro Alexandre de Seia e Cunha Ribeiro Pereira | Horas Previstas: 135 horas

9. Outros docentes e respetivas cargas letivas na unidade curricular

[1369] Paulo Alexandre Leal Barros Pereira | Horas Previstas: 67.5 horas
[1601] Afonso dos Santos Remédios | Horas Previstas: 135 horas

10. Objetivos de aprendizagem e a sua compatibilidade com o método de ensino (conhecimentos, aptidões e competências a desenvolver pelos estudantes).

Os estudantes que terminam com sucesso esta unidade curricular serão capazes de:

1. Demonstrar conhecimento sobre os princípios aplicáveis no âmbito do desenvolvimento de software;
2. Identificar as propriedades pretendidas no software e demonstrar conhecimento acerca das técnicas a usar para as influenciar;
3. Analisar e justificar a aplicação de padrões de desenho em contextos selecionados;
4. Desenhar, implementar e testar aplicações interativas de média dimensão;
5. Produzir documentação técnica onde se justificam as decisões tomadas no software desenvolvido.

10. Intended Learning objectives and their compatibility with the teaching method (knowledge, skills and competences by the students).

Students who successfully complete this course unit should be able to:

1. Demonstrate know-how on software development principles;
2. Identify software properties and demonstrate know how on the techniques used to influence them;
3. Analyze and justify the application of design patterns in selected contexts;
4. Design, implement and test medium sized interactive applications;
5. Write technical documentation justifying decisions made in the context of software development.

11. Conteúdos programáticos

- I. Princípios elementares de type driven development: encapsulamento e especificação de contratos e invariantes.
- II. Princípios elementares de test driven development: especificação explícita das dependências e respetiva resolução através de injeção.
- III. Composição de funções: high order functions, currying e aplicação parcial.
- IV. Composição de tipos: agregação, herança e polimorfismo.
- V. Princípios gerais de desenho de software: DRY, Law of Demeter e SOLID.
- VI. Seleção de padrões de desenho de uso geral e usados no âmbito do desenvolvimento de aplicações interativas.

11. Syllabus

- I. Basic principles of type driven development: encapsulation and contract specification and its invariants;
- II. Basic principles of test-driven development: explicit specification of dependencies and their resolution through dependency injection;
- III. Function composition: high order functions: currying and partial application;
- IV. Type composition: aggregation, inheritance and polymorphism;
- V. General principles of software development: DRY, Law of Demeter and SOLID;
- VI. Selection of design patterns of general usage and used in the context of interactive applications.

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

Esta unidade curricular introduz os conceitos e o vocabulário fundamental dos paradigmas e dos princípios usados no desenvolvimento de software.
Os objetivos 1, 2 e 3 são alcançados através dos itens I até VI do conteúdo programático.
O desenvolvimento de programas durante as sessões práticas permite atingir os objetivos 4 e 5.

12. Evidence of the syllabus coherence with the curricular unit's intended learning outcomes

This curricular unit introduces the concepts and fundamental vocabulary of the paradigms and principles used in software development.
Objectives 1, 2 and 3 are achieved through items I to VI of the program content.
The development of programs during practical sessions allows achieving objectives 4 and 5.

13. Metodologias de ensino e de aprendizagem específicas da unidade curricular articuladas com o modelo pedagógico

Previstas 30 aulas durante o semestre (15 de 3 horas e 15 de 1,5 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 para desenvolver programas interativos.

13. Teaching and learning methodologies specific to the curricular unit articulated with the pedagogical model

Planned during the semester in 30 lectures (15 of 3 hours and 15 of 1.5 hours). The lectures are intended for presentation of topics and practical examples. The main topics are further explored through practical work to develop programs using graphical user interface.

14. Avaliação

A avaliação é distribuída com exame final. Os resultados da aprendizagem 1, 2 e 3 são avaliados individualmente através do exame final. Durante o acompanhamento dos trabalhos de grupo realizados nas aulas práticas são avaliados os resultados da aprendizagem 4 e 5. Os resultados da aprendizagem 4 e 5 também são avaliados na discussão final dos trabalhos de grupo pedagogicamente fundamentais. Nota final corresponde a 40% do exame final e 60% dos trabalhos (nota da discussão).

14. Assessment

The assessment is distributed with a final exam. Learning outcomes 1, 2 and 3 are individually assessed through written final exam. During the monitoring of group work performed in class practices are assessed learning outcomes 4 and 5. Learning outcomes 4 and 5 are also evaluated in the final discussion of work group. Final grade corresponds to 40% of the final exam and 60% of the practical works.

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

O conhecimento das técnicas de programação event-driven e dos princípios usados no desenvolvimento de software é obtido através de aulas interativas e respetivos elementos de apoio, e da realização de uma aplicação, com a participação dos alunos, que necessita de especializar comportamentos, reutilizar código e utilizar interface com o utilizador, textual e gráfica. A competência para desenvolver boas práticas de desenho e de escrita de código para aplicações é desenvolvida através da realização de trabalhos com supervisão e da sua avaliação crítica.

15. Evidence of the teaching methodologies coherence with the curricular unit's intended learning outcomes

Knowledge of event-driven programming techniques and the principles used in software development is obtained through interactive classes and respective support elements, and through the creation of an application, with the participation of students, which requires specializing behaviors, reusing code and use user interface, both textual and graphical.
The competence to develop good practices in designing and writing code for applications is developed through carrying out work with supervision and its critical evaluation.

16. Bibliografia de consulta/existência obrigatória

M. Moskala, Effective Kotlin: Best practices, Marcin Moskala, 2019. ISBN-13: 978-8395452833; ISBN-10: 8395452837
A. Soshin, Hands-on Design Patterns with Kotlin, Packt publishing, 2018. ISBN-13: 978-1788998017; ISBN-10: 1788998014

17. Observações

Unidade Curricular Obrigatória

Data de aprovação em CTC:

Data de aprovação em CP: