<table>
<thead>
<tr>
<th>1st semester</th>
<th>2nd semester</th>
<th>3rd semester</th>
<th>4th semester</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Obligatory courses</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Ad Hoc Networks (2+2+1) Tamás LUKOVSZKI</td>
<td>Component-based software development (2+2+1) László Zsolt VARGA</td>
<td>Lab I-II. (8 kr.)</td>
<td>Lab III-IV. (8 kr.)</td>
</tr>
<tr>
<td>Data mining and information retrieval (2+2+1) András BENCZÚR</td>
<td>Interactive media design and development (2+2+1) Mártá TURCSÁNYI-SZABÓ</td>
<td>Building distributed systems (2+2+1) Tamás KOZSIK</td>
<td>Advanced cryptography (2+2+2) Péter SZIKLAI</td>
</tr>
<tr>
<td>Type models (2+0+0) László Zsolt VARGA</td>
<td>Advanced functional programming (2+2+1) Zoltán HORVÁTH</td>
<td>Advanced Java programming (2+2+1) Tamás KOZSIK</td>
<td>Applied cryptography project seminar (2+2+2) Péter SZIKLAI</td>
</tr>
<tr>
<td>Synthesis and verification (2+0+1) Tibor GREGORICS</td>
<td>Agile project management in informatics (2+2+1) Zoltán ISTENES</td>
<td>Analysis of distributed systems (2+2+1) Máté TEJFEL</td>
<td>Cryptography protocols (2+2+0) Attila KOVÁCS</td>
</tr>
<tr>
<td>High assurance object oriented software engineering (2+2+1) Sándor SIKE</td>
<td>Formal semantics (2+0+1) Zoltán HORVÁTH</td>
<td>Design of distributed systems (2+2+1) Zoltán HORVÁTH</td>
<td>Cryptography and its applications (2+2+0) Attila KOVÁCS</td>
</tr>
<tr>
<td>Web engineering (2+2+1) Zoltán ILLÉS</td>
<td>Formal methods in software development (2+2+1) Zoltán ISTENES</td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Optional courses</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Optional course (2+2+1)</td>
<td>Optional course (2+0+0)</td>
<td>Optional course (2+2+0)</td>
<td></td>
</tr>
<tr>
<td><strong>Total credits</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>30 credits</td>
<td>30 credits</td>
<td>32 credits</td>
<td>28 credits</td>
</tr>
</tbody>
</table>
Name of the course: Ad Hoc Networks
Faculty member responsible for the course: Dr. Tamás Lukovszki, associate professor
Responsible department: Faculty of Informatics, Department of Information Systems
Total credits: 5
Total hours: 5
Type of the course | lecture | practice | consultation
---|---|---|---
Hours per week | 2 | 2 | 1
Type of testing | exam | practice |
Topics:
Ad hoc networks do not use any extra infrastructure. The nodes of the network use a wireless communication interface and communicate directly and provide the routing necessary to deliver messages over multiple hops. We discuss medium access, routing algorithms, and methods dealing with the mobility of participants. The topics of the course: Modeling networks, capacity of wireless networks, topology control, routing, distributed localization, energy, dilation, congestion, mobility models.

Literature:

Name of the course: Data mining and information retrieval
Faculty member responsible for the course: Dr. András Benczúr, professor
Responsible department: Faculty of Informatics, Department of Information Systems
Total credits: 5
Total hours: 5
Type of the course | lecture | practice | consultation
---|---|---|---
Hours per week | 2 | 2 | 1
Type of testing | exam | practice |
Prerequisites:
The course requires basic knowledge in calculus, probability theory, and linear algebra. Knowledge of graphs and basic algorithms is an advantage.

Topics:
The aim of the course is to provide a basic, but comprehensive introduction to data mining. By the end of the course, students will be able to build models, choose algorithms, implement and evaluate them.
Detailed Program and Class Schedule:
1. Motivations for data mining. Examples of application domains. Methodology of knowledge discovery in databases (KDD) and data mining (DM). Formulation of main problems of data mining.
5. Introduction to the WEKA data mining software. Classification with WEKA.
11. Dimensionality reduction by spectral methods, singular value decomposition, low-rank approximation.
12. Search engines, web information retrieval, PageRank and beyond.

Literature:
Jiawei Han és Micheline Kamber: Data Mining: Concepts and Techniques, 2nd ed., Morgan Kaufmann Publishers, 2006.

Name of the course: Type models
Faculty member responsible for the course: Dr. László Zsolt Varga, associate professor
Responsible department: Faculty of Informatics, Department of Software Technology And Methodology
Total credits: 2
Total hours: 2

<table>
<thead>
<tr>
<th>Type of the course</th>
<th>lecture</th>
<th>practice</th>
<th>consultation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Hours per week</td>
<td>2</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Type of testing</td>
<td>exam</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Topics:

Literature:

Recommended literature:
Annabelle Mclver, Caroll Morgan ed. Programming Methodology, Monographs in computer
**Name of the course:** Synthesis and verification

**Faculty member responsible for the course:** Dr. Tibor Gregorics, associate professor

**Responsible department:** Faculty of Informatics, Department of Software Technology And Methodology

**Total credits:** 3

**Total hours:** 3

<table>
<thead>
<tr>
<th>Type of the course</th>
<th>lecture</th>
<th>practice</th>
<th>consultation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Hours per week</td>
<td>2</td>
<td></td>
<td>1</td>
</tr>
<tr>
<td>Type of testing</td>
<td></td>
<td></td>
<td>exam</td>
</tr>
</tbody>
</table>

**Topics:**
- Introduction. Basic notions of programs. The syntax and semantics of nondeterministic programs. The basic notion of correct programs. The Floyd method for proving partial correctness of flow charts programs. Examples.
- Formal definition of Floyd methods. Examples.
- Partial and total correctness of structured programs by Hoare methods. Examples.
- The Hoare’s methods are corrects and sounds. Theorems.
- The basic notions of parallel and concurrent programs. Hardware architectures and software architectures. Strategies of implementations and languages tools. Examples.
- Behavioural analysis of concurrent programs using semaphores, monitors, resources, remote procedure calls, message passing. Examples.
- Owicki’s and Gries’s method for proving the partial correctness of parallel programs. Examples.
- Owicki’s and Gries’s method for proving the total correctness of parallel programs. Free from deadlock and starvation, problems of termination. Examples.
- Formal derivation of weakly and strongly correct concurrent programs. Examples.
- Contracts and proofs.
- Contracts as a way of modeling a collection of agents.
- Summary.

**Literature:**
- Kröger, F.: Temporal Logic of Programs (Springer-Verlag, 1987)
<table>
<thead>
<tr>
<th>Hours per week</th>
<th>2</th>
<th>2</th>
<th>1</th>
</tr>
</thead>
<tbody>
<tr>
<td>Type of testing</td>
<td>exam</td>
<td>practice</td>
<td></td>
</tr>
</tbody>
</table>

**Topics:**

**Literature:**
Erich Gamma, Richard Helm, Ralph Johnson, John Vlissides: Design Patterns - Elements of Reusable Object-Oriented Software (Addison-Wesley Longman, Inc. 1995)


J. Warmer, A. Kleppe: The Object Constraint Language, Precise Modeling with UML (Addison-Wesley, 1999)


**Name of the course:** Web engineering
**Faculty member responsible for the course:** Dr. Zoltán Illés, associate professor
**Responsible department:** Faculty of Informatics, Department of Media & Educational Informatics

**Total credits:** 5
**Total hours:** 5

<table>
<thead>
<tr>
<th>Type of the course</th>
<th>lecture</th>
<th>practice</th>
<th>consultation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Hours per week</td>
<td>2</td>
<td>2</td>
<td>1</td>
</tr>
<tr>
<td>Type of testing</td>
<td>exam</td>
<td>practice</td>
<td></td>
</tr>
</tbody>
</table>

**Topics:**
This curriculum introduces the students with the modern, state-of-the-art client and server side web technologies, methodologies of web engineering, the programming and design patterns, especially with the web service oriented architectures. By the end of the course the student has a global overview of the up-to-date web trends and technologies, and, with the help of them, is able to develop a web application and web information systems.

Introduction to Web Technologies and Web Engineering: specialties, characteristics, categories of web applications.
Web Architectures: multi-tier, data-centric architectures,
Requirement Analysis of Web Applications
Specialties of Large Enterprise and Small and Medium Enterprise Web Applications
Development Process of Web Applications
Model-Based Web Application Design and Development, WebML
Testing, Quality Management.
Design of Web 2.0 és Enterprise 2.0 Applications
Web Business Models
Web project management
Design of Mobile Web Applications
Semantic Web Applications, integration to Web Information Systems
Web Application Models, Cloud computing
Service Oriented Architectures, Web Information Systems

**Literature:**

---

**Name of the course:** Component-Based Software Development

**Faculty member responsible for the course:** Dr. László Zsolt Varga, associate professor

**Responsible department:** Faculty of Informatics, Department of Software Technology And Methodology

**Total credits:** 5

**Total hours:** 5

<table>
<thead>
<tr>
<th>Type of the course</th>
<th>lecture</th>
<th>practice</th>
<th>consultation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Hours per week</td>
<td>2</td>
<td>2</td>
<td>1</td>
</tr>
<tr>
<td>Type of testing</td>
<td>exam</td>
<td>practice</td>
<td></td>
</tr>
</tbody>
</table>

**Topics:**
Component-Based Software Development

Introduction, basic notions. The notion of software development model. The UML as a primary model-based notation for all analysis and design activities. The notions of meta model and verified components.

The tasks and types of the component models. The interface description language (IDL). The notion and roles of the middleware. The short overview of CORBA, COM/DCOM and JavaBeans/EJB.

The notions and types of software architectures. The relationships of reference models, architectural patterns, reference architectures and software architectures. The main characteristics of the J2EE/EJB architecture.

Principles of the KobrA Method. Context realization defines the environment of the target system by enterprise or business process model, by usage model, by structural model and by interaction or behavioural model.

The main descriptive artifacts in a component specification are a functional model, a behavioral model and a structural model.

The main descriptive artifacts in a component realization are an interaction model, a behavioural or algorithmic model and a structural model.

COTS component: a commercial off-the-shelf component is a ready-to-use physical component from a third party that can be incorporated into an application.

SYNTHESIS: a tool for creating correct system from COTS components.

The new version of the SYNTHESIS tool.

Propositional Temporal Logic: syntax and semantics. Axiomatization of Propositional Temporal Logic.

First-Order Temporal Logic and its Semantics. Temporal semantics of concurrent programs. A
correctness proof method: the sometime system.
The tableau-based methods for synthesizing correct concurrent programs.
Model checking as an automatic technique for verifying finite state concurrent systems
Component verification by model checking

**Literature:**
Clarke, E. M. Jr., Grumberg, O., Peled, D. A.: Model Checking (The MIT Press, 1999)
Gross, H-G: Component-based Software Testing with UML (Springer-Verlag, 2005)

**Recommended literature:**
Kröger, F.: Temporal Logic of Programs (Springer-Verlag, 1987)

---

**Name of the course:** Interactive Media Design and Development

**Faculty member responsible for the course:** Mártar Turcsányszabó, associate professor

**Responsible department:** Faculty of Informatics, Department of Media & Educational Informatics

<table>
<thead>
<tr>
<th>Total credits: 5</th>
<th>Total hours: 5</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Type of the course</strong></td>
<td><strong>lecture</strong></td>
</tr>
<tr>
<td>Hours per week</td>
<td>2</td>
</tr>
<tr>
<td>Type of testing</td>
<td>exam</td>
</tr>
</tbody>
</table>

**Topics:**
The course introduces Human–Computer Interaction (HCI) involving the study, planning, and
design of the interaction between people (users) and computers.
Its aim is to understand the theoretical basics of Perception, Multimedia design, Information
Visualization, Interaction Design, the Virtual Continuum, Serious Games, Tangible,
Collaborative, Location-based, and Gesture-based technologies, etc.) and recent innovations in
these areas.
Activities involve the exploration of emerging interactive technologies designed for
demonstration, education, entertainment, navigation, narrative, support …etc. purposes and
their variety of creative applications in different disciplines and user interest groups.
Students from different disciplines form groups to design and implement a specified innovative
project that could well serve the basis of an industrial entrepreneurship.

**Literature:**


The Functional Art: An Introduction to Information Graphics and Visualization (Peachpit/Pearson
Recommended literature:
The Encyclopedia of Human-Computer Interaction, 2nd Ed. At: http://www.interaction-design.org/books/hci.html

Journal of Virtual World Research: http://jvwresearch.org/

Horizon Reports: http://www.nmc.org/horizon-project

Papers submitted to conferences:
- Museums and the Web: http://www.museumsandtheweb.com/
- CHI: http://chi2013.acm.org/
- IED: http://europe.immersiveeducation.org/events/ied-europe-summit-2012
- DIS: http://www.dis2012.org/

Name of the course: Advanced Functional Programming
Faculty member responsible for the course: Dr. Zoltán Horváth, professor
Responsible department: Faculty of Informatics, Department of Programming Languages And Compilers
Total credits: 5
Total hours: 5

<table>
<thead>
<tr>
<th>Type of the course</th>
<th>lecture</th>
<th>practice</th>
<th>consultation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Hours per week</td>
<td>2</td>
<td>2</td>
<td>1</td>
</tr>
<tr>
<td>Type of testing</td>
<td>exam</td>
<td>practice</td>
<td></td>
</tr>
</tbody>
</table>

Topics:
Algebraic types, type classes.
Higher-order types, existential types.
Uniqueness typing.
Dynamics, generic programming.
Purely functional data structures.
Parallel and distributed programming.
Combinators, combinator libraries.
Monadic programming.
Interactive programs, Functional Reactive Programming.
Embedded domain-specific languages.

Literature:
Hudak, P. *The Haskell School of Expression, 1st Edition*. Cambridge University Press, February

**Recommended literature:**

---

**Name of the course:** Agile project management in informatics  
**Faculty member responsible for the course:** Dr. Zoltán Istenes, associate professor  
**Responsible department:** Faculty of Informatics, Department of Software Technology And Methodology  
**Total credits:** 5  
**Total hours:** 5  

<table>
<thead>
<tr>
<th>Type of the course</th>
<th>lecture</th>
<th>practice</th>
<th>consultation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Hours per week</td>
<td>2</td>
<td>2</td>
<td>1</td>
</tr>
<tr>
<td>Type of testing</td>
<td>exam</td>
<td>practice</td>
<td></td>
</tr>
</tbody>
</table>

**Topics:**
- Classical Project Management Basic  
- Agile Introduction  
- Scrum overview  
- Roles, responsibilities  
- Scrum events, meetings, environment  
- Artifacts, documents  
- Product backlog, Sprint backlog  
- Sprint planning, running, review, retrospective  
- Estimation, Velocity  
- Backlog grooming  
- Release planning  
- Different burndown charts and agile metrics  
- Scrum of Scrums  
- Scrum simulation  
- Areas of application, scaling Scrum  
- Distributed projects  
- Agile transformation steps  
- Common pitfalls  
- Other topics  
- Lean and agile?  
- Kanban  
- Kanban vs Scrum  
- Exercises  
- Requirement engineering basics and refresher
Name of the course: Formal semantics
Faculty member responsible for the course: Dr. Zoltán Horváth, professor
Responsible department: Faculty of Informatics, Department of Programming Languages And Compilers
Total credits: 3
Total hours: 3

<table>
<thead>
<tr>
<th>Type of the course</th>
<th>lecture</th>
<th>practice</th>
<th>consultation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Hours per week</td>
<td>2</td>
<td></td>
<td>1</td>
</tr>
<tr>
<td>Type of testing</td>
<td>exam</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Topics:
Introduction: motivation, approaches to semantics definitions
Translational semantics, attribute grammars and their applications
Denotational and operational semantics of expressions
Natural semantics of imperative statements
Structural operational semantics of imperative statements
Semantics of abort, nondeterministic and parallel execution
Denotational semantics of imperative statements
Domain and fixed point theory
Semantics of functional language elements
Modeling blocks and procedures
Modeling exceptions
Full abstraction

Classification of semantic descriptions. Attribute grammars.
Translational semantics using attribute grammars.
Two-level grammar defining the syntax and semantics of programming languages.
Denotational semantics. Direct style semantics. Requirements of the fixed point.
Fixed point theory, definitions, theorems.
Direct style semantics: existence.
Extensions of While language with blocks declaring local variables and procedures. Examples.
Continuation style semantics for While language.
Structural operational semantics and its properties. An equivalence result: For every statement S of While language we have $S_{ns}[S] = S_{sos}[S]$.
Extensions of While language: abortion, non-determinism, parallelism. Operational semantics of a collection of agents that work within the limits set by the contract. Examples.
For every statement S of While language the structural operational semantics and the direct style denotational semantics are equivalent. Theorems.
The operational approach: Vienna Definition Language (VDL).
Summary.
<table>
<thead>
<tr>
<th>Name of the course:</th>
<th>Formal methods in software development</th>
</tr>
</thead>
<tbody>
<tr>
<td>Faculty member responsible for the course:</td>
<td>Dr. Zoltán Istenes, associate professor</td>
</tr>
<tr>
<td>Responsible department:</td>
<td>Faculty of Informatics, Department of Software Technology And Methodology</td>
</tr>
<tr>
<td>Total credits:</td>
<td>5</td>
</tr>
<tr>
<td>Total hours:</td>
<td>5</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Type of the course</th>
<th>lecture</th>
<th>practice</th>
<th>consultation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Hours per week</td>
<td>2</td>
<td>2</td>
<td>1</td>
</tr>
<tr>
<td>Type of testing</td>
<td>exam</td>
<td>practice</td>
<td></td>
</tr>
</tbody>
</table>

Topics:

Tools: AtelierB, Click’n’prove, ProB, B4free, Rodin.

Literature:

---

<table>
<thead>
<tr>
<th>Name of the course:</th>
<th>Building distributed applications</th>
</tr>
</thead>
<tbody>
<tr>
<td>Faculty member responsible for the course:</td>
<td>Dr. Tamás Kozsik, associate professor</td>
</tr>
<tr>
<td>Responsible department:</td>
<td>Faculty of Informatics, Department of Programming Languages And Compilers</td>
</tr>
<tr>
<td>Total credits:</td>
<td>5</td>
</tr>
<tr>
<td>Total hours:</td>
<td>5</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Type of the course</th>
<th>lecture</th>
<th>practice</th>
<th>consultation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Hours per week</td>
<td>2</td>
<td>2</td>
<td>1</td>
</tr>
<tr>
<td>Type of testing</td>
<td>exam</td>
<td>practice</td>
<td></td>
</tr>
</tbody>
</table>

Topics:
The course presents some important application domains for distributed programming, with special regard to present software industry challenges and scientific computations. After the completion of the course the students will not only understand the theoretical issues of distributed computing, but they will also be capable of designing and implementing distributed applications in general, and distributed object systems in particular. They will also learn
common technologies used in the software industry. The following topics will be addressed (related technologies that can be used for illustration purposes are in parentheses).

Multi-tier application model: Modularization of large software systems, optimal use of distributed architectures in the design of the components (with respect to efficiency and high availability). Transactional applications backed by information systems. (Java EE, JDBC, JPA, JTA)

Remote Procedure Call: (Java RMI, EJB)
Message-based communication: (JMS, PVM/MPI)
Web-programming: Web-applications (Java servlet, JSP, JSF), web-services (JAX-WS)
Component lookup: (JNDI, Jini).
Code mobility: (Java applet)
Grid systems: fulfilling high computational requirements.
Aspect-oriented programming: Used in the implementation of the above technologies. (AspectJ)

Literature:

Name of the course: Advanced Java Programming
Faculty member responsible for the course: Dr. Tamás Kozsik, associate professor
Responsible department: Faculty of Informatics, Department of Programming Languages And Compilers
Total credits: 5
Total hours: 5

<table>
<thead>
<tr>
<th>Type of the course</th>
<th>lecture</th>
<th>practice</th>
<th>consultation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Hours per week</td>
<td>2</td>
<td>2</td>
<td>1</td>
</tr>
<tr>
<td>Type of testing</td>
<td>exam</td>
<td>practice</td>
<td></td>
</tr>
</tbody>
</table>

Topics:
The purpose of the course is to acquire knowledge on, and enhance competence in, Java Standard Edition, beyond the fundamental language concepts and standard libraries.
- Generic definitions
- Annotations
- Reflection
- Multithreading
- Memory management, garbage collection
- Input-output, serialization
- Database management and persistence: JDBC and the fundamentals of JPA
- Network programming: TCP and UDP; HTTP
- Program design principles and best practices
- Exceptions, assertions
- Logging and testing

Literature:
James Gosling, Bill Joy, Guy Steele, Gilad Bracha. The Java Language
Name of the course: Analysis of Distributed Systems
Faculty member responsible for the course: Dr. Máté Tejfel, assistant professor
Responsible department: Faculty of Informatics, Department of Programming Languages And Compilers
Total credits: 5
Total hours: 5

<table>
<thead>
<tr>
<th>Type of the course</th>
<th>lecture</th>
<th>practice</th>
<th>consultation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Hours per week</td>
<td>2</td>
<td>2</td>
<td>1</td>
</tr>
<tr>
<td>Type of testing</td>
<td>exam</td>
<td>practice</td>
<td></td>
</tr>
</tbody>
</table>

Topics:
The goal of the subject is to give an overview for the student about how can we explain the parallel behaviour by algebraic methods and Petri-nets, and how work applications based on that models in practice.
The basic concepts of the course are processes, computational processes, parallelism, operations of processes, compositions of processes and properties of processes (liveness, deadlock-free, etc.). The theory of Petri-nets is explored more partially with many modelling example. The behavioural and structural properties, methods of analysis, famed subclasses and relationships between these subclasses are investigated. We define theorems about liveness, safetyness and reachability and present transformation, which preserve these properties. The course introduces the Petri-boxes, a special class of Petri-nets, which help us to model the program structures (sequences, branches and loops). Some tools for simulation and analysis of Petri-nets are also investigated. The second part of the course introduces the theory of algebraic models through a given example. The properties of the models, the methods of descriptions of processes and the possible compositions are examined. The denotational, operational and axiomatic semantics of the model is given and the relationships of these different descriptions are investigated.

Teaching methods: There will be lectures introducing the formal specification and properties of Petri nets and algebraic models and exercises where the students will create concrete examples. There will be also programming exercises where the students can use the learned methods.

Literature:

Name of the course: Design of Distributed Systems
Faculty member responsible for the course: Dr. Zoltán Horváth, professor
Responsible department: Faculty of Informatics, Department of Programming Languages And Compilers
Total credits: 5
Total hours: 5

<table>
<thead>
<tr>
<th>Type of the course</th>
<th>lecture</th>
<th>practice</th>
<th>consultation</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Hours per week</td>
<td>2</td>
<td>2</td>
<td>1</td>
</tr>
<tr>
<td>----------------</td>
<td>---</td>
<td>---</td>
<td>---</td>
</tr>
<tr>
<td>Type of testing</td>
<td>exam</td>
<td>practice</td>
<td></td>
</tr>
</tbody>
</table>

**Topics:**
Students will be able to express and verify the properties of the distributed programs using formal methods, apply different ways to create advanced compositions of simple programs, and solutions for interesting and difficult problems in a distributed way.
Dining/drinking philosophers, formal specification of distributed problems, properties of distributed systems, safety and progress properties of distributed programs, verification of safety critical properties, program compositions from components with proven properties, computing the value of an associative function, message channels, pipelined networks programming exercises where the students apply the learned methods in the practice.

**Literature:**

---

<table>
<thead>
<tr>
<th>Name of the course:</th>
<th>Advanced cryptography</th>
</tr>
</thead>
<tbody>
<tr>
<td>Faculty member responsible for the course:</td>
<td>Dr. Péter Sziklai, associate professor</td>
</tr>
<tr>
<td>Responsible department:</td>
<td>Faculty of Science, Institute of Mathematics, Department of Computer Science</td>
</tr>
<tr>
<td>Total credits:</td>
<td>6</td>
</tr>
<tr>
<td>Total hours:</td>
<td>6</td>
</tr>
<tr>
<td>Type of the course</td>
<td>lecture</td>
</tr>
<tr>
<td>Hours per week</td>
<td>2</td>
</tr>
<tr>
<td>Type of testing</td>
<td>exam</td>
</tr>
</tbody>
</table>

**Topics:**
The course have two main goals: discovering the mathematical background beyond several cryptographic constructions and introducing novel cryptographic primitives using interesting results from various topics of mathematics or computer science. For the first part, we present the necessary exact definitions, precise assumptions and rigorous proofs of security. For the second part, we present recent results, methods and its connections to cryptographic problems from finite fields to linear algebra.
Perfect and computational security, proofs by reduction, security definitions, pseudorandomness, message authentication codes, collision-resistant hash functions, one-way functions, cryptographic hardness assumptions, primality testing, factoring and computing discrete logarithms, arithmetics in finite fields and its applications, elliptic curve based cryptography, lattice based constructions, secure multiparty computation, secret sharing problems, applications for e-commerce.

**Literature:**
<table>
<thead>
<tr>
<th>Name of the course:</th>
<th>Applied cryptography project seminar</th>
</tr>
</thead>
<tbody>
<tr>
<td>Faculty member responsible for the course:</td>
<td>Dr. Péter Sziklai, associate professor</td>
</tr>
<tr>
<td>Responsible department:</td>
<td>Faculty of Science, Institute of Mathematics, Department of Computer Science</td>
</tr>
<tr>
<td>Total credits:</td>
<td>6</td>
</tr>
<tr>
<td>Total hours:</td>
<td>6</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Type of the course</th>
<th>lecture</th>
<th>practice</th>
<th>consultation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Hours per week</td>
<td>2</td>
<td>2</td>
<td>2</td>
</tr>
<tr>
<td>Type of testing</td>
<td>exam</td>
<td>practice</td>
<td></td>
</tr>
</tbody>
</table>

**Topics:**
The objective of the course is to develop and strengthen the ability to complete miniprojects, working in small groups (3 persons approx.). The practical aspects of the learned cryptographical solutions is emphasized, as well as focused team work concentrated on modeling and solving a security problem originated in a real, practical situation.

**Literature:**
van Lint, J.H.: Introduction to coding theory. Springer Verlag, 1982
Roman, S.: Coding and information theory. Springer Verlag, 1992
Beutelspacher, A.: Cryptology. The Mathematical Association of America, 1994
Brassard, G.: Modern cryptology. Springer Verlag, 1988
van Tilborg: An introduction to cryptology. Kluiver Academic Publisher, 1988

---

<table>
<thead>
<tr>
<th>Name of the course:</th>
<th>Cryptographic protocols</th>
</tr>
</thead>
<tbody>
<tr>
<td>Faculty member responsible for the course:</td>
<td>Dr. Attila Kovács, associate professor</td>
</tr>
<tr>
<td>Responsible department:</td>
<td>Faculty of Informatics, Department of Computer Algebra</td>
</tr>
<tr>
<td>Total credits:</td>
<td>4</td>
</tr>
<tr>
<td>Total hours:</td>
<td>4</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Type of the course</th>
<th>lecture</th>
<th>practice</th>
<th>consultation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Hours per week</td>
<td>2</td>
<td>2</td>
<td></td>
</tr>
<tr>
<td>Type of testing</td>
<td>exam</td>
<td>practice</td>
<td></td>
</tr>
</tbody>
</table>

**Topics:**
This course gives an overview of the basic building blocks used to engineer cryptographic protocols, and discusses in details the operation of mainstream cryptographic protocols used in
wired and wireless computer networks. In particular, TLS and IPsec are covered, as well as security protocols in WiFi networks. We also study protocols used in emerging wireless networks, such as wireless sensor networks and RFID systems.

Basic concepts and crypto primitives
Basic concepts and crypto primitives (cont'd)
Block encryption modes
Message authentication and authenticated encryption
Key exchange protocols
Random number generation
Verification of key exchange protocols with ProVerif
Public Key Infrastructures
TLS
WiFi security
IPsec
Security protocols for wireless sensor networks
Secure routing and wormhole detection
RFID security and privacy

Literature:

Name of the course: Cryptography and its applications
Faculty member responsible for the course: Dr. Attila Kovács, associate professor
Responsible department: Faculty of Informatics, Department of Computer Algebra
Total credits: 4
Total hours: 4

<table>
<thead>
<tr>
<th>Type of the course</th>
<th>lecture</th>
<th>practice</th>
<th>consultation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Hours per week</td>
<td>2</td>
<td>2</td>
<td></td>
</tr>
<tr>
<td>Type of testing</td>
<td>exam</td>
<td>practice</td>
<td></td>
</tr>
</tbody>
</table>

Topics:
Paradigms of provable security algorithmic reduction, algorithmic indistinguishability, simulatability.
Universal composability (UC) security framework:
The underlying computational model: PPT interactive Turing machines.
The model of protocol execution.
Protocol emulation. Special approaches: black box adversary, dummy adversary.
Ideal functionality and ideal protocol. Defining the security protocols: emulation of the ideal functionality.
Hybrid protocol.
The UC theorem.
UC with joint state (JUC).
Definition of ideal functionalities for cryptographic tasks.
Examples: UC-realization, composition, technique of hybrid protocols.

**Literature:**

