Universitatea de Vest din Timişoara

SYLLABUS / FIŞA DISCIPLINEI

1. Information on the study programme / Date despre programul de studii

1.1. Institution / Instituția de învățământ superior	Universitatea de Vest din Timişoara
1.2. Faculty / Facultatea	Matematică și Informatică
1.3. Department / Departamentul	Computer Science (Informatică)
1.4. Study program field	Computer Science (Informatică)
1.5. Study cycle/ Ciclul de studii	Bachelor / licență
1.6. Study programme / Programul de studii / calificarea*	Computer Science / Informatică în limba engleză / Database administration / *Administrator baze de date - 252101; Computer network administration / Administrator de retea de calculatoare - 252301; Analyst / Analist - 251201; Research assistant in computer science / Asistent de cercetare în informatica - 214918; Teacher in secondary schools / Profesor în învatamântul gimnazial - 233002; Programmer / Programator - 251202; Software systems designers / Proiectant sisteme informatice - 251101*

2. Information on the course / Date despre disciplină

2.1. Title of the course / Denumirea disciplinei	Functional Programming						
2.2. Teacher in charge of the course / Titularul activităților de curs	Mircea Marin						
2.3. Teacher in charge of the seminar / Titularul activităților de seminar							
2.4. Study year / Anul de studii	2	2.5. Semester / Semestrul	2	2.6. Examination type / Tipul de evaluare: E(xam)/C(olloquim)	C	2.7. Course type / Regimul disciplinei: M(andatory)/ E(lective)/ F(acultative)	DO

3. Estimated study time (number of hours per semester) /Timpul total estimat (ore pe semestru al activităților didactice)

3.1. Attendance hours per week / Număr de ore pe săptămână	3	out of which din care: 3.2 lecture/ curs	2	3.3. seminar/laborator	1
3.4. Attendance hours per semester / Total ore din planul de învățământ	42	out of which: 3.5 lecture / curs	28	3.6. seminar/laborator	14

Distribution of the allocated amount of time / Distribuția fondului de timp*	**hours/ ore**
Individual study /Studiu după manual, suport de curs, bibliografie și notițe	28
Supplementary documentation at library or using electronic repositories / Documentare suplimentară în bibliotecă, pe platformele electronice de specialitate	10
Preparing for laboratories, homework, reports etc. /Pregătire seminarii/laboratoare, teme,	18

referate, portofolii şi eseuri	
Exams / Examinări	6
Tutoring / Tutorat	4
3.7. Total number of hours of individual study / Total ore studiu individual	66
3.8. Total number of hours per semester / Total ore pe semestru	108
3.9. Number of credits (ECTS) / Număr de credite	4

4. Prerequisites (if it is the case) / Precondiții (acolo unde e cazul)

4.1. curriculum / de curriculum	none
4.2. skills / de competențe	none

5. Requirements (if it is the case) / Condiții (acolo unde e cazul)

5.1. for the lecture / de desfășurare a cursului	lecture room equipped with overhead projector and whiteboard
5.2. for the seminar, laboratory / de desfășurare a seminarului/laboratorului	lab equipped with computers with Racket language installed

6. Acquired skills / Competențe specifice acumulate

Professional skills / Competențe profesionale	●
Transversal skills / Competențe transversale	•

7. Objectives of the course / Obiectivele disciplinei (reieșind din grila competențelor specifice acumulate)

7.1. General objective / Obiectivul general al disciplinei	- aquaintance with the basic notions and principles of functional programming, and their proper usage to solve concrete problems
7.2. Specfic objectives / Obiectivele specifice	- aquaintance with the style of strict functional programming - aquire programming abilities wth the Racket language of functional programming

8. Content / Conținuturi*

8.1. Lecture / Curs	**Teaching strategies / Metode de predare**	**Remarks, details / Observații**
C1. Introduction. Programming paradigms. The Lambda calculus: reduction, evaluation strategies. Functional programming styles.	Lecture, conversation, illustration	
C2. Introduction to Racket.	Lecture, conversation, illustration	

Expressions, variables, functions, strict evaluation. Special forms: control constructs and constructs with local scoping. Bottom-up versus top-down design.		
C3. Lists. Internal representation of lists, dotted lists. Stopping evaluation with quote. Quasiquoted lists. Using lists as data stuctures: constructors, destructors, other list functions; applications.	Lecture, conversation, illustration	
C4. Other composite data structures: mutable pairs, vectors, hash tables. Assignment and mutability. Applications.	Lecture, conversation, illustration	
C5. Repetition through recursion. Types of recursion; tail recursive functions. Applications.	Lecture, conversation, illustration	
C6. Functionals. Passing functions as arguments. Writing functions that take functions as arguments. Lambda expressions. Applications.	Lecture, conversation, illustration	
C7. Repetition through iteration. Repeating actions a number of times. Repeating actions for each element in a list. General examples with do, map, filter, select, and apply.	Lecture, conversation, illustration	
C8. The environment-based model of computation. Lexical scoping and lexical closures. Applications.	Lecture, conversation, illustration	
C9. Object-oriented programming with lexical closures.	Lecture, conversation, illustration	
C10. Lazy evaluation in Racket: with promises and with functional	Lecture, conversation, illustration	

closures. Application: streams.		
C11. Pattern matching. Macros.	Lecture, conversation, illustration	
C12. Applications of Functional Programming in Artificial Intelligence (I).	Lecture, conversation, illustration	
C13. Applications of Functional Programming in Artificial Intelligence (II).	Lecture, conversation, illustration	
C14. Colloqium		

Recommended bibliography / Bibliografie

1) Daniel P. Friedman, Mitchell Wand, Christopher T. Haynes: *Essentials of Programming Languages*. *Second Edition*. The MIT Press. 2001.
2) Matthias Felleisen, Robert Bruce Findler, Matthew Flatt, Shiram Krishnamurthi: *How to Design Programs. An Introduction to Programming and Computing*. The MIT Press. 2003.
3) *Oliver Grillmeyer: Exploring Computer Science with Scheme*. Springer-Verlag. 1998.
4) Harold Abelson, Gerald Jay Sussman, Julie Sussman: *Structure and Interpretation of Computer Programs. Second Edition*. The MIT Press. 1996.
5) Mircea Marin, Viorel Negru, Isabela Dramnesc. *Principles and Practice of Functional Programming*. 2017.
6) Matthew Flatt, Robert Bruce Findler, PLT: *The Racket Guide*. Delivered with the free DrRacket platform.
7) Matthew Flatt, PLT: *The Racket Reference*. Delivered with the free DrRacket platform.

8.2. Seminar, lab / Seminar, laborator	**Teaching/learning strategies / Metode de predare/ invățare**	**Remarks, details / Observații**
L1. Labwork assignments for courses 1 and 2.	Questioning, dialogue, collaborative learning	The students can access directly online the requests for the exercises (http://web.info.uvt.ro/~mmarin/FP). The teacher gives details/explains/answers the students questions and checks/evaluates the way how they solved the exercises.
L2. Labwork assignments for courses 3 and 4.	Questioning, dialogue, collaborative learning	Idem.
L3. Labwork assignments for courses 5 and 6..	Questioning, dialogue, collaborative learning	Idem.
L4. Labwork assignments for courses 7 and 8.	Questioning, dialogue, collaborative learning	Idem.

L5. Labwork assignments for courses 9 and 10.	Questioning, dialogue, collaborative learning	Idem.
L6. Labwork assignments for courses 11 and 12.	Questioning, dialogue, collaborative learning	Idem.
L7. Labwork assignments for courses 13 and 14.	Questioning, dialogue, collaborative learning	Idem.

Recommended bibliography / Bibliografie

1. Daniel P. Friedman, Matthias Felleisen: *The Little Schemer*. Fourth Edition. 1996.

9. Correlations between the content of the course and the requirements of the IT field / Coroborarea conținuturilor disciplinei cu așteptările reprezentanților comunității epistemice, asociațiilor profesionale și angajatorilor reprezentativi din domeniul aferent programului

10. Evaluation / Evaluare*

Activity / Tip de activitate	10.1. Evaluation criteria / Criterii de evaluare**	10.2. Evaluation methods / Metode de evaluare***	10.3. Weight in the averaged mark
10.4. Lecture / Curs	1) Good understanding of the principles of functional logic programming 2)	Colloquim	50%
10.5. Seminar/ lab	Effective use of the principles and idioms of functional programming	1) Individual labwork assignments, evaluated every two weeks. 2) Dialog, team work	25%
	1) Problem solving by recursive thinking 2) Design, implement and tet some simple programs in Racket	Two Racket programming tests, at the mid and end of the semester	25%
10.6. Minimal knowledge for passing			
- ability to solve some simple problems by recursive thinking, and to implement their solutions in Racket - determine the computational complexity (in space and time) of some computations			

Date/ Data completării 1.10.2016

Signature (lecture) / Semnătura titularului de curs

Signature (seminar) Semnătura titularului de seminar

Universitatea de Vest din Timişoara

Signature (director of the department)
Semnătura directorului de departament
Conf.dr. Victoria Iordan