UAlg
UNIVERSIDADE DO ALGARVE

English version at the end of this document

Ano Letivo	2016-17
Unidade Curricular	TÓPICOS AVANÇADOS EM ENGENHARIA DE SOFTWARE
Cursos	ENGENHARIA INFORMÁTICA (2.º ciclo)
Unidade Orgânica	Faculdade de Ciências e Tecnologia
Código da Unidade Curricular	14741036
Área Científica	CIÊNCIA DE COMPUTADORES
Sigla	
Línguas de Aprendizagem	Inglês
Modalidade de ensino	Presencial
Docente Responsável	Pedro João Valente Dias Guerreiro

DOCENTE	TIPO DE AULA	TURMAS	TOTAL HORAS DE CONTACTO (*)
Pedro João Valente Dias Guerreiro	PL; T	T1; PL1	30T; 30PL

* 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
1º	S1	30T; 30PL	168	6

* A-Anual;S-Semestral;Q-Quadrimestral;T-Trimestral

Precedências

Sem precedências

Conhecimentos Prévios recomendados

Licenciatura em engenharia informática ou equivalente.

Objetivos de aprendizagem (conhecimentos, aptidões e competências)

- Compreender os princípios da programação funcional.
- Escrever programas funcionais concisos e elegantes, tirando partido da recursividade, do encontro de padrões e das funções de ordem superior.
- Articular a programação funcional com a programação orientada pelos objetos
- Desenhar, implementar, utilizar, conhecer as vantagens da estruturas de dados imutáveis.
- Analisar em abstrato as propriedades das funções.
- Compreender esclarecidamente a utilização dos tipos genéricos na programação funcional.
- Compreender e saber utilizar as técnicas que combinam a programação funcional com a noção de estado e os efeitos laterais.

(Estes objetivos de aprendizagem são fortemente inspirados nos do curso Coursera, Functional Programming in Scala Specialization, https://www.coursera.org/specializations/scala)

Conteúdos programáticos

1. Funções em Scala
2. Avaliação de funções
3. Funções de ordem superior
4. Classes, herança e polimorfismo
5. Encontro de padrões
6. Tipos genéricos
7. Funções enquanto objetos
8. Listas
9. Pares e tuplos
10. Outras coleções
11. Expressões for
12. Mapas
13. Expressões for avançadas
14. Mónadas
15. Indução estrutural
16. Correntes (*streams*)
17. Avaliação protelada
18. Sequências infinitas
19. Efeitos laterais em Scala

Metodologias de ensino (avaliação incluída)

Nas aulas teóricas, o professor discute os temas da cadeira, usando o seu computador para exibir os transparentes, para fazer demonstrações e para ilustrar o desenvolvimento de programas.

Dado que a cadeira tem poucos alunos, as aulas teóricas serão interativas, e os alunos deverão repetem imediatamente no seu computador os exemplos que o professor apresenta.

Nas aulas práticas, os alunos resolvem pequenos problemas de programação ou realizam trabalhos mais longos, com guião, no computador, supervisionados pelo professor.

Os alunos completarão a sua formação através de trabalho individual ou em grupo, realizado fora das aulas.

A avaliação usa a modalidade de ?avaliação por frequência?, nos termos da alínea b) do número 1 do artigo 9.º do Regulamento de Avaliação da Universidade do Algarve , de 31 de agosto de 2016. O exame assume a forma de uma prova escrita. A avaliação por frequência pode dispensar o exame.

Bibliografia principal

1. Programming in Scala. Martin Odersky, Lex Spoon and Bill Venners. 3rd edition. Artima. https://www.amazon.co.uk/Programming-Scala-Comprehensive-Step---Step-ebook/dp/B01EX49FOU
2. Coursera, Functional Programming in Scala Specialization, https://www.coursera.org/specializations/scala
3. Documentação da API do Scala, http://www.scala-lang.org/api/current/
4. Scala for the Impatient. Cay Horstmann, primeira parte, https://www.lightbend.com/resources/e-book/scala-for-the-impatient
5. Coleção de acetados das aulas.

UAlg
UNIVERSIDADE DO ALGARVE

Academic Year	2016-17
Course unit	ADVANCED TOPICS IN SOFTWARE ENGINEERING
Courses	INFORMATICS ENGINEERING
Faculty / School	Faculdade de Ciências e Tecnologia
Main Scientific Area	CIÊNCIA DE COMPUTADORES
Acronym	
Language of instruction	English
Teaching/Learning modality	Lectures and labs, for small groups.
Coordinating teacher	Pedro João Valente Dias Guerreiro

Teaching staff	Type	Classes	Hours (*)
Pedro João Valente Dias Guerreiro	PL; T	T1; PL1	30T; 30PL

* 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

Students should have a level of scientific and technical knowledge corresponding to a first cycle degree in Computer Science.

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

- Understand the principles of functional programming.
- Write concise and elegant functional programs, taking advantage of recursion, pattern matching and higher-order functions.
- Merge functional programming with object-oriented programming by objects
- Design, implement, use, know the benefits of immutable data structures.
- Analyze in the abstract properties of functions.
- Fully understand the use of generic types in functional programming.
- Use the techniques that combine functional programming with the concept of state and side effects.

(These learning objectives are strongly inspired by the Coursera course, Functional Programming in Scala Specialization, https://www.coursera.org/specializations/scala)

Syllabus

1. Functions in Scala
2. Evaluation of functions
3. Higher order functions
4. Classes, inheritance and polymorphism
5. Pattern matching
6. Generic types
7. Functions as objects
8. Lists
9. Pairs and tuples
10. Other collections
11. For expressions
12. Maps
13. Advances for expressions
14. Monads
15. Structural inductions
16. Streams
17. Lazy evaluation
18. Infinite sequences
19. Side effects in Scala

Teaching methodologies (including evaluation)

In the lectures, the teacher discusses the topic of the course, using his computer to present the course slides, to make experiments and demonstrations, and to illustrate the development of programs.

Given that this course has a small number of students, the lectures are interactive, and in many cases the students will repeat immediately in their computer the study cases the professor is developing.

In the labs, students solve small problems programming or perform longer programming assignments.

Students complete their training through individual or group work, done outside the classroom.

The evaluation uses the modality of "evaluation by frequency", as prescribed in the general regulations of the university. The examination takes the form of a written test. The exam may not be required if the grade in the evaluation by frequency is satisfactory.

Main Bibliography

1. Programming in Scala. Martin Odersky, Lex Spoon and Bill Venners. 3rd edition. Artima. https://www.amazon.co.uk/Programming-Scala-Comprehensive-Step---Step-ebook/dp/B01EX49FOU
2. Coursera, Functional Programming in Scala Specialization, https://www.coursera.org/specializations/scala
3. Scala API reference, http://www.scala-lang.org/api/current/
4. Scala for the Impatient. Cay Horstmann, primeira parte, https://www.lightbend.com/resources/e-book/scala-for-the-impatient
5. Lecture slides, provided by the course staff.