COURSE DESCRIPTION

1. Program information

1.1 University	**“Alexandru Ioan Cuza”, University of Iaşi**
1.2 Facultty	**Faculty of Computer Science**
1.3 Department	**Department of Computer Science**
1.4 Study domain	**Computer Science**
1.5 Study cycle	**Bachelor**
1.6 Study program / Qualification	Computer Science

2. Course Information

2.1 Discipline name			**Principles of Programming Languages**				
2.2 Course teacher			**Arusoaie Andrei**				
2.3 Seminar teacher			**Arusoaie Andrei**				
2.4 Yearof study	**2**	**2.5** Semester	**1**	**2.6** Evaluation type	**EVP**	**2.7** Discipline status*	**OP**

** OB – Obligatoriu / OP – Opţional*

3. Total estimated hours (hours per semester and didactic activities)

3.1 Hours per week	**4**	In which: **3.2** course	**2**	**3.3** seminar/laboratory	**2**
3.4 Hours in curriculum	**56**	In which: **3.5** course	**28**	**3.6** seminar/laboratory	**28**

Time distribution	hours
Manual study, course support, bibliography, and others	**56**
Suplementary documentation in the library, electronic forums, and on the field	**60**
Seminaries/laboratories preparation, homeworks, reports, portfolios, and essays	**0**
Tutoring	**0**
Evaluation	**4**
Other activities	**0**

3.7 Total hours per individual	**64**
3.8 Total hours per semester	**120**
3.9 Credits	**4**

4. Preconditions (in necessary)

4.1 Curriculum	**Logics for Computer Science, Object-Oriented Programming**
4.2 Competencies	**Imperative programming**

5. Conditions (if necessary)

5.1 For course operation	Lecture room
5.2 For seminary/laboratory operation	**Laboratory room**

6. Specific skills acquired

Professional skills	**C1.** Defining syntax of programming languages **C2.** Defining semantics of programming languages **C3.** Using a framework for defining programming languages
Transversal skills	**CT1.** Capacity to design a new programming language **CT2.** Capacity to create an interpreter for a programming language **CT3.** Capacity to learn fast a new programming language

7. Discipline objectives (din grila competenţelor specifice acumulate)

7.1 General objective	Presenting in a simple and correct manner the main concepts of programming languages and a formal framework that allows defining programming languages and running programs written in that languages.
7.2 Specific objectives	After taking this course, students will be able to : ▪ Explain concepts specific to various programming language paradigms ▪ Describe the syntax and the semantics of a programming language ▪ Use a framework for defining a programming language ▪ Design a programming language

8. Contents

8.1	**Lecture**	**Teaching methods**	**Observations** (hours and bibliography)
1.	Introduction to Programming languages. The main programming paradigms.	Slides, blackboard	2 hours, Chapters 1 and 13 in [1].
2.	Defining programming languages: syntax	Slides, blackboard	2 hours, Chapter 2 in [1].
3.	Defining programming languages: semantics	Slides, blackboard	2 hours, Chapter 6 in [1].
4.	Side-effects. I/O. Local variables.	Slides, blackboard	2 hours, [3].
5.	Non-determinism. Threads.	Slides, blackboard	2 hours, Chapter 8 in [1].
6.	Functions. Call-by-value. Call-by-reference. Name and the environment.	Slides, blackboard	2 hours, Chapter 4 in [1]

7.	Case studies: SIMPLE	Slides, blackboard	2 hours, [5]
8.	Tools for defining programming languages	Slides, blackboard	2 hours, one of these frameworks will be discussed: Coq, Maude, Isabelle, Racket, Rascal, Spoofax
9.	Operational Semantics: Small Step	Slides, blackboard	2 hours, [2]
10.	Operational Semantics: Big Step	Slides, blackboard	2 hours, [2]
11.	Paradigms: object-oriented programming	Slides, blackboard	2 hours, Chapter 10 in [1]
12.	Paradigms: functional programming	Slides, blackboard	2 hours, Chapter 11 in [1]
13.	Paradigms: logical programming	Slides, blackboard	2 hours, Chapter 12 in [1]
14.	Other types of semantics: axiomatic semantics	Slides, blackboard	2 hours

Bibliography

Main references:

[1] **Maurizio Gabbrielli and Simone Martini Programming Languages: Principles and Paradigms. Springer-Verlag London Limited 2010**

[2]. Programming Languages and Operational Semantics, Fernández, Maribel, 2014

[3]. K: A Semantic Framework for Programming Languages and Formal Analysis Tools, Grigore Roșu, Marktoberdorf'16, NATO Science for Peace and Security

[4]. K framework: kframework.org

[5]. K Overview and SIMPLE Case Study, Grigore Rosu si Traian Florin Serbanuta, K'11, ENTCS.

[6]. Pagina cursului: https://profs.info.uaic.ro/~arusoaie.andrei/lectures/PLP/plp.html

8.2	**Seminar / Laboratory**	**Teaching methods**	**Observations** (hours and bibliography)
1.	Installation of the K framework. Compilarea definitiilor de limbaj: kompile. Main working setup.	Free discussions.	2 hours, kframework.org
2.	Defining the syntax of IMP: kast.	Resolving the exercise sheet.	2 hours
3.	Defining the semantics of IMP: krun.	Resolving the exercise sheet.	2 hours
4.	Side-effects, I/O, locals in IMP.	Resolving the exercise sheet.	2 hours
5.	Exploring non-determinism. Threads in IMP.	Resolving the exercise sheet.	2 hours
6.	Implementing functions in IMP.	Resolving the exercise sheet.	2 hours

7.	Midterm	Midterm test	2 hours; live test in the lab.
8.	Other tools for defining languages. Experiments with these tools.	Demo and examples.	2 hours
9.	Presentation of the project proposals.	The teacher presents all the project proposals. Students have to decide the project they will develop until the end of the semester.	2 hours
10.	Project: Requirements and Diagrams.	Direct discussions.	2 hours
11.	Project: midterm presentation.	Evaluation. Direct discussions.	2 hours
12.	Project: arhitecture and implementation.	Direct discussions.	2 hours
13.	Project: testing	Direct discussions.	2 hours
14.	Project: final presentation.	Evaluation. Direct discussions.	2 hours

Bibliography

[1]. K framework: kframework.org
[2]. Lectures page: https://profs.info.uaic.ro/~arusoaie.andrei/lectures/PLP/plp.html

9. Course content synchronization with the expectations of the community representatives, professional associations and employers from the program domain

This discipline aims to develop the ability to learn quickly a new programming language, and to develop the necessary skills to design and implement a new programming language. Students will be able to easily adapt to whatever (domain specific) programming languages that software companies are using.

10. Evaluation

Activity type	**10.1 Evaluation criteria**	**10.2 Evaluation methods**	**10.3 The weigth of each evaluation form (%)**
10.4 Course	50 points	Test	50%
10.5 Seminar/ Laboratory	20 + 30 points	Project	50%
10.6 Standard minim de performanţă : 50 points			
Project points can be obtained in two stages: preliminary presentation (=20 points) and final presentation (=30 points). The points obtained at the preliminary presentation are valid only if the final presentation has been done.			

Data completării, Titular de curs, Titular de seminar,

Data avizării în departament, Director de departament,