1

PROGRAMMING LANGUAGES, TECHNOLOGIES AND PARADIGMS

Introduction to the course

2018-2019

Why should we learn about programming languages and paradigms?

- Most programming languages share similar notions and concepts, often expressed in different ways:
  - Conditional sentences
  - Iteration
  - Data structures…
- Some features identify families of languages
  - concurrency
  - inheritance…

Why should we learn about programming languages and paradigms?

- To ease learning new languages.
- To simulate some features in languages which lack them.
- To improve the ability to develop efficient algorithms.
- To improve the use of the available programming language.
- To increase the knowledge of the programmer.
- To ease the design of new languages.

Why should we learn about programming languages and paradigms?

To understand the design and implementation of languages.

- syntax – program construction rules
- semantics – meaning of programs
- implementation – how to execute them
- pragmatics – practical aspects of their use

To choose the appropriate language for a given application.

Imperative: Cobol/Fortran	Management/scientific apps.
OO: C++/Ada/Java	System programming
Functional: Haskell/LISP/ML	Symbolic computation
Logic: Prolog/Mercury …	Decision making

Main goal of the course

- Introducing the main concepts that can be found in most programming **languages**
- Discovering the main features and applications of the most prominent programming **paradigms**
- Learning about basic support **technologies** for these languages and paradigms

We aim at learning the general principles of programming languages. We illustrate them by using existing programming languages

Learning outcomes

At the end of the course, you will be able to:

- Distinguish the features that are specific in the main programming paradigms
- Solve a given problem by using different programming styles and write equivalent programs in different languages
- Choose the appropriate language for a given application and use the most efficient techniques to write programs in such a language

Connection with other courses

- (2°A) TAL: Syntactic definition of programming language
- (2°B) EDA: Programming techniques
- (2°B) Concurrencia y Sistemas Distribuidos: Concurrent programming
- (3°A) Ingeniería del Software: OO programming
- **Rama de *Computación***

  (4°A) Lenguajes de Programación y Procesadores de Lenguajes: Computation models, Theory of programming languages, Processing tools
- **Rama de *Ingeniería de Computadores***

  (4°A) Lenguajes y Entornos de Programación Paralela
- **Rama de *Ingeniería del Software***
  - (3°B) Métodos formales industriales: Certification and verification techniques and tools.
  - (4°A) Análisis, depuración y validación de software: Analysis, debugging and validation techniques and tools.

LTP: Contents

1: Introduction (2.5 weeks)

- Motivation.
- Basic concepts (types, polymorphism, reflection, …).
- Main programming paradigms: imperative, functional, logic, OO, concurrent,
- Further paradigms. Interaction-based and emerging paradigms.

2: Foundations of programming languages (3 weeks)

- Syntax and static semantis of programming languages.
- Dynamic semantics of programming languages. Semantic definition styles. Operational semantics. Axiomatic semantics.
- Semantic properties: correctness, completeness, equivalence. Specification vs programming.
- Implementation of programming languages: virtual machines and intermediate languages.

LTP: Contents

3: Functional programming (4.5 weeks)

- Brief introduction to the functional notation
- Types in Functional Programming:
  - Types and type inference
  - Type systems (predefined, functional, algebraic, parametric).
- Polymorphism: genericity, coercion and overloading.
- Computational model (reduction and evaluation).
- Advanced features:
  - Anonymous functions and function composition.
  - Iterators and compressors (foldr).

4: Logic paradigm (2 weeks)

- Logic paradigm: Logic variables and unification.
- Computational model (SLD resolution)

5: Support technologies and tools (2 weeks)

- Program debugging and validation.

LTP: Schedule (Theory)

THEORY

2018

L	M	Mi	J	V	S	D
SEPTEMBER						
					1	2
3	4	5	6	7	8	9
10	11	12	13	14	15	16
17	18	19	20	21	22	23
24	25	26	27	28	29	30

L	M	Mi	J	V	S	D
OCTOBER						
1	2	3	4	5	6	7
8M	9	10	11	12	13	14
15	16	17	18	19	20	21
22	23	24	25	26	27	28
29	30	31				

L	M	Mi	J	V	S	D
NOVEMBER						
			1	2	3	4
5	6	7	8	9	10	11
12	13	14	15	16	17	18
19	20	21	22	23	24	25
26	27	28	29	30		

L	M	Mi	J	V	S	D
DECEMBER					1	2
3	4	5J	6	7	8	9
10	11	12	13	14	15	16
17	18	19	20	21	22	23
24	25	26	27	28	29	30
31						

2019

L	M	Mi	J	V	S	D
JANUARY						
	1	2	3	4	5	6
7	8	9	10	11	12	13
14	15	16	17	18	19	20
21	22	23	24	25	26	27
28	29	30	31			

L	M	Mi	J	V	S	D
FEBRUARY						
				1	2	3
4	5	6	7	8	9	10
11	12	13	14	15	16	17
18	19	20	21	22	23	24
25	26	27	28			

L	M	Mi	J	V	S	D
MARCH						
				1	2	3
4	5	6	7	8	9	10
11	12	13	14	15	16	17
18	19	20	21	22	23	24
25	26	27	28	29	30	31

L	M	Mi	J	V	S	D
APRIL						
1	2	3	4	5	6	7
8	9	10	11	12	13	14
15	16	17	18	19	20	21
22	23	24	25	26	27	28
29	30					

	LTP Exam		Exams Period		Holidays		Invited Conference

	Theme 1
	Theme 2
	Theme 3
	Theme 4
	Theme 5

LTP: Schedule (Laboratory)

LABORATORY

2018

L	M	Mi	J	V	S	D
SEPTEMBER						
					1	2
3	4	5	6	7	8	9
10	11	12	13	14	15	16
17	18	19	20	21	22	23
24	25	26	27	28	29	30

L	M	Mi	J	V	S	D
OCTOBER						
1	2	3	4	5	6	7
8M	9	10	11	12	13	14
15	16	17	18	19	20	21
22	23	24	25	26	27	28
29	30	31				

L	M	Mi	J	V	S	D
NOVEMBER						
			1	2	3	4
5	6	7	8	9	10	11
12	13	14	15	16	17	18
19	20	21	22	23	24	25
26	27	28	29	30		

L	M	Mi	J	V	S	D
DECEMBER					1	2
3	4	5J	6	7	8	9
10	11	12	13	14	15	16
17	18	19	20	21	22	23
24	25	26	27	28	29	30
31						

2019

L	M	Mi	J	V	S	D
JANUARY						
	1	2	3	4	5	6
7	8	9	10	11	12	13
14	15	16	17	18	19	20
21	22	23	24	25	26	27
28	29	30	31			

L	M	Mi	J	V	S	D
FEBRUARY						
				1	2	3
4	5	6	7	8	9	10
11	12	13	14	15	16	17
18	19	20	21	22	23	24
25	26	27	28			

L	M	Mi	J	V	S	D
MARCH						
				1	2	3
4	5	6	7	8	9	10
11	12	13	14	15	16	17
18	19	20	21	22	23	24
25	26	27	28	29	30	31

L	M	Mi	J	V	S	D
APRIL						
1	2	3	4	5	6	7
8	9	10	11	12	13	14
15	16	17	18	19	20	21
22	23	24	25	26	27	28
29	30					

	LTP Exam		Exams Period		Holidays		Invited Conference

	Java
	Haskell
	Prolog

LTP: Evaluation

- **Classroom/homework activities (1 point)**: exercises, last-minute questions, attendance to invited talks, online tests, etc.
- **Theory (6 points)**: There will be two **partial exams:**
  - **November 9** (Themes 1 and 2, 40% of the theory score)
  - **January 14** (Themes 3, 4 and 5, 60% of the theory score)

  There will be a resit for all parts (**January 25**). A score of at least 4 (over 10) is required in each partial exam to compute the average with other scores.
- **Laboratory (3 points)**: Attendace to (at least) 80% of sessions is mandatory. A score of at least 4 (over 10) is required to compute the average with other scores. There will be two partial exams at the laboratory:
  - Part I (1 point). November 9 (together with the theory exam). Topic: Java (practices 1 and 2)
  - Parts II and III (2 points). January 14 (together with the theory exam). Topics: Haskell and Prolog (Practices 3 to 6)

  For students attending at least 80% of lab sessions, but not reaching the minimum score of 4 in the laboratory part, there will be a resit (of both parts I and II) by January 25

Working material

- The teaching material (slides, exercises, unattended activities, etc.) will be available on the web (Poliformat/Recursos).
- Bibliographic references are available at the School Library and/or at the University Library.