



MINISTERUL EDUCAȚIEI
UNIVERSITATEA „AUREL VLAICU” DIN ARAD
310130 Arad, B-dul Revoluției nr. 77, P.O. BOX 2/158 AR
Tel.: 0040-257- 283010; fax. 0040-257- 280070
<http://www.uav.ro>; e-mail: rectorat@uav.ro
Operator de date cu caracter personal nr. 2929

SYLLABUS

1. Study programme

| | |
|-----------------------------------|--|
| 1.1. Higher education institution | „AUREL VLAICU” UNIVERSITY OF ARAD |
| 1.2. Faculty | Faculty of Exact Sciences |
| 1.3. Department | Department of Mathematics and Computer Science |
| 1.4. Field of study | Informatics |
| 1.5. Academic year | 2024-2025 |
| 1.6. Study level | Bachelor |
| 1.7. Study programme | Informatics |
| 1.8. Form of education | Full-time education |

2. Course details

| | |
|---|----------------------------------|
| 2.1. Name of the course | Graph Algorithms |
| 2.2. Course coordinator | Sorin Nădăban, Ph.D. |
| 2.3. Seminar/laboratory/project coordinator | Iacob Oana Roxana, IT specialist |
| 2.4. Study year | 2 |
| 2.5. Semester | 1 |
| 2.6. Evaluation type | SE |
| 2.7. Course type | Basic Discipline |

3. Estimated total time (hours per semester)

| | |
|---|----|
| 3.1. Hours per week | 4 |
| 3.2. Lecture hours per week | 2 |
| 3.3. Seminar/laboratory/project hours per week | 2 |
| 3.4. Total hours per curriculum | 56 |
| 3.5. Lecture hours per curriculum | 28 |
| 3.6. Seminar/laboratory/project hours per curriculum | 28 |
| Time division [Hours] | |
| 3.4.1. Independent study from textbooks, course support, bibliography and notes | 94 |
| 3.4.2. Additional reading | 20 |

| | |
|--|------------|
| 3.4.3. Preparing of seminars/laboratories/projects, homework, papers, portfolios and essay | 20 |
| 3.4.4. Tutorial coaching | 20 |
| 3.4.5. Examinations | 4 |
| 3.4.6. Other activities | 10 |
| 3.7. Total individual study hours | 94 |
| 3.8. Total hours per semester | 150 |
| 3.9. Number of ECTS credits | 6 |

4. Prerequisites (if applicable)

| | |
|-------------------------|---|
| 4.1. Curriculum related | - |
| 4.2. Competence related | - |

5. Conditions (if applicable)

| | |
|------------------------------------|---|
| 5.1. Conditions for the lecture | - |
| 5.2. Conditions for the seminar | - |
| 5.3. Conditions for the laboratory | - |
| 5.4. Conditions for the project | - |

6. Specific educational objectives (competences to be acquired)

| | |
|-------------------------------------|---|
| 6.1. Professional competences | C1 Programming in high-level languages C3. Using professional tools in an interdisciplinary context. C4. Using the theoretical foundations of computer science and formal models. |
| 6.2. Transversal competences | CT1. Apply the rules of effective and rigorous work, responsibility for scientific domain, for optimal skills. CT2. Effective conduct of organized activities in an interdisciplinary group and development of empathetic capacities of interpersonal communication, relationship and collaboration with various groups. CT3. Use of effective methods and techniques of learning, information, research and development of the capacity to capitalize on knowledge, adapting to the requirements of a dynamic society and communication in an international language. |

7. Course outcomes (resulting from the specific educational objectives to be acquired)

| | |
|------------------------|--|
| 7.1. General outcomes | <ul style="list-style-type: none"> • The student should know the basic concepts and understand graph algorithms. • The student should develop the ability to correctly apply the acquired knowledge to solve various types of problems. • The student should develop thinking and deductive reasoning skills for solving complex graph theory problems. |
| 7.2. Specific outcomes | <ul style="list-style-type: none"> • - The student is able to demonstrate that they have acquired sufficient knowledge to understand the basic concepts. • The student is capable of correctly applying the fundamental methods and principles in solving graph algorithm problems. |

| | |
|--|--|
| | <ul style="list-style-type: none"> • The student is capable of recognizing the main classes/types of graph algorithm problems and selecting appropriate methods and techniques for solving them. • The student is able to complete projects for the mathematical modeling of a specific problem. |
|--|--|

8. Course outline

| 8.1 Lecture | Teaching methods | Remarks |
|--|--|---------|
| <p>1. Introduction to Graphs</p> <p>1.1. Graph representation</p> <p>1.2. Graph traversal: breadth-first search, depth-first search, topological sorting</p> <p>1.3. Operations on graphs</p> <p>1.4. Paths, circuits, and chains</p> <p>1.5. Trees</p> <p>2. Algorithms for Directed Graphs</p> <p>2.1. Path matrix: Roy-Warshall algorithm, Boolean composition method, Chen's algorithm, Kaufmann's algorithm</p> <p>2.2. Determining connected components</p> <p>2.3. Determining strongly connected components: Malgrange's algorithm, Chen's algorithm, Foulkes' algorithm</p> <p>2.4. Hamiltonian paths and circuits: Kaufmann's algorithm, Foulkes' algorithm, Chen's algorithm</p> <p>2.5. Optimal path algorithms: Ford's algorithm, Bellman-Kalaba algorithm, Dijkstra's algorithm, Floyd-Warshall algorithm</p> <p>2.6. Transport networks: Ford-Fulkerson algorithm</p> <p>2.7. Scheduling problems</p> <p>3. Algorithms for Undirected Graphs</p> <p>3.1. Determining Eulerian circuits</p> <p>3.2. Minimum spanning tree: Prim's algorithm, Kruskal's algorithm</p> <p>3.3. Assignment problems: Little's algorithm, Hungarian algorithm</p> | <p>Participatory lecture, presentation, problematization, exemplification, demonstration, dialogue</p> | |
| <p>8.2 Lecture references</p> <ol style="list-style-type: none"> 1. S. Nădăban, A. Şandru, <i>Graph Algorithms</i>, Editura Mirton, Timișoara, 2007 2. R. Diestel, <i>Graph Theory</i>, Springer-Verlag, Graduate Texts in Mathematics, vol 173, 2000 3. B. Korte, J. Vygen, <i>Combinatorial Optimization: Theory and Algorithms</i>, Springer, 2000 4. S. Nădăban, <i>Graph Algorithms</i>, course and lab support, 2017 5. J.A. Bondy, U.S.R. Murty, <i>Graph Theory</i>, Springer, 2008 | | |
| 8.3 Seminar | Teaching methods | Remarks |

| | | |
|---|---|---------|
| 1. Differential calculus 1.1. Number series and number sequences; 1.2. Limit and continuity; 1.3. Classes of functions; 1.4. Differential calculus of real functions. 2. Integral calculus 2.1. Primitivable functions; 2.2. Integrable functions; 2.3. Generalised integrals; 2.4. Sequences and series of functions. 3. Differential and integral calculus in \mathbb{R}^n 3.1. Differential calculus in \mathbb{R}^n 3.2. Integral calculus in \mathbb{R}^n . | Exercise, discussion and debate, exemplification, project | |
| 8.4 Seminar references 1. S.Nădăban, Calculus- Elemente de calcul diferential si integral, Editura Mirton, Timisoara, 2010. 2. S.Nădăban, MathEco - Analiză matematică, Editura Mirton, Timisoara, 2001. 3. Edwin Jed Herman, Gilbert Strang, Calculus Volume 1, Openstax, Rice University, 2020. 4. Paul Dawkins, Calculus 2, 2007. 5. MIT Opencourseware, Single Variable Calculus, 2006. 6. S.Nădăban, Calcul diferential si integral, suport de curs si seminar, SUMS 2024. | | |
| 8.5 / Laboratory | Teaching methods | Remarks |
| 1. Introduction to Graphs 1.1. Graph representation 1.2. Graph traversal: breadth-first search, depth-first search, topological sorting 1.3. Operations on graphs 1.4. Paths, circuits, and chains 1.5. Trees 2. Algorithms for Directed Graphs 2.1. Path matrix: Roy-Warshall algorithm, Boolean composition method, Chen's algorithm, Kaufmann's algorithm 2.2. Determining connected components 2.3. Determining strongly connected components: Malgrange's algorithm, Chen's algorithm, Foulkes' algorithm 2.4. Hamiltonian paths and circuits: Kaufmann's algorithm, Foulkes' algorithm, Chen's algorithm 2.5. Optimal path algorithms: Ford's algorithm, Bellman-Kalaba algorithm, Dijkstra's algorithm, Floyd-Warshall algorithm 2.6. Transport networks: Ford-Fulkerson algorithm 2.7. Scheduling problems 3. Algorithms for Undirected Graphs 3.1. Determining Eulerian circuits 3.2. Minimum spanning tree: Prim's algorithm, Kruskal's algorithm 3.3. Assignment problems: Little's algorithm, Hungarian algorithm | Problem-based learning, exemplification, dialogue | |
| 8.6 Laboratory references | | |

1. S. Nădăban, A. Şandru, *Graph Algorithms*, Editura Mirton, Timişoara, 2007
2. R. Diestel, *Graph Theory*, Springer-Verlag, Graduate Texts in Mathematics, vol 173, 2000
3. B. Korte, J. Vygen, *Combinatorial Optimization: Theory and Algorithms*, Springer, 2000
4. S. Nădăban, *Graph Algorithms*, course and lab support, 2017
5. J.A. Bondy, U.S.R. Murty, *Graph Theory*, Springer, 2008

| | | |
|------------------------|------------------|---------|
| 8.7 / Project | Teaching methods | Remarks |
| 8.8 Project References | | |

9. Corroboration / validation of course putline (if applicable)

The course content is consistent with what is done in other universities in the country and abroad. For a better adaptation to the labor market requirements of the content of the discipline, meetings were held with representatives of employers.

10. Evaluation / Grading

| Activity type | Evaluation criteria | Evaluation methods | Percentage of the final grade |
|---|---|--|-------------------------------|
| 10.1. Lecture | <input type="checkbox"/> knowledge; <input type="checkbox"/> logical coherency; <input type="checkbox"/> acquiring the specialty language; <input type="checkbox"/> criteria that envisage attitudinal aspects: seriousness, conscientiousness and interest for the subject. | 1. Written exam (final, during the exam session) 2. Active participation in courses | 40% 10% |
| 10.2. Seminar | | | |
| 10.3. Laboratory | -capacity of using the acquired knowledge; -capacity of applying in practice; -conscientiousness and interest for the study. | 1. Ongoing written work: assignments, projects 2. Final written exam (during the exam session) 3. Active participation in labs | 50% |
| 10.4. Project | | | % |
| 10.5 Minimal performance standard Correct understanding of the basic theoretical concepts and their application in solving a simple problem. | | | |

Titular
Prof. dr. Nădăban Sorin

Asistent
Iacob Oana Roxana, IT specialist

DIRECTOR DEPARTMENT
Lect. dr. Lorena Popa

DECAN
Prof. dr. Nădăban Sorin