STUDY MODULE DESCRIPTION FORM

Name of the module/subject **Multiparadigm programming**		Code **1010334591010337136**
Field of study **Information Engineering**	Profile of study (general academic, practical) **(brak)**	Year /Semester **5 / 9**
Elective path/specialty **Information Technologies**	Subject offered in: **Polish**	Course (compulsory, elective) **obligatory**
Cycle of study: **First-cycle studies**	Form of study (full-time,part-time) **part-time**	
No. of hours Lecture: **8** Classes: **-** Laboratory: **8** Project/seminars: **-**		No. of credits **3**
Status of the course in the study program (Basic, major, other) **(brak)**	(university-wide, from another field) **(brak)**	
Education areas and fields of science and art **technical sciences**		ECTS distribution (number and %) **3 100%**

Responsible for subject / lecturer:

dr inż. Grażyna Brzykcy
email: grazyna.brzykcy@put.poznan.pl
tel. 616653714
Wydział Elektryczny
ul. Piotrowo 3A 60-965 Poznań

Responsible for subject / lecturer:

dr inż. Adam Meissner
email: adam.meissner@put.poznan.pl
tel. 616653714
Wydział Elektryczny
ul. Piotrowo 3A 60-965 Poznań

Prerequisites in terms of knowledge, skills and social competencies:

1	**Knowledge**	Student has basic knowledge of logic, theory of recursive functions, imperative and declarative programming, object-oriented programming, data bases, operating systems and computer networks.
2	**Skills**	Student is able to acquire information from literature, data bases and other sources; student is able to integrate acquired information, to interpret it, to draw conclusions and to formulate and justify judgments. Student is able to communicate in English and to read descriptions and manuals of software tools, applications and similar documents.
3	**Social competencies**	Student understands the necessity and possibility of continuous education and development of different skills (linguistic, professional, personal and social). Student understands a responsibility associated to his own work. Student is able to adhere to team work rules and to take responsibility for cooperative tasks.

Assumptions and objectives of the course:

An overview of computation paradigms and presentation of basic concepts, techniques and programming abstractions. Acquiring the skills of selecting an appropriate computation model for a given problem; gaining the practice in multiparadigm programming.

Study outcomes and reference to the educational results for a field of study

Knowledge:

1. Student has organized knowledge with theoretical foundations of basic program constructions, algorithm implementations, paradigms and programming styles, software verification methods, formal languages, compilers, platforms. - [[K_W05]]

Skills:

1. Student is able to use software platforms and environments for simple programs encoding, running and testing in imperative, object-oriented and declarative programming languages. - [[K_U10]]

Social competencies:

1. Student understands the importance of stringent accomplishment of a given project with proper notation standards, proper language. Student understands the importance of keeping deadlines. - [[K_K07]]

Assessment methods of study outcomes

Lecture

Written test based on lecture (basic concepts and simple tasks).

Laboratory

Students? marks are based on continuous assessment of their programming activity and results of two written tests (creation of simple programs).

Course description

Lectures

Declarative computation paradigm. Concepts and techniques of the functional and deterministic logic programming. Iterative and recursive programming, metaprograming, abstract data types. Declarative concurrency. Programming models with an explicit state. A class as a data abstraction in object-oriented programming. Relational programming and data bases. Distributed programming in open systems. Constraint programming.

Laboratory

Creation of simple programs in multiparadigm programming environment Mozart with programming languge Oz.

Basic bibliography:

1. Roy P. van, Haridi S.: Concepts, Techniques and Models of Computer Programming, The MIT Press, 2004.
2. Mozart Consortium: The Mozart programming system, http://www.mozart-oz.org, 2006.

Additional bibliography:

1. Kowalski R.: Logic for problem solving, North-Holland, 1979.

Result of average student's workload

Activity	Time (working hours)
1. Lecture	8
2. Laboratory	8
3. Preparation to laboratory and tests	48

Student's workload

Source of workload	hours	ECTS
Total workload	64	3
Contact hours	16	1
Practical activities	48	2