Ficha de Unidade Curricular A3ES
Modelação e Programação
Licenciatura em Engenharia Informática e Multimédia
2024-25

1. Caracterização da Unidade Curricular

1.1 Designação
[2840] Modelação e Programação / Modeling and Programming

1.2 Sigla da área científica em que se insere
INF

1.3 Duração
Unidade Curricular Semestral

1.4 Horas de trabalho
162h 00m

1.5 Horas de contacto
Total: 67h 30m das quais TP: 45h 00m | P: 22h 30m

1.6 ECTS
6

1.7 Observações
Unidade Curricular Obrigatória

2. Docente responsável [1552] Pedro Viçoso Fazenda

3. Docentes e respetivas cargas letivas na unidade curricular Não existem docentes definidos para esta unidade curricular

4. Objetivos de aprendizagem (conhecimentos, aptidões e competências a desenvolver pelos estudantes)

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

(1) Usar a Unified Modeling Language (UML) como ferramenta de modelação, nomeadamente o seu diagrama de classes.

(2) Aplicar a modelação e programação orientada a objetos para construir aplicações.

(3) Criticar a estrutura de aplicações modeladas com objetos.

(4) Aplicar a modelação e programação orientada a eventos para construir aplicações.

(5) Desenvolver aplicações gráficas, incluindo aplicações que utilizem recursos multimédia.

(6) Usar ferramentas para edição, execução e depuração de aplicações.

4. Intended learning outcomes (knowledge, skills and competences to be developed by the students)

When students complete this course, they will be able to:

(1) Use the Unified Modelling Language (UML) as a tool for modelling class diagrams.

(2) Creating object-oriented applications.

(3) Model applications using the object-oriented programming paradigm.

(4) Create event-driven programming applications.

(5) Create graphical user interface applications, including applications that use multimedia resources.

(6) Use and integrated development environment to edit, execute and debug applications.

5. Conteúdos programáticos

I. Introdução ao Java.

II. Noções de classe, instância, atributo e método.

III. Noções de herança e conceitos associados: derivação, polimorfismo, ligação dinâmica, classes abstratas e interfaces.

IV. Modelação de cenários utilizando o diagrama de classes do UML.

V. Exceções, sua definição, lançamento e tratamento.

VI. Entrada e saída de dados.

VII. Estruturas de dados dinâmicas (com especial enfoque nas listas e iteradores) e classes genéricas.

VIII. Conceitos de programação orientada a eventos: evento, receptor de evento e envio e recepção de eventos.

IX. Desenvolvimento de aplicações baseadas em eventos.

X. Apresentação da interface gráfica Swing e seus principais componentes. O padrão de desenho Model View Controller (MVC).

XI. Conceitos de programação com recursos multimédia.

XII. Desenvolvimento de aplicações Swing com recursos multimédia.

XIII. Padrões de desenho

5. Syllabus

I. Introduction to the Java programming language.

II. Notion of class, instance, method and attribute.

III. Notion of inheritance and behavioural subtyping, polymorphism, dynamic binding, abstract classes and interfaces.

IV. Modelling applications using UML class diagrams.

V. Exceptions and error handling.

VI. Data input and output.

VII. Dynamic Data Structures (Lists, Arrays, Sets, Queues, Hash-based structures, etc.) and generic classes.

VIII. Concept of event-driven programming: event, handler/listener and event dispatching.

IX. Development of event-driven applications.

X. Java Swing widget toolkit graphical user interface. The Model View Controller (MVC) design pattern. XI.

XI. Creating interactive programs with multimedia resources.

XII. Development of Swing applications with multimedia resources.

XIII. Design Patterns.

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

Esta unidade curricular lida com os conceitos e práticas da modelação e concepção de aplicações, nomeadamente o paradigma da programação orientada por objetos, o paradigma da programação orientada por eventos e a programação com recursos multimédia. Estes conceitos são concretizados na linguagem Java nomeadamente na sua componente estrutural, explorando o paradigma dos objetos, e na sua componente dinâmica, explorando os modelos de execução sequencial e de execução por eventos.

6. Evidence of the syllabus coherence with the curricular unit´s intended learning outcomes

This course mainly focuses on practical concepts concerning applications modelling and programming, particularly focused on the object-oriented programming paradigm, event-driven applications and programming with graphical user interfaces and multimedia resources. These concepts are tested and carried out using the Java programming language for learning structure, taking advantage of its object-oriented paradigm, and its dynamic behaviour, exploring the sequential and event-driven execution models.

7. Metodologias de ensino (avaliação incluída)

A metodologia desenvolve-se em duas componentes de ensino:
TP: Exposição teórica, onde por cada tema teórico são apresentados exemplos e resolvidos exercícios;
P: Prática laboratorial onde são realizados trabalhos práticos.
A avaliação é distribuída sem exame final, composta pelas seguintes componentes pedagogicamente fundamentais: TP, com 3 trabalhos práticos (TP1, TP2, TP3) realizados em grupo; e TF com um trabalho final individual, ambas com nota mínima de 9,50. A nota final é NF = 0,5xTP+0,5xTF.
A nota de TP é dada por TP=0,4xTP1+0,2xTP2+0,4xTP3, onde cada trabalho prático tem nota mínima de 8,00.
A componente TF é composta por duas partes:
Parte_A: planeamento e desenho de uma aplicação;
Parte_B: codificação e apresentação da aplicação.
A nota TF é obtida por TF=0,4xParte_A+0,6xParte_B, onde cada parte tem nota mínima de 8,00.

7. Teaching methodologies (including assessment)

The methodology is based on two teaching components:
-TP: Theoretical exposition, where for each theoretical topic examples are presented and exercises solved;
-P: Laboratory practice where practical work is carried out.
The assessment is distributed without a final exam, and consists of the following pedagogically fundamental components: TP, with 3 practical assignments (TP1, TP2, TP3) carried out in groups; and TF with a final individual assignment, both with a minimum garde of 9.50. The final grade is obtained by NF = 0.5xTP+0.5xTF.
The TP grade is given by TP=0.4xTP1+0.2xTP2+0.4xTP3, where each practical assignment has a minimum grade of 8.00.
The TF component is made up of two parts:
Part_A: planning and designing an application;
Part_B: coding and presentation of the application.
The TF garde is obtained by TF=0.4xPart_A+0.6xPart_B, where each part has a minimum mark of 8.00.

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

Nas aulas é dado o programa correspondente aos objetivos de aprendizagem (1) a (6). São apresentados exemplos e resolvidos exercícios. Nas aulas laboratoriais pretende-se que os estudantes antecipem soluções, para isso, é fornecido antecipadamente um guia laboratorial para cada um dos trabalhos. Tendo em conta o cumprimento dos objectivos da disciplina, os trabalhos laboratoriais focam-se essencialmente em: a) Introdução à programação Java e estruturas de dados; b) os tópicos abordados em (3), divididos em objectos e instâncias e herança e polimorfismo; c) os tópicos abordados em (5) e (6), com uma aplicação final (normalmente um jogo) com várias alíneas obrigatórias e outras opcionais. Tendo em consideração o cumprimento dos objetivos (3), (5) e (6), as soluções propostas pelos estudantes são discutidas no âmbito da turma. Na discussão final é avaliado o trabalho, realizado autonomamente per cada aluno, com particular destaque para os relatórios, aproveitando a oportunidade para salientar aspectos manifestados nos objetivos de aprendizagem (3), (5), e (6) que sejam considerados oportunamente relevantes.

8. Evidence of the teaching methodologies coherence with the curricular unit´s intended learning outcomes

In class the syllabus of the course is taught to fulfil the learning objectives stated in (1) to (5). Examples are presented in class with hands-on exercises. In the lab, students are expected to anticipate solutions by being received, in anticipation, a lab guide for each practical project that they must fully complete at home and in the following TP classes. Attending to the objectives of the course, these practical projects are essentially divided according to the following items:

a) Introduction to Java programming and data structures;

b) The topics referred in (3), divided into object-oriented programming and inheritance, polymorphism, abstract classes and interfaces;

c) The topics referred in (5) and (6), with the development of a fully functional final application (usually a game) with several mandatory items and a few optional items.

Considering the execution of the objectives stated in (3), (5) and (6), all solutions proposed by the students are discussed in class. In the final discussion of each project each individual team-project is evaluated (with a detailed review of the documentation and reporting). The outcome of this evaluation is highly dependent on what each student learned in (3), (5) and (6).

9. Bibliografia de consulta/existência obrigatória

Savitch W, Java: An Introduction to Problem Solving and Programming (8th edition), Pearson, 2017.

10. Data de aprovação em CTC 2024-07-17

11. Data de aprovação em CP 2024-06-26