**SYLLABUS**

**1. Information regarding the programme**

| | |
|---|---|
| 1.1 Higher education institution | **Babeş Bolyai University** |
| 1.2 Faculty | **Faculty of Mathematics and Computer Science** |
| 1.3 Department | **Department of Computer Science** |
| 1.4 Field of study | **Computer Science** |
| 1.5 Study cycle | **Master** |
| 1.6 Study programme / Qualification | **High Performance Computing and Big Data Analytics** |

**2. Information regarding the discipline**

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| 2.1 Name of the discipline | **Programming paradigms** | | | | | | |
| 2.2 Course coordinator | **Prof.PhD. Bazil Parv** | | | | | | |
| 2.3 Seminar coordinator | **Prof.PhD. Bazil Parv** | | | | | | |
| 2.4. Year of study | **1** | 2.5 Semester | **1** | 2.6. Type of evaluation | **E** | 2.7 Type of discipline | **compulsory** |
| 2.8 Code of the discipline | **MME8028** | | | | | | |

**3. Total estimated time** (hours/semester of didactic activities)

| | | | | | |
|---|---|---|---|---|---|
| 3.1 Hours per week | **4** | Of which: 3.2 course | **2** | 3.3 seminar/lab | **1s+1pr** |
| 3.4 Total hours in the curriculum | **56** | Of which: 3.5 course | **28** | 3.6 seminar/lab | **28** |
| Time allotment: | | | | | Hours |
| Learning using manual, course support, bibliography, course notes | | | | | **22** |
| Additional documentation (in libraries, on electronic platforms, field documentation) | | | | | **22** |
| Preparation for seminars/labs, homework, papers, portfolios and essays | | | | | **20** |
| Tutorship | | | | | **15** |
| Evaluations | | | | | **15** |
| Other activities: ..... | | | | | - |

| | |
|---|---|
| 3.7 Total individual study hours | **94** |
| 3.8 Total hours per semester | **150** |
| 3.9 Number of ECTS credits | **6** |

**4. Prerequisites** (if necessary)

| | |
|---|---|
| 4.1. curriculum | • Fundamentals of Programming<br>• Object-Oriented Programming<br>• Functional and Logic Programming |
| 4.2. competencies | • Average programming skills |

**5. Conditions** (if necessary)

| | |
|---|---|
| 5.1. for the course | • Videoprojector, Internet access |
| 5.2. for the seminar /lab activities | • Computers, Internet access, UML tool |

**6. Specific competencies acquired**

| | |
|---|---|
| **Professional competencies** | • Understanding and working with basic concepts in computer programming;<br>• Capability of analysis and synthesis;<br>• Proficient use of tools and languages specific to software systems development;<br>• Knowing the specifics of main programming paradigms. |
| **Transversal competencies** | • Professional communication skills; concise and precise description, both oral and written, of professional results;<br>• Independent work capabilities; able to fulfill different roles;<br>• Antepreneurial skills. |

**7. Objectives of the discipline** (outcome of the acquired competencies)

| | |
|---|---|
| 7.1 General objective of the discipline | • Know and understand fundamental concepts of programming.<br>• Be able to apply different programming paradigms to different programming projects |
| 7.2 Specific objective of the discipline | At the end of the course, students should<br>• know the main features of different programming paradigms: procedural, object-oriented, concurrent, functional, logical, event-based, scripting<br>• have a good understanding of the following concepts: value, type, variable, binding, procedural abstraction, data abstraction, object, class, component, interface, polymorphism;<br>• learn the similarities and differences between different programming paradigms in terms of the concepts they implement |

**8. Content**

| 8.1 Course | Teaching methods | Remarks |
|---|---|---|
| 1. *Programming paradigms*. Definitions. Main programming paradigms. Programming styles. Evolution of programming languages | • Interactive exposure<br>• Explanation<br>• Conversation<br>• Didactical demonstration | |
| 2. *Basic concepts 1*. Values and types. Variables and storage | • Interactive exposure<br>• Explanation<br>• Conversation<br>• Didactical demonstration | |
| 3. *Basic concepts 2*. Bindings and scope. Control flow | • Interactive exposure<br>• Explanation<br>• Conversation<br>• Didactical demonstration | |
| 4. *Advanced concepts 1*. Type systems. Composite types | • Interactive exposure<br>• Explanation<br>• Conversation<br>• Didactical demonstration | |
| 5. *Advanced concepts 2*. Subroutines and control abstraction (procedural abstraction) | • Interactive exposure<br>• Explanation<br>• Conversation<br>• Didactical demonstration | |
| *6. Advanced concepts 3*. Data abstraction and object orientation. Generic abstraction | • Interactive exposure<br>• Explanation<br>• Conversation<br>• Didactical demonstration | |
| 7. *Advanced concepts 4*. Errors and events. Concurrency | • Interactive exposure<br>• Explanation<br>• Conversation | |

| | | |
|---|---|---|
| | • Didactical demonstration | |
| 8. *Paradigms 1*. Imperative programming | • Interactive exposure<br>• Explanation<br>• Conversation<br>• Didactical demonstration | |
| 9. *Paradigms 2*. Object-oriented programming | • Interactive exposure<br>• Explanation<br>• Conversation<br>• Didactical demonstration | |
| 10. *Paradigms 3*. Concurrent programming | • Interactive exposure<br>• Explanation<br>• Conversation<br>• Didactical demonstration | |
| 11. *Paradigms 4*. Functional programming | • Interactive exposure<br>• Explanation<br>• Conversation<br>• Didactical demonstration | |
| 12. *Paradigms 5*. Logic programming | • Interactive exposure<br>• Explanation<br>• Conversation<br>• Didactical demonstration | |
| 13. *Paradigms 6*. Event-driven programming | • Interactive exposure<br>• Conversation | |
| 14. *Paradigms 7*. Scripting | • Interactive exposure<br>• Explanation<br>• Conversation<br>• Didactical demonstration | |

**Bibliography**


1. SCOTT, MICHAEL L.: *Programming Language Pragmatics*, 4th ed, Morgan-Kaufmann, 2016
2. SEBESTA, ROBERT W.: *Concepts of Programming Languages*, 10th ed, Pearson Education, 2012
3. SZYPERSKI, CLEMENS: *Component Software. Beyond Object-Oriented Programming*, Addison-Wesley (1st ed. 1998, 2nd ed. 2002 with GRUNTZ, DOMINIK and MURER, STEFAN).
4. STROUSTRUP, BJARNE: *The C++ Programming Language Special Edition*, Addison-Wesley, 2000 chapter 2
5. VAN ROY, PETER; HARIDI, SEIF: *Concepts, Techniques and Models of Computer Programming*, MIT Press, 2004
6. WATT, David A.: *Programming Language Design Concepts*, Wiley, 2004
7. WEGNER, PETER: Concepts and paradigms of OOP, OOPSLA '89 Keynote talk


| 8.2 Seminar / laboratory | Teaching methods | Remarks |
|---|---|---|
| 1. Establishing the programming language PL for paper<br>Paper title: Programming language analysis – PL<br>Requirements for paper<br>Calendar/schedule of seminars/activities<br>Weeks 1-4 | Conversation, debate, case studies, presentations | Seminar is organized as a total of 14 hours – 2 hours every other week |
| 2. Paper presentations phase 1<br>Weeks 5-9 | Presentation, discussion | |
| 3. Paper presentation phase 2<br>Weeks 10-14 | Presentation, discussion | |
| 4. Paper final version<br>Week 14 | | |

**Bibliography**

Students will search and use programming paradigms documentation

- on the department server (win/labor/Romana/master/PP)
- on the web, using main CS databases

The ELISA project http://jklunder.home.xs4all.nl

**9. Corroborating the content of the discipline with the expectations of the epistemic community, professional associations and representative employers within the field of the program**

- This course follows the IEEE and ACM Curriculla Recommendations for Software Engineering studies;
- Courses with similar content are taught in the major universities in Romania offering similar study programs;
- Course content is considered very important by the software companies for improving average software development skills

**10. Evaluation**

| Type of activity | 10.1 Evaluation criteria | 10.2 Evaluation methods | 10.3 Share in the grade (%) |
|---|---|---|---|
| 10.4 Course | • Knowledge of the basic concepts of programming<br>• Ability to apply different paradigms to different problem domains | • Written exam grade | 40% |
| 10.5 Seminar/lab activities | • Ability to study and analyse literature regarding a concrete programming language | • Paper grade<br>• Seminar/lab attendance<br>• Default | 40%<br>10%<br>10% |
| Paper evaluation detailed | | • Scheduling and presentation<br>• Presentation phase 1<br>• Presentation phase 2<br>• Compliance to general requirements<br>• Structure<br>• Content | 2 x 5%<br>10%<br>10%<br>10%<br>20%<br>40% |
| 10.6 Minimum performance standards | | | |
| • At least grade 5 (from a scale of 1 to 10) at written exam, and paper work. | | | |

| Date | Signature of course coordinator | Signature of seminar coordinator |
|---|---|---|
| April 19, 2018 | Prof.PhD. Bazil PARV | Prof.PhD. Bazil PARV |
| Date of approval | | Signature of the head of department |
| ..... | | Prof.PhD. Anca ANDREICA |