

---

**Ano Letivo** 2020-21

---

**Unidade Curricular** PROGRAMAÇÃO ORIENTADA POR OBJETOS

---

**Cursos** ENGENHARIA INFORMÁTICA (1.º ciclo)  
ENGENHARIA ELETRÓNICA E TELECOMUNICAÇÕES (Mestrado Integrado)

---

**Unidade Orgânica** Faculdade de Ciências e Tecnologia

---

**Código da Unidade Curricular** 140064323

---

**Área Científica** CIÊNCIA DE COMPUTADORES

---

**Sigla**

---

**Línguas de Aprendizagem** Português-PT

---

**Modalidade de ensino** Presencial diurno

---

**Docente Responsável** Helder Aniceto Amadeu de Sousa Daniel

| DOCENTE                               | TIPO DE AULA | TURMAS            | TOTAL HORAS DE CONTACTO (*) |
|---------------------------------------|--------------|-------------------|-----------------------------|
| Helder Aniceto Amadeu de Sousa Daniel | PL; T        | T1; PL1; PL2; PL3 | 28T; 84PL                   |

\* 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

Programação Imperativa

Laboratório de Programação

Algoritmos e Estruturas de Dados

### Objetivos de aprendizagem (conhecimentos, aptidões e competências)

No fim desta disciplina os alunos deverão ser capazes de entender e aplicar os princípios e as técnicas de programação orientada por objectos. Em particular, deverão ser capazes de gerar uma especificação UML recorrendo a padrões de desenho e implementar essa especificação na linguagem JAVA.

### Conteúdos programáticos

1. Princípios e conceitos fundamentais da programação orientada por objetos.
2. Introdução à modelação orientada por objetos e à UML- Unified Modeling Language.
3. Projeto de classes
4. Herança, polimorfismo e binding
5. Interfaces
6. Gestão de erros e exceções
7. Classes aninhadas
8. Padrões de projeto fundamentais: Template Method, Strategy, Iterator, Composite
9. Classes e métodos genéricos
10. Java Collection Framework
11. Streams
12. Internet networking (java.net)

### **Metodologias de ensino (avaliação incluída)**

Método clássico de ensino e aprendizagem para as disciplinas científico-tecnológicas.

Aulas teóricas expositivas com recurso ao quadro e projector de vídeo.

Aulas práticas com tutoriais, problemas e mini-projectos de programação, incluindo actividades que vão desde a modelação, implementação e depuração.

Avaliação

Nota final = 0,6 Exame + 0,4 Avaliação Prática Arredondamentos só na nota final.

### **EXAMES**

Os exames consistem numa prova escrita com consulta de até 25 páginas A4 numa fonte de tamanho não inferior a 9. São admitidos a exame, independentemente da época, os estudantes cuja nota de avaliação prática seja  $\geq 7,5$  valores.

### **AVALIAÇÃO PRÁTICA**

Média ponderada de trabalhos práticos.

Os trabalhos são realizados em grupo, de inscrição obrigatória. A nota prática do grupo é convertida numa nota prática individual no momento da discussão dos trabalhos. A nota prática é, portanto, individual, estando dependente do desempenho de cada elemento do grupo.

---

### **Bibliografia principal**

[Referência principal]

Horstmann, Big Java: Early Objects 7th ed, Wiley, December 2018. ISBN: 978-1-119-49909-1

[Referência complementar sobre padrões de projeto]

Erich Gamma, Richard Helm, Ralph Johnson, John Vlissides. Design Pattern Elements of Reusable Object-Oriented Software, Addison-Wesley, 1995

[Referência complementar sobre UML]

UML Quick Ref: <http://www.uml.org>

**Academic Year** 2020-21

**Course unit** OBJECT-ORIENTED PROGRAMMING

**Courses** INFORMATICS (COMPUTER SCIENCE) (1st Cycle)  
ELECTRONIC ENGINEERING AND TELECOMMUNICATIONS (Integrated Master's)

**Faculty / School** FACULTY OF SCIENCES AND TECHNOLOGY

**Main Scientific Area**

**Acronym**

**Language of instruction** Português-PT

**Teaching/Learning modality** Daytime presential

**Coordinating teacher** Helder Aniceto Amadeu de Sousa Daniel

| Teaching staff                        | Type  | Classes           | Hours (*) |
|---------------------------------------|-------|-------------------|-----------|
| Helder Aniceto Amadeu de Sousa Daniel | PL; T | T1; PL1; PL2; PL3 | 28T; 84PL |

\* 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

Imperative Programming

Programming Lab

Algorithms and Data Structures

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

1. Enumerate, describe, and justify the object-oriented concepts, principles, and techniques.
2. Application modeling using UML, emphasizing on architectural aspects (diagrams of classes and objects)
3. To Select and employ fundamental design patterns.
4. To use JAVA as a object-oriented programming language

### Syllabus

1. Object-Oriented concepts and principles;
2. Introduction to Object Oriented Modelling and UML (Unified Modelling Language);
3. Designing classes
4. Inheritance, polymorphims, and binding
5. Interfaces
6. Exceptions and Error Handling
7. Nested classes
8. Fundamental Design Patterns : Template Method, Strategy, Iterator, Composite
9. Generics
10. Java Collection Framework
11. Stream processing
12. Internet networking (java.net)

### Teaching methodologies (including evaluation)

Classic method of teaching and learning for scientific-technological subjects.

Lectures using the board and video projector.

Hands-on lab classes with tutorials, problems, and programming mini-projects, including activities ranging from modeling, implementation, and debugging.

Evaluation

Final grade = 0,6 Exam + 0,4 Practical Assessment; Round off in the final grade only.

### EXAMS

The exams consist of a written test with consultation of up to 25 A4 pages in a font size of not less than 9. Students, whose practical grade is  $\geq 7.5$ , are admitted to the exam.

### PRACTICAL ASSESSMENT

Weighted average of practical work.

The works are performed in groups with mandatory registration. The group practical note is converted into an individual practical note at the time of discussion. The practical grade is therefore individual, depending on the performance of each group member.

---

### Main Bibliography

[Main reference]

Horstmann, Big Java: Early Objects 7th ed, Wiley, December 2018. ISBN: 978-1-119-49909-1

[Other optional references]

Erich Gamma, Richard Helm, Ralph Johnson, John Vlissides. Design Pattern Elements of Reusable Object-Oriented Software, Addison-Wesley, 1995

UML Quick Ref: <http://www.uml.org>