ERASMUS+ INCOMING STUDENTS
ACADEMIC OFFER - COURSES TAUGHT IN ENGLISH
COMPUTER SCIENCE
SPRING 2024/2025

1st year, 2nd semester

COMPUTER SCIENCE BACHELOR

Discrete mathematics – lecture

The purpose of the course is to familiarize the student with combinatorics along with the necessary elements of algebra and number theory. Introduction to graph theory and its applications.

Aims of the Subject

- The aim of the subject is to introduce students to combinatorics, including essential elements of algebra and number theory. It also provides an introduction to graph theory and its applications.

3 ECTS

Learning Outcomes

In Terms of Knowledge

- Defines basic theorems of number theory.
- Understands and applies the concepts of recursion and induction, and is able to use reasoning rules and conduct simple proofs.
- Defines fundamental theorems of combinatorics.
- Formulates and simplifies practical problems using mathematical language.

In Terms of Skills

- Solves recurrence equations.

- Utilizes the principle of mathematical induction.
- Performs calculations in matrix arithmetic.
- Formulates and simplifies practical problems using mathematical language.

In Terms of Social Competencies

- Demonstrates creative and inventive thinking.
- Understands the need and recognizes the opportunities for continuous improvement of personal competencies.

Program Content

1. Introductory concepts and relations.
2. Introduction to graph theory.
3. Graphs and trees.
4. Induction and recursion.
5. Elements of combinatorics.
6. Basic counting methods.

Teaching Methods

- Lecture (partially using multimedia tools).
- Didactic discussion.
- Classical problem-based method.
- Project-based method.

Education verification method: exam, activity during classes

Discrete mathematics – project

The purpose of the course is to familiarize the student with combinatorics along with the necessary elements of algebra and number theory. Introduction to graph theory and its applications.

Aims of the Subject

- The aim of the subject is to introduce students to combinatorics, including essential elements of algebra and number theory. It also provides an introduction to graph theory and its applications.

4 ECTS

Learning Outcomes

In Terms of Knowledge

- Defines basic theorems of number theory.
- Understands and applies the concepts of recursion and induction, and is able

to use reasoning rules and conduct simple proofs.
- Defines fundamental theorems of combinatorics.
- Formulates and simplifies practical problems using mathematical language.

In Terms of Skills

- Solves recurrence equations.
- Utilizes the principle of mathematical induction.
- Performs calculations in matrix arithmetic.
- Formulates and simplifies practical problems using mathematical language.

In Terms of Social Competencies

- Demonstrates creative and inventive thinking.
- Understands the need and recognizes the opportunities for continuous improvement of personal competencies.

Program Content

1. Introductory concepts and relations.
2. Introduction to graph theory.
3. Graphs and trees.
4. Induction and recursion.
5. Elements of combinatorics.
6. Basic counting methods.

Teaching Methods

- Lecture (partially using multimedia tools).
- Didactic discussion.
- Classical problem-based method.
- Project-based method.

Education verification method: exam, activity during classes

ZDW: Operating Systems / Computer hardware and application management systems- tutorials

Aims of the Subject:

- Understanding fundamental concepts related to operating systems.

- Familiarizing students with the role of the operating system in the functioning of a computer system.
- Introducing students to the most important operating systems.

Learning Outcomes:

In terms of knowledge:

- Defines fundamental concepts related to operating systems.
- Can explain the role of the operating system in the functioning of a computer system.
- Chooses appropriate system solutions for specified goals.

In terms of skills:

- Acquires practical skills in managing various operating systems.
- Gains practical skills in managing operational memory.
- Develops practical skills in administering file systems, virtual memory, and dynamic libraries.

In terms of social competences:

- Demonstrates the ability for continuous learning, improving, and enhancing professional, personal, and social competencies.

Program Content:

1. Structure and functions of an operating system.
2. Concept of processes and threads. Processor time allocation.
3. Management of operational memory allocation.
4. Virtual memory mechanisms.
5. File systems.
6. Input-output system handling.
7. Working in networks.
8. User accounts in operating systems. Access control.
9. Mechanism of dynamic libraries.
10. System services/daemons.

Teaching Methods:

- Lecture
- Classical problem-based method
- Project-based method

Education verification method: exam, activity during classes

ZDW: Operating Systems / Computer hardware and application management systems- lecture

Aims of the Subject:

- Understanding fundamental concepts related to operating systems.
- Familiarizing students with the role of the operating system in the

functioning of a computer system.
- Introducing students to the most important operating systems.

Learning Outcomes:

In terms of knowledge:

- Defines fundamental concepts related to operating systems.
- Can explain the role of the operating system in the functioning of a computer system.
- Chooses appropriate system solutions for specified goals.

In terms of skills:

- Acquires practical skills in managing various operating systems.
- Gains practical skills in managing operational memory.
- Develops practical skills in administering file systems, virtual memory, and dynamic libraries.

In terms of social competencies:

- Demonstrates the ability for continuous learning, improving, and enhancing professional, personal, and social competencies.

Program Content:

1. Structure and functions of an operating system.
2. Concept of processes and threads. Processor time allocation.
3. Management of operational memory allocation.
4. Virtual memory mechanisms.
5. File systems.
6. Input-output system handling.
7. Working in networks.
8. User accounts in operating systems. Access control.
9. Mechanism of dynamic libraries.
10. System services/daemons.

Teaching Methods:

- Lecture
- Classical problem-based method
- Project-based method

Education verification method: exam, activity during classes

Aims of the Subject:

- Understanding fundamental concepts related to operating systems.
- Familiarizing students with the role of the operating system in the functioning of a computer system.
- Introducing students to the most important operating systems.

Learning Outcomes:

In terms of knowledge:

- Defines fundamental concepts related to operating systems.
- Can explain the role of the operating system in the functioning of a computer system.
- Chooses appropriate system solutions for specified goals.

In terms of skills:

- Acquires practical skills in managing various operating systems.
- Gains practical skills in managing operational memory.
- Develops practical skills in administering file systems, virtual memory, and dynamic libraries.

In terms of social competencies:

- Demonstrates the ability for continuous learning, improving, and enhancing professional, personal, and social competencies.

Program Content:

1. Structure and functions of an operating system.
2. Concept of processes and threads. Processor time allocation.
3. Management of operational memory allocation.
4. Virtual memory mechanisms.
5. File systems.
6. Input-output system handling.
7. Working in networks.
8. User accounts in operating systems. Access control.
9. Mechanism of dynamic libraries.
10. System services/daemons.

Teaching Methods:

- Lecture
- Classical problem-based method
- Project-based method

Education verification method: exam, activity during classes

Basics of programming 2-tutorials

Aims of the Subject:

- To impart knowledge in the subject area.
- To enhance the ability to design computer programs and create efficient source code.

Learning Outcomes:

In terms of knowledge:

- Acquires knowledge in the area covered by the subject, including the construction and development of computer programs.
- Understands and can explain methods for analyzing the subject matter.
- Understands and can explain selected topics related to modeling methods.

In terms of skills:

- Has achieved and refined the ability to design efficient source code for computer programs.
- Can apply knowledge to solve complex and non-standard IT problems.
- Can use available literature to formulate and solve IT problems.

In terms of social competencies:

- Understands the need and opportunities for continuous improvement of professional, personal, and social qualifications.
- Recognizes the need and knows how to continuously enhance personal competencies.
- Is aware of and understands the non-technical aspects and consequences of the work of a computer engineer.

Program Content:

1. Definition and typology of programming languages, including an overview and examples. General structure and methods of code translation for computer programs.
2. Program construction, covering data types such as basic and complex types, constants, variables, arrays, as well as record and file types.
3. File operations including opening and closing files, writing to disk, arithmetic and logical operators, assignment and conditional statements, and macro substitution.
4. Refinement of iterative operations in program construction, with elements of object-oriented programming, including classes, objects, properties, and methods for handling objects.

Teaching Methods:

- Lectures

- Project-based methods
- Auditory/laboratory exercises

Education verification method: exam, activity during classes

Basics of programming 2- project

Aims of the Subject:

- To impart knowledge in the subject area.
- To enhance the ability to design computer programs and create efficient source code.

Learning Outcomes:

In terms of knowledge:

- Acquires knowledge in the area covered by the subject, including the construction and development of computer programs.
- Understands and can explain methods for analyzing the subject matter.
- Understands and can explain selected topics related to modeling methods.

In terms of skills:

- Has achieved and refined the ability to design efficient source code for computer programs.
- Can apply knowledge to solve complex and non-standard IT problems.
- Can use available literature to formulate and solve IT problems.

In terms of social competences:

- Understands the need and opportunities for continuous improvement of professional, personal, and social qualifications.
- Recognizes the need and knows how to continuously enhance personal competencies.
- Is aware of and understands the non-technical aspects and consequences of the work of a computer engineer.

Program Content:

1. Definition and typology of programming languages, including an overview and examples. General structure and methods of code translation for computer programs.
2. Program construction, covering data types such as basic and complex types, constants, variables, arrays, as well as record and file types.
3. File operations including opening and closing files, writing to disk, arithmetic and logical operators, assignment and conditional statements, and macro substitution.
4. Refinement of iterative operations in program construction, with elements of

object-oriented programming, including classes, objects, properties, and methods for handling objects.

Teaching Methods:

- Lectures
- Project-based methods
- Auditory/laboratory exercises

Education verification method: exam, activity during classes

Basics of programming 2- lecture

Aims of the Subject:

- To impart knowledge in the subject area.
- To enhance the ability to design computer programs and create efficient source code.

Learning Outcomes:

In terms of knowledge:

- Acquires knowledge in the area covered by the subject, including the construction and development of computer programs.
- Understands and can explain methods for analyzing the subject matter.
- Understands and can explain selected topics related to modeling methods.

In terms of skills:

- Has achieved and refined the ability to design efficient source code for computer programs.
- Can apply knowledge to solve complex and non-standard IT problems.
- Can use available literature to formulate and solve IT problems.

In terms of social competencies:

- Understands the need and opportunities for continuous improvement of professional, personal, and social qualifications.
- Recognizes the need and knows how to continuously enhance personal competencies.
- Is aware of and understands the non-technical aspects and consequences of the work of a computer engineer.

Program Content:

1. Definition and typology of programming languages, including an overview and examples. General structure and methods of code translation for computer programs.
2. Program construction, covering data types such as basic and complex types,

constants, variables, arrays, as well as record and file types.

3. File operations including opening and closing files, writing to disk, arithmetic and logical operators, assignment and conditional statements, and macro substitution.
4. Refinement of iterative operations in program construction, with elements of object-oriented programming, including classes, objects, properties, and methods for handling objects.

Teaching Methods:

- Lectures
- Project-based methods
- Auditory/laboratory exercises

Education verification method: exam, activity during classes

Numerical methods in computer science – lecture

The purpose of the class is to acquire basic knowledge of the effect of floating point arithmetic on the result of numerical calculations, to learn numerically correct algorithms for basic tasks of mathematical analysis and linear algebra.

Aims of the Subject:

- To acquire basic knowledge regarding the impact of floating-point arithmetic on numerical computation results.
- To learn numerically accurate algorithms for fundamental tasks in mathematical analysis and linear algebra.

Learning Outcomes:

2 ECTS

In terms of knowledge:

- Understands and can explain algorithmic methods for solving optimization problems.
- Knows and comprehends basic methods for formulating mathematical models for practical and theoretical optimization problems.
- Understands the principles of formulating and classifying optimization problems.

In terms of skills:

- Can select appropriate numerical methods for solving optimization problems.
- Can apply knowledge to solve complex and non-standard optimization problems.

- Can utilize available literature to formulate and solve IT problems.

In terms of social competencies:

- Understands the need and opportunities for continuous improvement of professional, personal, and social qualifications.
- Recognizes the need and knows how to continuously enhance personal competencies.
- Can engage in professional discussions with users of numerical methods in practice.

Program Content:

1. Solving systems of linear equations, including Gaussian elimination (review), LU decomposition (Doolittle's algorithm), Cholesky decomposition, and issues with numerically ill-conditioned matrices.
2. Techniques for sparse matrices, methods of representing sparse matrices, optimal ordering of equations, Minimum Degree method, elimination trees, supernodes, factorization, and interior-point methods.
3. Iterative methods for solving systems of equations, including classic methods (Jacobi method, Gauss-Seidel method, SOR method), and projection methods (conjugate gradients method, Arnoldi method, GMRES method).
4. Mathematical formulation of optimization problems, specific and general optimization criteria, and objective functions. Principles of formulating and classifying optimization problems, practical and mathematical formulations, and examples of optimization problems.
5. Methods for nonlinear static optimization problems, optimality conditions for unconstrained and equality-constrained problems, optimality conditions for general nonlinear programming problems (Karush-Kuhn-Tucker conditions), and standard form of nonlinear programming problems.
6. Gradient-free methods: stochastic overview, Nelder-Mead algorithm, orthogonal direction method with optimal step length selection, Hooke-Jeeves method, Powell method, and Rosenbrock method.
7. Gradient methods: steepest descent method, conjugate gradient method, and variable metric methods (Davidon-Fletcher-Powell, Broyden-Fletcher-Goldfarb-Shanno).
8. Methods for optimizing multidimensional nonlinear problems with constraints, including gradient-based methods with backtracking, projected gradient methods, methods with penalty functions, and sequential quadratic programming methods.

Teaching Methods:

- Lectures
- Project-based methods
- Classical problem-based methods

Education verification method: exam, activity during classes

Numerical methods in computer science – tutorials

The purpose of the class is to acquire basic knowledge of the effect of floating point arithmetic on the result of numerical calculations, to learn numerically correct algorithms for basic tasks of mathematical analysis and linear algebra.

Aims of the Subject:

- To acquire basic knowledge regarding the impact of floating-point arithmetic on numerical computation results.
- To learn numerically accurate algorithms for fundamental tasks in mathematical analysis and linear algebra.

Learning Outcomes:

In terms of knowledge:

- Understands and can explain algorithmic methods for solving optimization problems.
- Knows and comprehends basic methods for formulating mathematical models for practical and theoretical optimization problems.
- Understands the principles of formulating and classifying optimization problems.

2 ECTS

In terms of skills:

- Can select appropriate numerical methods for solving optimization problems.
- Can apply knowledge to solve complex and non-standard optimization problems.
- Can utilize available literature to formulate and solve IT problems.

In terms of social competencies:

- Understands the need and opportunities for continuous improvement of professional, personal, and social qualifications.
- Recognizes the need and knows how to continuously enhance personal competencies.
- Can engage in professional discussions with users of numerical methods in practice.

Program Content:

1. Solving systems of linear equations, including Gaussian elimination

(review), LU decomposition (Doolittle's algorithm), Cholesky decomposition, and issues with numerically ill-conditioned matrices.

2. Techniques for sparse matrices, methods of representing sparse matrices, optimal ordering of equations, Minimum Degree method, elimination trees, supernodes, factorization, and interior-point methods.
3. Iterative methods for solving systems of equations, including classic methods (Jacobi method, Gauss-Seidel method, SOR method), and projection methods (conjugate gradients method, Arnoldi method, GMRES method).
4. Mathematical formulation of optimization problems, specific and general optimization criteria, and objective functions. Principles of formulating and classifying optimization problems, practical and mathematical formulations, and examples of optimization problems.
5. Methods for nonlinear static optimization problems, optimality conditions for unconstrained and equality-constrained problems, optimality conditions for general nonlinear programming problems (Karush-Kuhn-Tucker conditions), and standard form of nonlinear programming problems.
6. Gradient-free methods: stochastic overview, Nelder-Mead algorithm, orthogonal direction method with optimal step length selection, Hooke-Jeeves method, Powell method, and Rosenbrock method.
7. Gradient methods: steepest descent method, conjugate gradient method, and variable metric methods (Davidon-Fletcher-Powell, Broyden-Fletcher-Goldfarb-Shanno).
8. Methods for optimizing multidimensional nonlinear problems with constraints, including gradient-based methods with backtracking, projected gradient methods, methods with penalty functions, and sequential quadratic programming methods.

Teaching Methods:

- Lectures
- Project-based methods
- Classical problem-based methods

Education verification method: exam, activity during classes

2 ECTS

Numerical methods in computer science - project

The purpose of the class is to acquire basic knowledge of the effect of floating point arithmetic on the result of numerical calculations, to learn numerically correct algorithms for basic tasks of mathematical analysis and linear algebra.

Aims of the Subject:

- To acquire basic knowledge regarding the impact of floating-point arithmetic on numerical computation results.
- To learn numerically accurate algorithms for fundamental tasks in mathematical analysis and linear algebra.

Learning Outcomes:

In terms of knowledge:

- Understands and can explain algorithmic methods for solving optimization problems.
- Knows and comprehends basic methods for formulating mathematical models for practical and theoretical optimization problems.
- Understands the principles of formulating and classifying optimization problems.

In terms of skills:

- Can select appropriate numerical methods for solving optimization problems.
- Can apply knowledge to solve complex and non-standard optimization problems.
- Can utilize available literature to formulate and solve IT problems.

In terms of social competences:

- Understands the need and opportunities for continuous improvement of professional, personal, and social qualifications.
- Recognizes the need and knows how to continuously enhance personal competencies.
- Can engage in professional discussions with users of numerical methods in practice.

Program Content:

1. Solving systems of linear equations, including Gaussian elimination (review), LU decomposition (Doolittle's algorithm), Cholesky decomposition, and issues with numerically ill-conditioned matrices.
2. Techniques for sparse matrices, methods of representing sparse matrices, optimal ordering of equations, Minimum Degree method, elimination trees, supernodes, factorization, and interior-point methods.
3. Iterative methods for solving systems of equations, including classic methods (Jacobi method, Gauss-Seidel method, SOR method), and projection methods (conjugate gradients method, Arnoldi method, GMRES method).
4. Mathematical formulation of optimization problems, specific and general optimization criteria, and objective functions. Principles of formulating and classifying optimization problems, practical and mathematical formulations, and examples of optimization problems.
5. Methods for nonlinear static optimization problems, optimality conditions for unconstrained and equality-constrained problems, optimality conditions for general nonlinear programming problems (Karush-Kuhn-Tucker conditions), and standard form of nonlinear programming problems.
6. Gradient-free methods: stochastic overview, Nelder-Mead algorithm, orthogonal direction method with optimal step length selection, Hooke-

Jeeves method, Powell method, and Rosenbrock method.

7. Gradient methods: steepest descent method, conjugate gradient method, and variable metric methods (Davidon-Fletcher-Powell, Broyden-Fletcher-Goldfarb-Shanno).
8. Methods for optimizing multidimensional nonlinear problems with constraints, including gradient-based methods with backtracking, projected gradient methods, methods with penalty functions, and sequential quadratic programming methods.

Teaching Methods:

- Lectures
- Project-based methods
- Classical problem-based methods

Education verification method: exam, activity during classes

Protection of intellectual property

Aims of the Subject: 1

- To acquire knowledge about intellectual property and its protection in Poland and globally.

Learning Outcomes:

In terms of knowledge:

- Understands and can explain concepts related to intellectual property, including copyright.
- Knows the typical relationships between intellectual property protection and fair competition, innovation, and economic growth.
- Understands the principles of intellectual property protection.
- Can distinguish between moral rights and economic rights in copyright.

In terms of skills:

- Can accurately define a work and other intellectual property subjects in legal and economic terms.
- Can evaluate which works are not protected by copyright and justify why.
- Can select information and statistical data to analyze the economic impact of intellectual property.

In terms of social competencies:

- Is aware of the social and economic importance of intellectual property protection.
- Acts professionally with respect for intellectual property, including creating scholarly texts and simple information without infringing copyright.

Program Content:

1. The historical development of intangible asset protection.
2. International and national aspects of intellectual property protection.
3. The origins and current status of copyright and related rights.
4. The relationship between intellectual property protection and competition policy, combating unemployment, innovation, and economic growth.
5. The subject matter and parties involved in copyright law - definitions.
6. Moral rights of authors concerning protected works.
7. The concept and catalogue of economic rights and exploitation fields of a work. Selected issues related to licensing.
8. Forms of infringement of moral and economic copyright rights, including plagiarism, piracy, and databases. The role of collective rights management organizations.
9. The concept and principles of fair use for private and public purposes. Library and school rights. The right of quotation.
10. Special protection for computer programs, images, and correspondence.
11. Protection of inventions, trademarks, and industrial designs. Community trademarks.
12. Civil and criminal liability principles for intellectual property rights violations.

Teaching Methods:

- Conversational lectures
- Case studies

Education verification method: exam, activity during classes

Basics of creativity

Aims of the Subject

- Acquaint students with concepts related to creativity.
- Equip students with knowledge of deliberate creativity.
- Introduce students to the subjective concept of human nature.
- Develop fluency, flexibility, and originality of thinking.
- Enhance the ability to perceive and assign new meanings to reality.
- Prepare students for creative problem-solving.

In Terms of Knowledge

- Identifies differences between heteronomous and reductionist views versus the subjective view of human nature.
- Distinguishes between creative and standard behaviors.
- Explains concepts related to the development of a creative person.
- Defines creative, deliberate, and standard actions.
- Describes the evolution of the meaning and scope of the concept of

creativity.

- Identifies what fluency, flexibility, and originality of thinking are.
- Explains selected methods of creative problem-solving.
- Describes the problem-solving approach according to the techne method.
- Discusses the advantages and disadvantages of using the techne method for problem-solving.

In Terms of Skills

- Modifies one's perception of reality.
- Connects different ideas, concepts, and thoughts.
- Argues one's position and viewpoints.
- Demonstrates readiness to break mental and operational patterns.
- Applies selected methods of creative problem-solving.

In Terms of Social Competencies

- Maintains autonomy in thinking and actions.
- Organizes personal actions in an innovative way.
- Shows flexibility in thinking and actions.

Program Content

1. History of the concept of creativity.
2. Understanding the concept of creativity – various concepts of creativity.
3. The techne method and scientism.
4. Heuristics – thinking through analogy, metaphor, and abstraction.
5. Selected methods of creative problem-solving.
6. Subjective view of human nature.
7. Conscious development of creative dispositions.

Teaching Methods

- Didactic discussion
- Brainstorming
- Workshop method

Education verification method: exam, activity during classes

**COMPUTER SCIENCE
BACHELOR**

ZDW: Programming of web applications / Framework for web applications – lecture

The aim of the study transferring knowledge in the field of the subject. The students will learn the ability to program and design web applications - run on web pages /The aim of the course is to gain knowledge about the use of Frameworks. The students will be acquainted with the knowledge of the .NET Framework and other technologies, how to create applications using Ruby on Rails Framework, creating scripts, functional programming, creating systems in the cloud.

Aims of the Subject

- Providing knowledge in the subject area.
- Achieving the ability to program and design web applications that run on web pages.

Learning Outcomes

- **Knowledge**
  - Acquires knowledge in the subject area, including the ability to design and program web applications.
  - Knows programming languages and understands which language to use for solving selected IT problems.
  - Knows and understands selected issues in the field, including methods of modeling.
- **Skills**
  - Can apply appropriate programming languages and development tools for building web applications and web services.
  - Can use knowledge to solve complex and unconventional IT problems.
  - Can utilize available literature to formulate and solve IT problems.
- **Social Competencies**
  - Understands the need for and knows the possibilities of continuous professional, personal, and social development.
  - Understands the need for and knows the possibilities of continuous self-improvement.
  - Is aware of the importance and understands the non-technical aspects and consequences of the work of an IT engineer.

Program Content

1. Overview and development of internet programming technologies: HTML,

XHTML, Java, PHP, etc.
2. Development of HTML as a tool for building web applications. Characteristics of PHP technology – its origins, development, language elements, and applications.
3. Characteristics of the Common Gateway Interface (CGI) model.
4. Servlet and applet applications. The three-tier model of web application operation in software engineering. Data access layer in the model.
5. The significance of object-oriented (OOA/D) approach to building web applications. COM/CORBA models – Component Object Model/Common Object Request Broker Architecture.
6. Database management technologies in the web. Development of SQL as a tool for developing web applications for database management.

Teaching Methods

- Lectures
- Project-based methods
- Laboratory exercises
- Seminars / didactic discussions

Education verification method: exam, activity during classes

ZDW: Programming of web applications / Framework for web applications – workshops

The aim of the study transferring knowledge in the field of the subject. The students will learn the ability to program and design web applications - run on web pages /The aim of the course is to gain knowledge about the use of Frameworks. The students will be acquainted with the knowledge of the .NET Framework and other technologies, how to create applications using Ruby on Rails Framework, creating scripts, functional programming, creating systems in the cloud.

Aims of the Subject

- Providing knowledge in the subject area.
- Achieving the ability to program and design web applications that run on web pages.

Learning Outcomes

- **Knowledge**
  - Acquires knowledge in the subject area, including the ability to design and program web applications.
  - Knows programming languages and understands which language to use for solving selected IT problems.
  - Knows and understands selected issues in the field, including methods of modeling.
- **Skills**
  - Can apply appropriate programming languages and development tools

    for building web applications and web services.
    - Can use knowledge to solve complex and unconventional IT problems.
    - Can utilize available literature to formulate and solve IT problems.
- **Social Competencies**
    - Understands the need for and knows the possibilities of continuous professional, personal, and social development.
    - Understands the need for and knows the possibilities of continuous self-improvement.
    - Is aware of the importance and understands the non-technical aspects and consequences of the work of an IT engineer.

Program Content

1. Overview and development of internet programming technologies: HTML, XHTML, Java, PHP, etc.
2. Development of HTML as a tool for building web applications. Characteristics of PHP technology – its origins, development, language elements, and applications.
3. Characteristics of the Common Gateway Interface (CGI) model.
4. Servlet and applet applications. The three-tier model of web application operation in software engineering. Data access layer in the model.
5. The significance of object-oriented (OOA/D) approach to building web applications. COM/CORBA models – Component Object Model/Common Object Request Broker Architecture.
6. Database management technologies in the web. Development of SQL as a tool for developing web applications for database management.

Teaching Methods

- Lectures
- Project-based methods
- Laboratory exercises
- Seminars / didactic discussions

Education verification method: exam, activity during classes

ZDW: Programming of web applications / Framework for web applications – project The aim of the study transferring knowledge in the field of the subject. The students will learn the ability to program and design web applications - run on web pages /The aim of the course is to gain knowledge about the use of Frameworks. The students will be acquainted with the knowledge of the .NET Framework and other technologies, how to create applications using Ruby on Rails Framework, creating scripts, functional programming, creating systems in the cloud.	**2**

Aims of the Subject

- Providing knowledge in the subject area.
- Achieving the ability to program and design web applications that run on web pages.

Learning Outcomes

Knowledge

- Acquires knowledge in the subject area, including the ability to design and program web applications.
- Knows programming languages and understands which language to use for solving selected IT problems.
- Knows and understands selected issues in the field, including methods of modeling.

Skills

- Can apply appropriate programming languages and development tools for building web applications and web services.
- Can use knowledge to solve complex and unconventional IT problems.
- Can utilize available literature to formulate and solve IT problems.

Social Competencies

- Understands the need for and knows the possibilities of continuous professional, personal, and social development.
- Understands the need for and knows the possibilities of continuous self-improvement.
- Is aware of the importance and understands the non-technical aspects and consequences of the work of an IT engineer.

Program Content

1. Overview and development of internet programming technologies: HTML, XHTML, Java, PHP, etc.
2. Development of HTML as a tool for building web applications. Characteristics of PHP technology – its origins, development, language elements, and applications.
3. Characteristics of the Common Gateway Interface (CGI) model.
4. Servlet and applet applications. The three-tier model of web application operation in software engineering. Data access layer in the model.
5. The significance of object-oriented (OOA/D) approach to building web applications. COM/CORBA models – Component Object Model/Common Object Request Broker Architecture.
6. Database management technologies in the web. Development of SQL as a

tool for developing web applications for database management.

Teaching Methods

- Lectures
- Project-based methods
- Laboratory exercises
- Seminars / didactic discussions

Education verification method: exam, activity during classes

Object oriented programming 2 – lecture

The aim of the course is to transfer knowledge in the field of the subject. The students will improve the ability to design computer programs in the OOP - Object Oriented Programming paradigm, creating effective source code and using the advantages of the object-oriented approach.

Aims of the Subject

- Ability to design a special-purpose information system – an intelligent information system.
- Familiarize students with knowledge related to embedded systems and other technologies, script creation, functional programming, and cloud-based systems.

Learning Outcomes

Knowledge

- Understands information systems and the process of information management within an organization.
- Identifies tools dedicated to data design, collection, and exploration.
- Defines the benefits of using databases in engineering and business practice.
- Formulates data models that describe objects and processes occurring in practice.
- Applies appropriate techniques for implementing data models.

Skills

- Designs databases and their structures.
- Implements and designs databases.
- Creates data integrity mechanisms.
- Writes queries for databases.
- Manipulates data.

Social Competencies

- Approaches data model building, database design, implementation, and exploration creatively.
- Actively utilizes databases in engineering and business practice.
- Focuses on the effective use of database design and exploration tools.
- Is aware of the limitations of data models and their implementation.

Program Content

1. General characteristics of special-purpose systems. Characteristics of selected intelligent embedded systems. Real-life examples.
2. Designing an information system in conjunction with a specialized embedded component.
3. System installation and analysis of potential implementation errors.

Teaching Methods

- Laboratory exercises
- Workshop method
- Didactic discussions
- Project-based method
- Demonstrations

Education verification method: exam, activity during classes

Object oriented programming 2 – workshops

The aim of the course is to transfer knowledge in the field of the subject. The students will improve the ability to design computer programs in the OOP - Object Oriented Programming paradigm, creating effective source code and using the advantages of the object-oriented approach.

Aims of the Subject

- Ability to design a special-purpose information system – an intelligent information system.
- Familiarize students with knowledge related to embedded systems and other technologies, script creation, functional programming, and cloud-based systems.

Learning Outcomes

Knowledge

- Understands information systems and the process of information management within an organization.
- Identifies tools dedicated to data design, collection, and exploration.
- Defines the benefits of using databases in engineering and business practice.
- Formulates data models that describe objects and processes occurring in practice.
- Applies appropriate techniques for implementing data models.

Skills

- Designs databases and their structures.
- Implements and designs databases.
- Creates data integrity mechanisms.
- Writes queries for databases.
- Manipulates data.

Social Competencies

- Approaches data model building, database design, implementation, and exploration creatively.
- Actively utilizes databases in engineering and business practice.
- Focuses on the effective use of database design and exploration tools.
- Is aware of the limitations of data models and their implementation.

Program Content

1. General characteristics of special-purpose systems. Characteristics of selected intelligent embedded systems. Real-life examples.
2. Designing an information system in conjunction with a specialized embedded component.
3. System installation and analysis of potential implementation errors.

Teaching Methods

- Laboratory exercises
- Workshop method
- Didactic discussions
- Project-based method
- Demonstrations

Education verification method: exam, activity during classes

Object oriented programming 2 – project

The aim of the course is to transfer knowledge in the field of the subject. The students will improve the ability to design computer programs in the OOP - Object Oriented Programming paradigm, creating effective source code and using the advantages of the object-oriented approach.

Aims of the Subject

- Ability to design a special-purpose information system – an intelligent information system.
- Familiarize students with knowledge related to embedded systems and other technologies, script creation, functional programming, and cloud-based systems.

Learning Outcomes

Knowledge

- Understands information systems and the process of information management within an organization.
- Identifies tools dedicated to data design, collection, and exploration.
- Defines the benefits of using databases in engineering and business practice.
- Formulates data models that describe objects and processes occurring in practice.
- Applies appropriate techniques for implementing data models.

Skills

- Designs databases and their structures.
- Implements and designs databases.
- Creates data integrity mechanisms.
- Writes queries for databases.
- Manipulates data.

Social Competencies

- Approaches data model building, database design, implementation, and exploration creatively.
- Actively utilizes databases in engineering and business practice.
- Focuses on the effective use of database design and exploration tools.
- Is aware of the limitations of data models and their implementation.

Program Content

1. General characteristics of special-purpose systems. Characteristics of selected intelligent embedded systems. Real-life examples.
2. Designing an information system in conjunction with a specialized embedded component.
3. System installation and analysis of potential implementation errors.

Teaching Methods

- Laboratory exercises

- Workshop method
- Didactic discussions
- Project-based method
- Demonstrations

Education verification method: exam, activity during classes

Online data base- lecture

Aims of the Subject

- Conveying knowledge in the subject area.
- Achieving the ability to design a practical relational database.
- Gaining proficiency in using a selected Database Management System (DBMS) for implementing a web-based practical database.

Learning Outcomes

Knowledge

- Gaining knowledge in database technology and database management systems on the global network.
- Understanding and knowing selected issues in data modeling.
- Knowing and understanding tools and data analysis languages (including SQL).

Skills

- Achieving the ability to design the structure of a practical relational database and selecting and using a database management system.
- Utilizing knowledge to solve complex and unusual IT problems.
- Using available literature sources for formulating and solving IT problems.

Social Competencies

- Understanding the need and knowing the possibilities of continuously improving professional, personal, and social qualifications.
- Understanding the need and knowing the possibilities for continuous self-improvement.
- Being aware of the importance and understanding the non-technical aspects and consequences of the activities of an IT engineer.

Program Content

1. Definition of databases and Database Management Systems (DBMS). Areas of application of database technology. Web-based databases. Global database services: library, search, and scientific databases.
2. Data models. Relational data model. Relations, relation schemas, attributes, and keys. Database schema.
3. Reducing redundancy. Normalization of relations. Examples of practical

databases.

4. Query languages: QBE (Query by Example) and SQL (Structured Query Language). Selected DBMS implementations for developing practical web-based databases.

Teaching Methods

- Lectures
- Project-based methods
- Laboratory exercises

Education verification method: exam, activity during classes

Online data base- lecture- project

Aims of the Subject

- Conveying knowledge in the subject area.
- Achieving the ability to design a practical relational database.
- Gaining proficiency in using a selected Database Management System (DBMS) for implementing a web-based practical database.

Learning Outcomes

Knowledge

- Gaining knowledge in database technology and database management systems on the global network.
- Understanding and knowing selected issues in data modeling.
- Knowing and understanding tools and data analysis languages (including SQL).

Skills

- Achieving the ability to design the structure of a practical relational database and selecting and using a database management system.
- Utilizing knowledge to solve complex and unusual IT problems.
- Using available literature sources for formulating and solving IT problems.

Social Competencies

- Understanding the need and knowing the possibilities of continuously improving professional, personal, and social qualifications.
- Understanding the need and knowing the possibilities for continuous self-improvement.
- Being aware of the importance and understanding the non-technical aspects and consequences of the activities of an IT engineer.

Program Content

1. Definition of databases and Database Management Systems (DBMS). Areas of application of database technology. Web-based databases. Global database services: library, search, and scientific databases.
2. Data models. Relational data model. Relations, relation schemas, attributes, and keys. Database schema.
3. Reducing redundancy. Normalization of relations. Examples of practical databases.
4. Query languages: QBE (Query by Example) and SQL (Structured Query Language). Selected DBMS implementations for developing practical web-based databases.

Teaching Methods

- Lectures
- Project-based methods
- Laboratory exercises

Education verification method: exam, activity during classes

Web technologies- project

Aims of the Subject

- Conveying knowledge in the subject area.
- Achieving the ability to design the functional structure of a web service.
- Gaining the capability to utilize web services and tools using selected software.

Learning Outcomes

Knowledge

- Gaining knowledge in using web services/tools on the global web with selected software.
- Understanding and knowing methods for analyzing the subject matter.
- Knowing and understanding selected issues in modeling methods.

Skills

- Achieving the ability to design the functional and informational structure of a web page/service.
- Using knowledge to solve complex and atypical IT problems.
- Utilizing available literature sources for formulating and solving IT problems.

Social Competencies

  - Demonstrating ongoing professional, personal, and social skill development.
  - Understanding the need for and knowing the possibilities of continuously improving one's competencies.
  - Being aware of and understanding the non-technical aspects and consequences of an IT engineer's work.

Program Content

1. **Description of the World Wide Web (WWW):**
   - Definition and typology of technology.
   - Examples of web services: email, FTP, information services, Internet TV and radio, social networking sites, web browsers, and internet communicators.
2. **SGML - Standard Generalized Markup Language:**
   - Selected subsets: HTML, VRML, WOML, MathML.
   - HTML – Hypertext Markup Language.
   - Data transmission protocols.
3. **Technical Conditions:**
   - Requirements (computer, peripherals, etc.) for listening to music (or internet radio) or watching TV streams (sports games or other broadcasts).
4. **Web Services and Programming Languages:**
   - Examples: Library of Congress (LoC) web service, Electronic Data Interchange (EDI), e-shopping, online bookstores like Amazon.com.
   - Useful programming languages for building web pages.
   - Web services for e-government and e-administration.

Teaching Methods

- Lectures
- Project-based methods
- Laboratory exercises

Education verification method: exam, activity during classes

Web technologies- tutorials

Aims of the Subject

- Conveying knowledge in the subject area.
- Achieving the ability to design the functional structure of a web service.
- Gaining the capability to utilize web services and tools using selected software.

Learning Outcomes

Knowledge

  - Gaining knowledge in using web services/tools on the global web with selected software.

- Understanding and knowing methods for analyzing the subject matter.
- Knowing and understanding selected issues in modeling methods.

Skills

- Achieving the ability to design the functional and informational structure of a web page/service.
- Using knowledge to solve complex and atypical IT problems.
- Utilizing available literature sources for formulating and solving IT problems.

Social Competencies

- Demonstrating ongoing professional, personal, and social skill development.
- Understanding the need for and knowing the possibilities of continuously improving one's competencies.
- Being aware of and understanding the non-technical aspects and consequences of an IT engineer's work.

Program Content

1. **Description of the World Wide Web (WWW):**
   - Definition and typology of technology.
   - Examples of web services: email, FTP, information services, Internet TV and radio, social networking sites, web browsers, and internet communicators.
2. **SGML - Standard Generalized Markup Language:**
   - Selected subsets: HTML, VRML, WOML, MathML.
   - HTML – Hypertext Markup Language.
   - Data transmission protocols.
3. **Technical Conditions:**
   - Requirements (computer, peripherals, etc.) for listening to music (or internet radio) or watching TV streams (sports games or other broadcasts).
4. **Web Services and Programming Languages:**
   - Examples: Library of Congress (LoC) web service, Electronic Data Interchange (EDI), e-shopping, online bookstores like Amazon.com.
   - Useful programming languages for building web pages.
   - Web services for e-government and e-administration.

Teaching Methods

- Lectures
- Project-based methods
- Laboratory exercises

Education verification method: exam, activity during classes

Aims of the Subject

- Convey knowledge about the subject area.
- Achieve the ability to design the structure of XML documents.
- Gain the capability to utilize XML standards in Electronic Data Interchange (EDI) systems and in the World Wide Web (WWW).

Learning Outcomes

Knowledge

- Gain knowledge in the use of XML standards and services on the global web with selected software.
- Understand and know methods for analyzing the subject matter.
- Know and understand selected issues related to modeling methods.

Skills

- Achieve the ability to design the structure of XML documents for data exchange in EDI systems and web services.
- Use knowledge to solve complex and atypical IT problems.
- Utilize available literature sources for formulating and solving IT problems.

Social Competencies

- Understand the need for and know the possibilities of continuous improvement of professional, personal, and social skills.
- Understand the need for and know the possibilities of continuous improvement of personal competencies.
- Be aware of and understand the non-technical aspects and consequences of an IT engineer's work.

Program Content

1. **Definition of Document Description Language – XML (eXtensible Markup Language):**
   - The origin and development of XML standards, including applications in World Wide Web services.
   - Examples of XML-defined forms in information services, library services, and data transmission.
2. **SGML – Standard Generalized Markup Language and XML Subset:**
   - Basic terms, user-defined tags, namespaces, Processing Instructions (PI) in XML documents.
   - Characteristics of XML Schema.
3. **Software for Processing XML Documents:**
   - Parsers and XML code readers.
   - XML document formatting.
   - XSL/XSLT (eXtensible Stylesheet Language / XSL Transformation).

- Query languages related to XML, such as XIRQL (eXtensible Information Retrieval Query Language).

4. **Using XML in Electronic Data Interchange (EDI) Systems:**
    - Applications in e-business, e-commerce, e-finance, e-banking, e-education.
    - Use in CMS/LMS (Content Management System / Learning Management System) services.

Teaching Methods

- Lectures
- Project-based methods
- Laboratory exercises

Education verification method: exam, activity during classes

XML technologies- workshops

Aims of the Subject

- Convey knowledge about the subject area.
- Achieve the ability to design the structure of XML documents.
- Gain the capability to utilize XML standards in Electronic Data Interchange (EDI) systems and in the World Wide Web (WWW).

Learning Outcomes

Knowledge

- Gain knowledge in the use of XML standards and services on the global web with selected software.
- Understand and know methods for analyzing the subject matter.
- Know and understand selected issues related to modeling methods.

Skills

- Achieve the ability to design the structure of XML documents for data exchange in EDI systems and web services.
- Use knowledge to solve complex and atypical IT problems.
- Utilize available literature sources for formulating and solving IT problems.

Social Competencies

- Understand the need for and know the possibilities of continuous improvement of professional, personal, and social skills.
- Understand the need for and know the possibilities of continuous improvement of personal competencies.
- Be aware of and understand the non-technical aspects and consequences of an IT engineer's work.

Program Content

1. **Definition of Document Description Language – XML (eXtensible Markup Language):**
   - The origin and development of XML standards, including applications in World Wide Web services.
   - Examples of XML-defined forms in information services, library services, and data transmission.
2. **SGML – Standard Generalized Markup Language and XML Subset:**
   - Basic terms, user-defined tags, namespaces, Processing Instructions (PI) in XML documents.
   - Characteristics of XML Schema.
3. **Software for Processing XML Documents:**
   - Parsers and XML code readers.
   - XML document formatting.
   - XSL/XSLT (eXtensible Stylesheet Language / XSL Transformation).
   - Query languages related to XML, such as XIRQL (eXtensible Information Retrieval Query Language).
4. **Using XML in Electronic Data Interchange (EDI) Systems:**
   - Applications in e-business, e-commerce, e-finance, e-banking, e-education.
   - Use in CMS/LMS (Content Management System / Learning Management System) services.

Teaching Methods

- Lectures
- Project-based methods
- Laboratory exercises

Education verification method: exam, activity during classes

Software engineering- lecture

Aims of the Subject

- Introduce students to issues, models, and stages of software development, including methodologies and supporting tools.
- Develop skills in specifying and formalizing software requirements, creating software models/designs using various methodologies (both structural and object-oriented) and supporting tools, designing interfaces, and participating in the processes of implementation, testing, validation, and software development.

Learning Outcomes

Knowledge

- Understand and select an appropriate software lifecycle model for a

given application.
- Formulate tasks for each stage of the software lifecycle.
- Describe system requirements using formal techniques.
- Analyze the structure of the designed application.
- Choose suitable methods for modeling applications.
- Propose a data coding system.
- Select a software quality assurance system.

Skills

- Analyze real-world systems for IT support.
- Formalize the informational content of documents.
- Develop a set of requirements for information systems.
- Design functional and informational structures of information systems using various techniques.
- Create application models using different modeling techniques.
- Apply principles of human-computer interface design and use appropriate application testing techniques.

Social Competencies

- Demonstrate creativity in analyzing and designing applications.
- Identify relationships between different phases of application development.
- Share ideas with others.
- Understand non-technical aspects and consequences of an IT engineer's work, including its impact on the environment and associated responsibilities.

Program Content

1. **Typical Stages of Software Development and Their Content:**
    - Software development models (waterfall, evolutionary, iterative, agile, XP).
    - Prototyping methods.
2. **Analysis of Information and Decision Systems:**
    - Modeling business processes and specifying document content.
    - BPMN and BNF notations.
3. **Types of Requirements:**
    - Requirement acquisition, consolidation, and documentation.
    - Preparation of software specifications according to IEEE standards and acceptance criteria.
    - Formalization notations (templates, scenarios, use cases, hierarchical lists).
    - Requirements management.
4. **Software Design/Modeling:**

- Structural and object-oriented methodologies.
- Notations.

5. **System Architecture Design:**
    - Overview of contemporary architectures.
6. **Detailed Techniques for Modeling Processes and Data Structures:**
    - Conceptual, logical, and implementation models.
    - Model mapping.
7. **CASE Environments:**
    - Role of data dictionaries and repositories.
    - Techniques for working with CASE tools.
    - Integrated software development environments.
8. **Standards and User Interface Design:**
    - Principles of designing a proper interface.
    - Supporting tools.
    - Internationalization of interfaces - issues and methods.
9. **Data Coding:**
    - Types and principles.
    - Code construction.
    - Check digits.
10. **Software Testing and Validation:**
    - Objectives, scope, and types of testing methods.
    - Organization of the testing process.
11. **Software Development During the Maintenance Phase:**
    - Process of making changes to software.
    - Configuration management.
12. **Software Quality Assurance Systems:**
    - TQM, ISO 9000x, CMM, and EFQM models.

Teaching Methods

- Problem-based lectures, multimedia techniques, thematic discussions.
- Projects, thematic discussions.

Education verification method: exam, activity during classes

Software engineering- workshop

Aims of the Subject

- Introduce students to issues, models, and stages of software development, including methodologies and supporting tools.
- Develop skills in specifying and formalizing software requirements, creating software models/designs using various methodologies (both structural and object-oriented) and supporting tools, designing interfaces, and participating in the processes of implementation, testing, validation, and software development.

Learning Outcomes

- **Knowledge**
    - Understand and select an appropriate software lifecycle model for a

    given application.
    - Formulate tasks for each stage of the software lifecycle.
    - Describe system requirements using formal techniques.
    - Analyze the structure of the designed application.
    - Choose suitable methods for modeling applications.
    - Propose a data coding system.
    - Select a software quality assurance system.
- **Skills**
    - Analyze real-world systems for IT support.
    - Formalize the informational content of documents.
    - Develop a set of requirements for information systems.
    - Design functional and informational structures of information systems using various techniques.
    - Create application models using different modeling techniques.
    - Apply principles of human-computer interface design and use appropriate application testing techniques.
- **Social Competencies**
    - Demonstrate creativity in analyzing and designing applications.
    - Identify relationships between different phases of application development.
    - Share ideas with others.
    - Understand non-technical aspects and consequences of an IT engineer's work, including its impact on the environment and associated responsibilities.

Program Content

Lectures

1. **Typical Stages of Software Development and Their Content:**
    - Software development models (waterfall, evolutionary, iterative, agile, XP).
    - Prototyping methods.
2. **Analysis of Information and Decision Systems:**
    - Modeling business processes and specifying document content.
    - BPMN and BNF notations.
3. **Types of Requirements:**
    - Requirement acquisition, consolidation, and documentation.
    - Preparation of software specifications according to IEEE standards and acceptance criteria.
    - Formalization notations (templates, scenarios, use cases, hierarchical lists).
    - Requirements management.
4. **Software Design/Modeling:**
    - Structural and object-oriented methodologies.
    - Notations.
5. **System Architecture Design:**
    - Overview of contemporary architectures.
6. **Detailed Techniques for Modeling Processes and Data Structures:**
    - Conceptual, logical, and implementation models.

    - Model mapping.
7. **CASE Environments:**
    - Role of data dictionaries and repositories.
    - Techniques for working with CASE tools.
    - Integrated software development environments.
8. **Standards and User Interface Design:**
    - Principles of designing a proper interface.
    - Supporting tools.
    - Internationalization of interfaces - issues and methods.
9. **Data Coding:**
    - Types and principles.
    - Code construction.
    - Check digits.
10. **Software Testing and Validation:**
    - Objectives, scope, and types of testing methods.
    - Organization of the testing process.
11. **Software Development During the Maintenance Phase:**
    - Process of making changes to software.
    - Configuration management.
12. **Software Quality Assurance Systems:**
    - TQM, ISO 9000x, CMM, and EFQM models.

Project

1. **Diagram Editors (e.g., MS Visio) and Working Principles:**
    - Using diagramming tools.
2. **Analysis - Description of Client Actions in a Real System:**
    - Analyzing client interactions.
3. **Analysis - Modeling Business Processes:**
    - Creating business process models.
4. **Analysis - Document Content and Formatting Analysis, BNF Notation:**
    - Analyzing document content and structure.
5. **Requirements - Modeling Requirements, Functional Trees, Context Diagrams:**
    - Creating requirement models and diagrams.
6. **CASE Tools - Working Principles:**
    - Using CASE tools effectively.
7. **Modeling Functional Structure of Applications:**
    - Designing application functionality.
8. **Data Modeling - ERD Diagrams:**
    - Creating Entity-Relationship Diagrams.
9. **Designing Data Coding Systems:**
    - Developing data coding strategies.
10. **Interface Design:**
    - Designing user interfaces.

Teaching Methods

- Problem-based lectures, multimedia techniques, thematic discussions.

- Projects, thematic discussions.

Education verification method: exam, activity during classes

Awareness of the decisions' value- workshop

Course Objectives

- Prepare students to analyze their own hierarchy of values.
- Demonstrate the relationship between the choices they make and their needs and values.
- Enhance students' ability to make independent and conscious decisions.

Learning Outcomes

In terms of knowledge:

- Students define and list the stages of decision-making.
- Students explain the specifics of individual and group decision-making.

In terms of skills:

- Students effectively use axiological concepts.
- Students analyze and verify their hierarchy of values.
- Students improve their ability to make decisions (both individually and in groups) based on their values.
- Students analyze different decision-making styles.

In terms of social competencies:

- Students justify their views and reasoning.
- Students demonstrate responsibility for their choices.
- Students maintain openness to different decision-making styles.
- Students focus on the needs of others when making decisions.

Course Content

1. Perception of values in selected philosophical currents (relativism, absolutism, objectivism, subjectivism).
2. Real versus declared values.
3. Selected axiological problems of the modern world.
4. Perception of good and evil.
5. Awareness of the current hierarchy of values.
6. Conflict of values.
7. Freedom and responsibility in action.
8. Goals and values in life planning – the significance of having a goal in life.

Teaching Methods

- Didactic discussion
- Brainstorming
- Dramatic techniques
- Workshop method

Education verification method: exam, activity during classes