JAVA

Syllabus

Topic: Basics of Software Development in JAVA

Lecturer: Miloš Kosanović

Course objectives:

The course main objective is to teach students the basics of software development. Other goals are to teach and enable students to think algorithmically, to enable them to analyze and breakdown real-life problem into algorithmic steps, and then implement these steps in some programming language (in this case JAVA). Student will also learn to recognize and use basic and advanced data structures like arrays, strings, multidimensional arrays, hash tables, stack and queues. They will learn to write, compile, build and debug program in IntelliJ IDEA. Prior knowledge is not needed. Special attention will be given to practical work, coding, and the usage of available online services that enable student to learn programming concepts through practical examples and games.

Course outcomes:

The participants will acquire:

- problem solving skills – to analyze real life problem, find and develop algorithmic steps to solve it and then implement these steps in JAVA# programming language
- basic knowledge of programming in JAVA
- experience with developing and debugging software in IntelliJ IDEA
- knowledge about basic and advanced data structures
- ability to use version controls tools like GIT/SVN

Theoretical teaching topics:

1. Algorithms and the elements of programming languages. Program control flow.
2. Basic data types and data structures
3. Modular programming, Functions, recursion
4. Advanced data structures: Strings, Hash tables, Stack and Queue
5. Version control and Git

Practical exercises:

The number of practical work will be 70%. The practical exercises will be implemented in IntelliJ IDEA.

Evaluation:

Students will be evaluated based on:

1. Homework (20%) – There will be 2 homework assignments.
2. Activity (30%) – will be assessed by professor during lectures and will be based on student activity, speed and skill shown during lecture assignments.
3. Final exam (50%)– will consist of ABCD test and practical test. Student will get assignments for practical test that he needs to solve by using a computer.

Student will pass the exam if he has more than 50% of points.

Course duration:

It will last for 40 school classes (36 for lectures and 4 for tests). Lectures will be completed in a block system in one-month time (10 days, 4 lectures per day)

Course content:

Day	Topic	Practical	Homework And Test
1	•Introduction to Computer Science and IT. •Jobs in IT •Computer SW and HW architecture. •Introduction to algorithms.	•Introduction to IntelliJ IDEA. •Cloud storage services and Dropbox.	
2	•Introduction to JAVA. •Arithmetic, logic and relation operators. •Variables and data types. •Variable scope. Local and global variables. •Input and output statements. If statement.	•Coding and debugging a computer program.	
3	•Switch statement. •Algorithms with Loops. •Infinite Loop and break and continue statements. •Introduction to data structures and Arrays.	•Assignment with For and while loops.	
4	•Operations with arrays. •Different array sorting algorithms •Introduction to Multidimensional arrays. •Operations with Multidimensional arrays.	•Assignment with Arrays and Lists.	HW 1
5	•Introduction to Classes, objects and methods. •Introduction to standard library classes. •Data type conversion.	•Type conversion. •Using Math and Date library classes.	
6	•Introduction to modular programming. •Declaration and definition of functions. •Referent data types. •Passing arguments to functions.	•Assignments with functions.	
7	•Recursion. •Array as function parameter.	•Strings Assignments.	HW2

	•Introduction to String data type. •Methods of String class and operations with strings		
8	•Introduction to generic classes. •Advanced data structures. •Hash tables, Linked list, Stack, Queue.	•Hash tables, Stack, Queue	
9	•Introduction to version control. •Introduction to Git.	•Git Assignment.	
10	•Summary of the course. •The analysis of the final test.		Final Test

Topic: Object-oriented programming

Lecturer: Ivica Marković

Course objectives:

The course will introduce concepts and methodology of object oriented programming in Java. Goal of the course is to teach participants how to abstract a problem in an object oriented style and program a solution in Java for that problem. Students will also learn to develop Windows applications by using Java IDE (Integrated Development Environment). Special attention will be given to practical work.

Course outcomes:

Mastering the object oriented programming and skills to independently design solutions to problems using the object-oriented concept and to learn how to develop applications in Java programing language. Students will acquire:

- strong understanding of OOP concepts,
- good understanding of standard Java libraries,
- further develop real life problem solving skills.

Theoretical teaching topics:

- The concepts of object oriented programming. Classes and object.
- Programming in Java.
- Static class members. Inheritance. Polymorphism. Abstract classes and interfaces.
- Exception handling.
- Working with files in Java.
- Windows programming in Java.

Practical exercises:

Share of practical work in total working hours will be 40% to 60%.

Evaluation:

Students will be evaluated based on:

4. Homework (20%) – There will be 2 homework assignments.
5. Activity (30%) – will be assessed by professor during lectures and will be based on student activity, speed and skill shown during lecture assignments.
6. Final exam (50%)– will consist of ABCD test and practical test. Student will get assignments for practical test that he needs to solve by using a computer.

Student will pass the exam if he has more than 50% of points.

Course duration:

It will last for 40 school classes (36 for lectures and 4 for tests). Lectures will be completed in a block system in one-month time (10 days, 4 lectures per day)

Course content:

Day	Topic	Practical	Homework and Testing
1	1. class: Procedural, modular and object-oriented programming techniques. Class definition, attributes and methods. Class member access. Objects creation. 2. class: Reference this, static class members, getter and setter methods.	1. class: Develop object-oriented application in Java IDE. Adding class by using class wizard. Creating class diagram. 2. class: Adding getters/setters and static attributes and methods in previously created application.	
2	1. class: Value and reference types. Enumerations. Types and conversions. 2. class: Pass parameters to methods. Methods with variable number of parameters.	1, 2. class: Java application with several classes and enumerations.	
3	1. class: Constructors. 2. class: Inheritance. Definition of derived class. Constructors of derived classes.	1, 2. class: Application containing classes with constructors, base and derived classes.	
4	1. class: Polymorphism. Methods overriding in Java. 2. class: Abstract methods and abstract classes.	1, 2. class: Application with polymorphism and abstract classes.	

5	1. class: Interfaces in Java. 2. class: Packets and Java standard library.	1, 2. class: Application containing interfaces and using classes from Java standard library.	HW 1
6	1. class: Exception handling: exception throwing, exception catching. Uncaught exceptions. 2. class: Generic methods and classes.	1, 2. class: Application containing generic classes and exceptions.	
7	1. class: Input-output. Working with textual files in Java. 2. class: Working with binary files in Java.	1, 2. class: Application containing reading data from files, and writing data into files.	
8	1. class: Windows forms and windows components in Java. 2. class: Events and listeners in Java.	1, 2. class: Simple Windows applications.	
9	1. class: New controls, their methods and events.	1, 2, 3. class: Windows application with several forms.	HW2
10	Summary of the course.		Final Test

Topic: Introduction to relational databases

Lecturer: Miloš Bogdanović

Course objectives: To achieve understanding of designing and manipulating relational data structures

Course outcomes: Main desired outcomes are**:**

- Ability of participants to independently design relational data structures of moderate complexity.
- Ability of participants to independently develop SQL queries for the web projects of moderate complexity

Theoretical teaching topics: Different techniques and approaches for information modeling. Relational Database Management Systems – main concepts, definition, architecture. Conceptual design of databases – Entity-Relationship (ER) modeling. Logical database design: datatypes, transformation of conceptual to logical database design, Implementation of database by using SQL language, Data constraints, Referential integrity. Physical database design: schema normalization and denormalization, SQL query optimization and indexing, application and security aspects of the database design (access control, data audit), different functions of the RDBMS (views, stored procedures, triggers, transactions).

Data view and manipulation by using SQL: data inserting, data deletion, data updates, viewing data (sorting, viewing data from multiple tables, basic data processing, working with time and date functions, subqueries, aggregate queries). SQL query optimization – continuation.

Practical exercises: Exercising basic information modeling: for the given form, define concepts, their characteristics, their constraints and their relationships. Exercising hierarchical data modeling: for the given website, develop XML schema. Using draw.io service for drawing ER models. Exercising basic understanding of the ER concepts: for the number of given short and longer sentences, identify concepts and relationships and draw basic ER models. Exercising understanding of data and relation properties: for the given narrative text, design ER models. Exercising advanced ER concepts, namely, identificators, complex and multi-valued attributes, weak entities, asociative entities, cardinality and modality, relationships with optional and mandatory entity participation, identifying and non-identifying relationships, recursive relationships: for the given narrative text, design ER models. Exercising transformation of conceptual to logical database design: for the given conceptual ER model, design database schema, by using WWW SQL designer web service. MySQL software installation and customization. SQL exercises: for the given texts and given database designs, develop the set of INSERT, UPDATE and SELECT queries.

Evaluation: The overall grade is composed of the partial assessments of homework (20% of overall grade, total of 3 homework assignments), final exam (50% of overall grade, practical test with the narrative form of requirements which need to be transposed to a logical and physical database design + narrative functional requirements for the given database structure which need to be transposed to SQL queries) and participants engagement during class assignments (30% of overall grade, include trainers personal assessment of the participants skills).

Course duration: The course is 40 classess long (36 for lectures and practical work, 4 for a final exam). The 4 classes blocks will be implemented by day.

Course content:

Day	Topic	Practice	Homework and testing
1	**Class 1.** Client-server architecture DIKTW pyramid: Data, Information, Knowledge, Wisdom Meta-data **Class 2.** Hierarchical models – XML Data model features, data features **Class 3.** Data modeling approaches and tools Relational databases, RDBMS **Class 4.** Process of database design	Define basic functional requirements for the given website idea Identify concepts, their features, constraints and relationships for the given examples Based on the identified functional requirements, design XML schema Summary discussion	
2	**Class 1.** Conceptual, logical and physical design Key elements of the conceptual design Entity types, weak entities **Class 2.** Associative entities, category	Using draw.io web service for drawing ER schemas Design simple ER schemas for the given	

	entities Attributes **Class 3.** ER notation **Class 4.** Cardinality	short and longer sentences – identification of entities, entity types and relationships and their cardinalities Demonstration of conceptual design process on example Independent design of the conceptual model – 4 assignments	
3	**Class 1.** Advanced ER concepts: identificators, complex and multi-valued attributes **Class 2.** Weak entities, associative entities, modality, relationship with optional and mandatory entity participation, identifying and non-identifiying relationships, recursive relationships **Class 3-4.** Exercises	Identify modalities for the given short examples For the previously done assignments, implement advanced ER concepts above – 4 assignments Independent design of ER models of moderate complexity – 3 assignments	Homework – design conceptual model for 2 given narrative descriptions of websites
4	**Class 1.** Transformation of conceptual to logical design Data types (textual, numeric, date and time) **Class 2-3.** WWW SQL Designer online tool **Class 4.** Transformation of subtypes Creating constraints	Working with WWW SQL Designer online tool Demonstration – transformation of conceptual to logical design – reference example Saving, opening and generating SQL code in WWW SQL Designer MySQL software installation and customization	Homework – transform conceptual to logical model for two given models – 2 assignments
5	**Class 1.** Introduction to SQL. Key principles **Class 2.** CREATE TABLE command **Class 3.** Application and security aspects of database design. **Class 4.** INSERT command	Creating databases with the code generated from WWW SQL Designer Inserting data in the created database – demonstration and assignment	

6	**Class 1.** Basic structure of SELECT, UPDATE and DELETE SQL queries Operators **Class 2.** Criteria for data manipulation, using joker characters Sorting **Class 3.** Joining data (INNER, LEFT, RIGHT, FULL) Using aliases DISTINCT clause **Class 4.** Basic data processing	Assignments – design of SELECT, UPDATE and DELETE queries based on the given narrative – functional requirements Assignments – joining data and basic data processing	Homework – for the given sentences (functional requirements) design SELECT, UPDATE and DELETE queries
7	**Class 1.** Basic structure of aggregate queries Aggregate functions: MIN, MAX, SUM, AVG, COUNT **Class 2-3.** GROUP BY, HAVING clauses **Class 4.** SQL date and time functions	Demonstration of examples of aggregate queries Assignments – design of aggregate queries Demonstration of examples of using date and time functions	
8	**Class 1-2.** Functions of RDBM systems Views, Stored procedures, Triggers, Transactions **Class 3.** Indexing **Class 4.** Schema normalization	Demonstration of RDBMS functions Discussion on schema normalization approaches	
9	**Class 1-2.** Pre-exam **Class 3-4.** Discussion and self-evaluation	Participants are given the example of exam. They work on the solutions for the given problems. After the completion, teacher presents the solution, summary discussion takes place, based on which participants are self-evaluated.	Self-evaluation
10	**Class 1-4.** Final exam		Final exam

Topic: Web programming

Lecturer: Nikola Vitković

Course objectives:

The course main objective is to teach students the fundaments of web development in Java. This course covers the most current tools available for developing rich web applications. Students will learn about: networks, fundaments of web, web services and web servers, HTML, CSS, JavaScript and TypeScript, Java for web, Spring framework and other related technologies. Concerning Spring as an enterprise application framework, students will acquire knowledge about Java Annotations, Aspect Oriented Programming (AOP), Java Persistent Api (JPA), Hibernate, Maven, etc. Prior knowledge about web programming is not needed. Students will acquire systematic knowledge about web programming tools, and Spring MVC framework, and they will learn about best practice techniques used in web development. Throughout the course, students will work on small scale web applications, in order to acquire knowledge about web application development processes. As a result of the curse, students will be trained to develop a complete web application using Spring MVC framework.

Course outcomes:

The participants will acquire:

- Knowledge about Networks, Web protocols, Web services, REST API, and other important technologies.
- Front End Development techniques – They will acquire good knowledge of HTML5, and introductory knowledge of CSS, JavaScript and TypeScript.
- Java advance programming techniques.
- Good knowledge of Spring enterprise framework.
- Knowledge about web and application servers. They will learn how to use and configure Tomcat web server.
- High level of Spring MVC framework.
- Advanced SQL knowledge.
- Skills to use ORM software - Hibernate.
- Ability to build and configure project in Maven.
- Skills to implement an appropriate planning strategy for developing websites.
- Skills to create fully functional web application.

Theoretical teaching topics:

6. World Wide Web and networks.
7. Web application (structure and architecture), web services and web servers.
8. Front end technologies and their application (HTML, CSS, JavaScript and TypeScript).
9. J2EE – Basic introduction to Java Enterprise Edition (Web and Application servers).
10. Spring framework.

11. Spring MVC framework (Model – View – Controller, REST Services, Security, etc.).
12. Java and Relation databases (JDBC).
13. Object Relation Mapping technologies. Hibernate example.
14. Project management in Maven.
15. Best practice techniques in web development.

Practical exercises:

The number of practical work will be 40-60%. The practical exercises will be implemented in adequate Integrated Development Environment (IDE).

Evaluation:

Students will be evaluated based on:

7. Homework (20%) – There will be 3 homework assignments.
8. Activity (30%) – will be assessed by professor during lectures and will be based on student activity, speed and skill shown during lecture assignments.
9. Final exam (50%) – will consist of ABCD test and practical test. Student will get assignments for practical test that he needs to solve by using a computer.

Student will pass the exam if he has more than 50% of points.

Course duration:

It will last for 106 school classes (102 for lectures and 4 for tests). Lectures will be completed in a block system in two and a half month time (27 days, 4 lectures per day). Additional lectures can be held in accordance to student's needs and available time.

Course content:

Day	Topic	Practical	Homework and Testing
1.	**Class 1-2.** Introduction to World Wide Web (WWW). Web servers. Web Services. Web application architecture. **Class 3-4.** Introduction to HTML. HTML5 as semantic representation of web content. Basic Structure of HTML.	Creation of the first basic web page.	
2.	**Class 1-2**. HTML text formatting elements. HTML links. **Class 3-4.** HTML Tables.	Creation of the web document width application of learned tags.	

3.	**Class 1-3.** HTML forms. Theory and application. **Class 4.** Validation of form elements trough HTML attributes.	Creating web pages with forms. Basic validation	
4.	**Class 1.** Introduction to CSS. **Class 2.** CSS structure and integration. **Class 3-4.** CSS elements (Implementation).	Creation of simple web app, using prepared boilerplate.	HW1
5.	**Class 1.** Introduction to JavaScript (JS). JS Versions and compatibility. ECMA standard. **Class 2-4.** Variables and data types.	Analysis of JS integration into HTML page.	
6.	JS program elements introductory level: **Class 1-2.** Loops, conditional statements **Class 3.** Objects. **Class 4.** Exceptions	Creation of first JS example width error handling.	
7.	**Class 1-4.** Exercise: Creation of small scale web site with learned technologies and techniques.	Web site development.	
8.	**Class 1.** Introduction to TypeScript (TS). History. Differences between JS and TS. **Class 2.** Application of TS. **Class 3-4.** TS compilation. Tools.	Analysis of provided example.	
9.	TS program elements introductory level: **Class 1-2.** Loops, conditional statements **Class 3.** Objects. **Class 4.** Exceptions	Students will create same example as with JS, but they will use TS.	
10.	**Class 1-4.** Recapitulation of learned material.	Student's questions. Knowledge assessment.	Self – evaluation HW2
11.	**Class 1-2.** Introduction to J2EE. Web and Application Servers. **Class 3-4.** Tomcat web server.	Installing Tomcat web server	
12.	**Class 1.** Introduction to Spring. **Class 2.** Dependency Injection (DI).	Analysis of examples with DI and AOP implemented.	

	Class 3. Inversion Of Control (IoC) **Class 4.** Aspect Oriented Programming (AOP).		
13.	**Class 1-4.** Installation of required components. Integration of Spring libraries and dependent libraries. Creation of first Spring application.	Creation of first Spring application.	
14.	**Class 1.** Java Annotations and their implementation in Spring. **Class 2-4.** Spring DI with annotations and auto wiring	Practical example	
15.	**Class 1.** Java Beans. **Class 3-4.** Definition of Spring configuration with configuration file.	Practical example	
16.	**Class 1-4.** Definition of Spring configuration with Java Code	Practical example	
17.	**Class 1.** Introduction to Spring MVC. Integration with IDE. **Class 2-4.** Controllers and Views.	Creation of simple Spring MVC application.	
18.	**Class 1-2.** Data binding **Class 2-4.** Form Validation and Security	Implementation of form validation	Self-evaluation
19.	**Class 1.** Implementing Session **Class 2-4.** Exercise: Creation of Spring MVC web application	Creation of Spring MVC web application	
20.	**Class 1.** Java and Relation database systems **Class 2.** JDBC basics **Class 3-4.** Connection with database by using JDBC.	Database connection and manipulation - Practical example.	
21.	**Class 1.** Introduction to Object Relation Mapping (ORM). **Class 2-4.** Hibernate and database connection and manipulation	Database example with Hibernate	HW3
22.	**Class 1.** Hibernate Annotations. **Class 2.** Hibernate advance mapping. **Class 3-4.** Integration of Hibernate and Spring MVC framework.	Practical Example. Hibernate and Spring MVC	

23.	**Class 1.** Maven introductory course. **Class 2-3.** Maven environment and POM (Project Object model). **Class 4.** Definition of project in POM.	Creation of Maven configuration file and MVC integration.	
24.	**Class 1-4.** Exercise: Creation of web application, part I Application of acquired techniques and technologies.	Creation of web application	Practical assessment of acquired knowledge
25.	**Class 1-4.** Exercise: Creation of web application, part II Application of acquired techniques and technologies	Creation of web application	
26	**Class 1-4.** Recapitulation / Preparation for final exam	Student's questions. Knowledge assessment.	Self-evaluation
27.	Final exam		Final Exam

Topic: Methodologies and tools for software development

Lecturer: Miloš Kosanović

Course objectives:

The course will introduce the concepts and methodologies of the agile software development, life-cycle, development processes and Project Management frameworks.

Course outcomes:

The participants will understand all processes and activities in software development and will be able to participate in all phases of software project cycle.

Theoretical teaching topics:

1. Methodologies overview (frameworks and approaches). Software development life cycle and software development process. User interface design.
2. Requirements Engineering. Agile methodology (SCRUM).
3. Task planning. Task estimation and implementation.
4. Architecture design. UML language. Testing and documenting.
5. Deployment and Maintenance. Project Management frameworks.

Practical exercises:

20% to 50% of working hours. The practical exercises will include scrum meeting simulation, sprint planning meeting simulation, use case analysis and discussion, software tools for fast wireframe and UI development, overview and demonstration of tools for project management and issue tracking.

Course content: Lecturer will need to complete a course from the selected field with its own course curriculum.

Course duration: The course will last for **24 school classes (20 for lectures and 4 for tests)**. Lectures will be completed in a block system (6 days with 4 classes per day).

Curriculum

The first day should be the general introduction into the course topics and the software development cycle in general. The content and the learning pace will depend on the candidate's previous knowledge and experience. The remaining 5 days will follow the rule: 2 school classes of lecture, 2 school classes of practical work. The goal of the course will be to follow the development of one real life project example from its inception to the first production release of the product. It will be requested from the candidates to create a team for the project, analyze the requirements, specify high level requirements, create the tasks or user stories, describe the testing and quality procedures and describe their solution to the real life problems.

Grading

1. Homework assignments will be graded with 20%
2. Lecturer will grade each student based on general impression with 30%.
3. Final ABCD test and project (which will contain requirements specification document) will be graded with 50%

The student will pass each test if he has correctly answered more than 50% of the questions. The student will pass the course if he has more than 50% of points.

Tests

The test questions and the topic of the project assignment will not be disclosed due to confidentiality reasons.

Course content:

Day	**Topic**	**Practical**	**Homework And Test**

1	•Introduction, Course overview, Methodologies overview (framework and approaches) •SDLC - Software Development Life Cycle. •Waterfall method, V model •Incremental model, Iterative model (RUP, EUP), Prototype model, Spiral model •UI Design - Wire-Frames and Mock-ups	•Create wireframes (use Balsamiq)	HW 1
2	•Requirements Analysis & Specification. Functional and non-functional requirements •User stories •Introduction to project assigment •Agile methodology (SCRUM, Kanban, XP), Comparison between standard and agile approach. •Team roles in agile development •Test Driven Development TDD, Behaviour Driven Development BDD •Continious Integration •Agile software development process.	•Test project explanation and proposal •SCRUM meeting simulation, • Case stady I – Requirements meeting	
3	•SCRUM •KANBAN •Planning and estimating tasks •The Vision or Long term planning, Scope change. Time constraints and other problems.	•WBS – Work breakdown structure •Sprint planning meeting simulaton	HW 2
4	•Tehnical design (specification). UML diagrams. Use Case, Activity and class diagram, Sequence diagrams, •System design, DataBase design •Coding - good coding practice, •Software Testing and Quality Assurance. •Software Maintenance and end user support	•UML diagrams (starUML, gliphy)) •Questions and discussion •Writing unit tests.	
5	•Project Managment •Project Managmenet frameworks overview (JIRA, redmine, microsoft project, gunter) •Other tools for project management, development and issue tracking, Reporting bugs	•Introduction to some Project management tool •Write tasks in some project management tool	
6	•Coding - good coding practice. •Software Maintenance and end user support •Documenting projects		Final Test