



Generation AI Curriculum Framework and Educational Program

For Grades 10-12

Version: 0.5

Last edition: Jun 3, 2025

Pages: 67



FOUNDATION FOR ARMENIAN SCIENCE AND TECHNOLOGY

3 Hakob Hakobyan Street, Yerevan, Republic of Armenia

education@fast.foundation

Content of table

PREFACE	2
SYLLABUS AND ACADEMIC PLAN	4
EMPLARY SYLLABUS and ACADEMIC PLAN	6
FUNDAMENTAL COMPETENCIES	7
SUBJECT PROGRAMMES	12
ALGEBRA AND ELEMENTS OF MATHEMATICAL ANALYSIS (ADVANCED)	12
COMPUTER SCIENCE WITH PYTHON PROGRAMMING LANGUAGE	15
ARTIFICIAL INTELLIGENCE	38

PREFACE

This document establishes the curriculum framework and subject-specific educational programs for the "Generation AI" school initiative, developed as integral components of a comprehensive general education program. The curriculum is founded upon the strategic objective of the ["GenerationAI" project](#): to cultivate a generation of future innovators and leaders in the field of Artificial Intelligence.

The curriculum is designed for Grades 10-12 with particular emphasis on the intensive study of machine learning. **The curriculum has been developed in accordance with general education processes and is intended for 2.5 academic years or 5 semesters.**

International Experience Study and Comparison

During the development of the "Generation AI" curriculum, the Fast Foundation conducted a comprehensive study of existing international experience in artificial intelligence education, particularly within the sphere of secondary education, with the objective of creating a program that corresponds to the Armenian context and the goals of the "Generation AI" project, whilst simultaneously aligning with global standards.

During the development process, international experience was considered as a foundation, particularly the [AI4K12 initiative](#) from the United States, which proposes a progressively developing framework of artificial intelligence education skills appropriate for secondary education levels. Specifically, the "5 Big Ideas in AI" have been taken into consideration, which serve as important guideposts within the curriculum. These are:

1. **Perception:** AI systems receive information from the external world through sensors and transform it into comprehensible representations.
2. **Representation & Reasoning:** AI utilizes mathematical models and logical structures to represent knowledge and make decisions.
3. **Learning:** AI systems can improve their performance through experience and data analysis without additional specific programming.
4. **Natural Interaction:** AI facilitates communication with humans through recognition of language, voice, images, or gestures.
5. **Societal Impact:** AI transforms society, bringing forth new ethical, legal, and social challenges and opportunities.

Based on the "[AI Competency Framework for Students](#)" published by UNESCO in September 2024, a comparative analysis was conducted. Specifically, the four core competency areas defined within that framework were studied and compared: 1. human-centered thinking and collaboration, 2. AI ethics, 3. AI technology and applications, 4. AI system design. The content analysis of the curriculum demonstrated that its structure and objectives naturally align with these designated areas, encompassing both technical and value-based ethical components. Concurrently, certain additions were made to the curriculum, particularly within the AI literacy and AI ethics modules, to ensure its correspondence with the international standards established by UNESCO.

Additionally, the experiences of other countries are being studied and compared in parallel¹—**China, India, Singapore, Norway, South Korea**, and the **UAE**—where AI is being progressively integrated across all educational levels, from primary school through to secondary education and higher education. The analysis of these experiences enabled additional refinements to be made, ensuring the curriculum's competitiveness and contemporary relevance.

Active steps in the direction of artificial intelligence education are being undertaken in numerous countries, incorporating it across various educational levels from primary through to higher education. However, the comparison of these experiences becomes effective only when it is clearly defined what is meant by "AI education"—considering the depth of knowledge, level of content, and format of application. Within this context, three main fundamental approaches to AI education can be distinguished:

1. **Application of AI tools in Education.** The use of artificial intelligence tools in the teaching and learning process—for example, for the personalization of educational content, automation of assessment, or enhancement of teaching effectiveness. In this case, AI functions as an auxiliary tool rather than a subject of study.

¹In China, AI education is incorporated at the state level from primary school through to secondary education. The educational program includes the study of AI history, development of technical skills, as well as discussion of ethical and social impacts.

In India, AI has been included at the school level as a "Skill Subject" for grades 6-12 ([NEP 2020 policy](#)). Singapore develops students' programming and AI project capabilities through the [Student Outreach Programme](#) of the "AI Singapore" initiative. Norway focuses on AI learning in higher education and [research](#). South Korea is introducing AI-enhanced [textbooks](#) in schools in 2025 for personalized learning purposes. Meanwhile, in the United Arab Emirates, [AI education](#) begins from age 4, encompassing ethics and prompt engineering.

1. **Use of Ready-made AI Models.** Students apply fully trained and fine-tuned AI models (e.g., through APIs), understanding their operational principles without deeply studying the underlying algorithms and mathematical foundations. This approach is primarily focused on the development of practical skills and AI technological literacy.
2. **Development of AI Systems.** In this direction, students thoroughly study the structure of AI models, programming and mathematical algorithms, ultimately acquiring capabilities for designing, training, and improving their own AI models. This approach requires systematic education in mathematics, statistics, and programming.

The "Generation AI" school program curriculum is focused precisely on the third level—"AI Development"—on the development and in-depth study of models, ensuring comprehensive advancing of students' knowledge and skills.

The curriculum has also been formulated based on international educational standards, combining leading global approaches with local experience. It aligns with the US Next Generation Science Standards (NGSS) and Computer Science Teachers Association (CSTA) standards, the EU DigCompEdu and AI and Education initiatives, the UK Computing Program of Study, and China's AI Curriculum Guidelines.

Consequently, the developed curriculum for the "Generation AI" school program aims to integrate mathematical thinking, statistical analysis, programming skills, and practical AI development, cultivating students' preparedness for scientific, industrial, and creative fields alike.

SYLLABUS AND ACADEMIC PLAN

The following subjects are defined as mandatory components of the "Generation AI" school program syllabus:

1. **"Advanced Algebra"** in accordance with the requirements of the state standard and program for the subject ["Algebra and Elements of Mathematical Analysis"](#) intended for senior grades, and the methodology developed by the Fast Foundation, which facilitates the applicability of acquired knowledge and prepares students for the study of artificial intelligence.
2. **"Computer Science with Python Programming Language"** in accordance with the curriculum defined by the Fast Foundation and approved by the MESCS.

3. **"Artificial Intelligence"** and **"AI Project-Based Learning"** in accordance with the curriculum defined by the Foundation and approved by the MESCS.
4. **The extracurricular components**, according to subjects, are program defined by the Foundation and endorsed by the MESCS: "Artificial Intelligence in Real Life", "Python Programming in Real Life", and "Algebra in Real Life" project groups. The objective of these project-based extracurricular groups, which have one weekly extracurricular lesson, is to provide senior grade students with the opportunity to undertake project-based learning within the framework of the program's three subjects in an extracurricular format and in small groups, in order to deepen and apply acquired knowledge and skills, and to understand their applicability in real life.

The courses in the "Generation AI" school program syllabus do not replace other school subjects. Students study the subject "Algebra and Elements of Mathematical Analysis" using the methodology developed by the Foundation.

The program is designed for the third level of secondary education (Grades 10-12) as follows:

10 th grade	11 th grade	12 th grade
Compulsory subject		
Algebra and Elements of Mathematical Analysis	Algebra and Elements of Mathematical Analysis	Algebra and Elements of Mathematical Analysis
Artificial Intelligence	Artificial Intelligence	Artificial Intelligence
Computer Science: Python Language	Computer Science: Python Language	AI final project-led learning
Extracurricular Component (e.g. PBL, iBL, CBL)		
AI Professional Orientation	Artificial Intelligence in Real Life	Artificial Intelligence in Real Life
Algebra in Real Life	Algebra in Real Life	Algebra in Real Life
Python Programming in Real Life	Python Programming in Real Life	Python Programming in Real Life

EMPLARY SYLLABUS and ACADEMIC PLAN

Subject Group	Subject	Weekly Hours				
		X-1	X-2	XI-1	XI-2	XII-1
Track Component		17	17	18	18	18
Advanced Study Subjects	Algebra and Elements of Mathematical Analysis	6	6	6	6	6
	Geometry	3	3	3	3	3
	Natural Science Subject (Physics is recommended)	4	4	4	4	4
	Computer Science (Python Programming)	4	2	2	2	0
	Artificial Intelligence (AI)	0	3	3	3	3
	AI final project-led learning	0	0	0	0	2
General Education Component		16	16	15	15	15
State-Mandated Subject List	Armenian Language	2	2	2	2	2
	Armenian Literature	3	3	3	3	3
	Armenian History	3	3	3	3	3
	Social Studies	2	2	2	2	2
	English	2	2	2	2	2
	Physical Education	2	2	2	2	2
	NMP (Initial Military Training)	1	1	1	1	1
	DLCS (Digital Literacy and Computer Science)	1	0	0	0	0
Total		33	33	33	33	33

Individual (Elective) Component	18				
AI Professional Orientation (from compulsory PO group class hours)	1	0	0	0	0
Artificial Intelligence in Real Life	0	3	3	3	3
Python Programming in Real Life	3	3	3	3	3
Algebra in Real Life	3	3	3	3	3

FUNDAMENTAL COMPETENCIES

The curriculum and subject program of the "Generation AI" are built upon three fundamental competency areas that reflect the educational and technological requirements of the 21st century. These competencies have been formulated based on combining international and local standards, including the RA State Standard for General Education, UNESCO, OECD, EU, AI4K12, and WEF corresponding documents.²

A. Global Digital Citizenship Educational Competencies.

1. **Digital and AI Literacy.** Data analysis, information evaluation, digital security, and recognition of reliable sources.
2. **Inclusive and Ethical Thinking.** Understanding, perception, and application of the moral aspects of artificial intelligence and technology implementation.
3. **Communication and Collaborative Skills.** Effective communication and teamwork in diverse digital environments.
4. **Self-Learning and Self-Assessment.** Setting learning objectives, selecting learning strategies, and regular self-assessment for self-development purposes.

B. Professional competencies

² The RA State Standard for General Education as a mandatory foundation for ensuring compliance with [national educational standards](#) for educational content; [UNESCO AI Competency Framework for Students](#), which aims to enhance students' preparedness for the AI digital age; [UNESCO Global Citizenship Education](#), which promotes the development of universal values, coexistence, and social responsibility; WEF Future of Jobs Report (2025), focused on the skills and competencies required for the future job market; OECD Learning Compass 2030; EU Key Competences for Lifelong Learning; AI4K12 Five Big Ideas in AI.

1. **Mathematical Thinking and Application.** The student can apply fundamental mathematical knowledge and skills (advanced-level functions, probability theory, and linear algebra) for modelling, analysis, and solving real-world problems.
2. **Advanced Programming Competency (Python language).** Solving real-life problems through structural, object-oriented, and data-driven thinking.
3. **Data Science, Statistics, and Machine Learning.** The student can collect/gather data, visualize, process, analyze, and draw conclusions using machine learning methods and algorithms.
4. **Neural Networks and Deep Learning.** The student can utilize basic neural architectures and apply classical algorithms. Master's various neural networks and their applications. Can work with textual data, applies lexical models and technologies for natural language processing (NLP) domains.
5. **AI Scientific Thinking and Research Capabilities.** During the learning process, the student develops advanced scientific thinking according to artificial intelligence development trends. Can apply acquired knowledge for independent research project implementation, study scientific articles, and analyze professional and current scientific literature. This competency aims to prepare students for continuing further education at higher education levels (undergraduate, graduate) as well as subsequently commencing research activities (PhD).

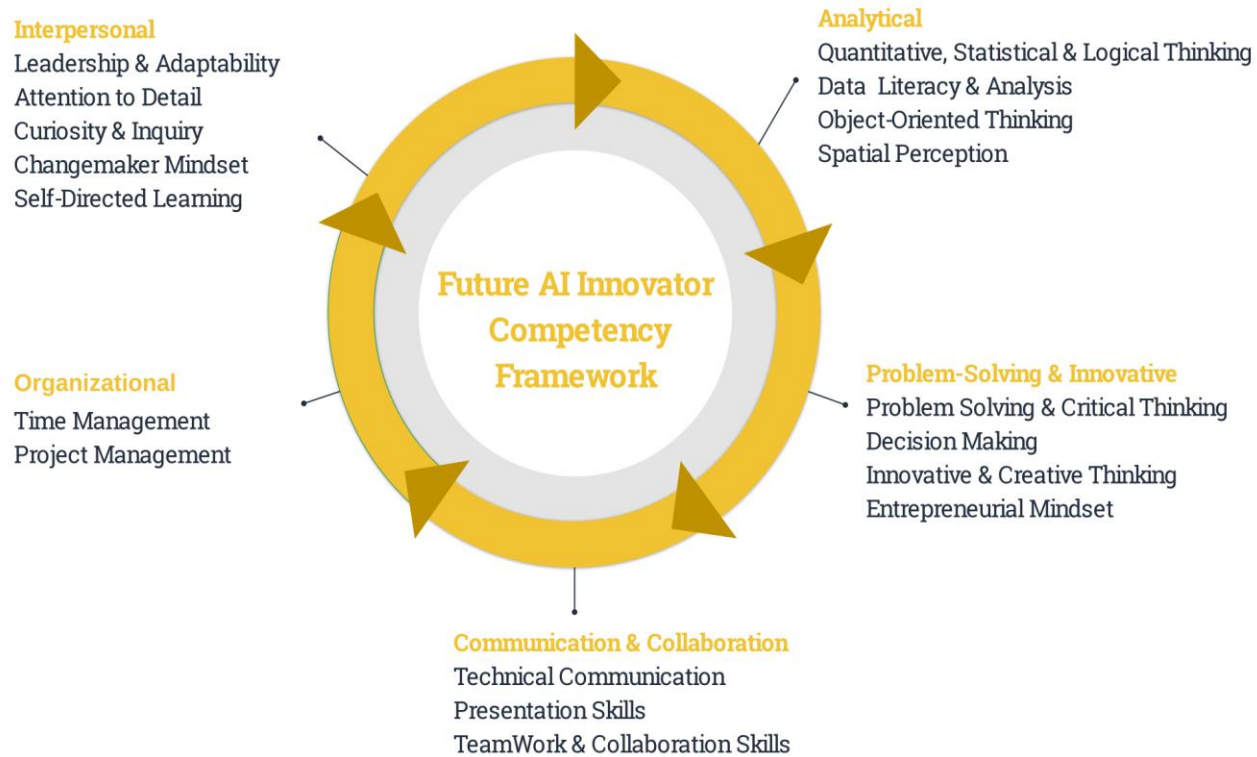
C. Personal and innovation-oriented competencies

1. **Analytical Thinking and Problem-Solving Skills.** Students can analyze complex problems by breaking them down into smaller steps, evaluate different solutions, and select effective approaches.
2. **Innovative and Creative Thinking.** Students develop the ability to independently formulate ideas for technological projects, test and propose innovative solutions, particularly for the open and non-standard problems that frequently arise in the AI field.
3. **Career Planning and Professional Orientation.** Students acquire the necessary skills to recognize their interests, study different professions, and envision their future professional path.

This table summarizes the core soft and hard/digital skills that students acquire within the framework of the subjects "Artificial Intelligence" (AI), "Advanced Algebra and Elements of Mathematical Analysis", and "Python Programming" in grades 10–12. The skills are presented not by grade levels but organized according to thematic content. They include technical knowledge and practical capabilities, as well as teamwork, communication, critical thinking, and other soft/hard skills.

Soft skills	Digital or hard skills
Critical Thinking	Mathematical Thinking
Analytical Thinking	Numerical Analysis Skills
Problem Solving	Statistical Modelling Capability
Observation Skills	Analysis of Functions, Graphs, and Variables
Attention	Data Collection, Organization, and Analysis Capability
Interest	Knowledge of Fundamental Artificial Intelligence Concepts
Teamwork	Python Programming Language Application
Adaptability	Knowledge of Programming Structural Elements
Communication	Application of Data Structures and Algorithms
Organizational Skills	Construction and Training of Machine Learning Models
Decision Making	Data Preparation and Cleaning Skills
Creativity	Model Quality Assessment and Optimization

Innovative Thinking	Application of Mathematical Models to Real-World Problems
Leadership	Modelling and Analysis of Complex Systems
Collaboration	Technical Documentation Development and Formatting
Strategic Thinking	Version Control and Debugging
Systems Thinking	Application of Data Science and AI Tools (Colab, Jupyter, Scikit-learn, NumPy, Matplotlib, PyTorch, TensorFlow)
Independence	Application of Core AI Models (Regression, Decision Tree, CNN, NLP, Transformers)
Ability to Assimilate New Information	
Planning Skills	



All competencies and skills developed within the program are directed towards the holistic development of the learner, integrating theoretical knowledge with practical application. Soft skills contribute to the enhancement of critical thinking, collaboration, creativity, and communication capabilities, whilst hard or digital skills provide technological literacy, data analysis, modelling, and programming competencies, enabling pupils to navigate confidently within the digital and complex problem-rich environment of the 21st century.

SUBJECT DETAILED CURRICULUM

ALGEBRA AND ELEMENTS OF MATHEMATICAL ANALYSIS (ADVANCED)

The "Algebra and Elements of Mathematical Analysis" subject program fully aligns with the mathematics curriculum prescribed by the State Educational Standards for General Education (arlis.am/hy/acts/53585). However, in this instance, the Foundation develops a new methodology that meets contemporary requirements, modernizing the content and integrating directions essential for artificial intelligence. This enables students to understand more profoundly the applied significance of mathematics and its connection with fundamental concepts of artificial intelligence.

A key characteristic of the methodology is the meticulously constructed system of learning outcomes, which defines more detailed outcomes for each chapter. The 'simple to complex' explanatory methodology is incorporated, aiming to ensure accessibility and opportunities for progress for every learner. The active integration of simulation platforms, particularly GeoGebra, helps make theoretical concepts visible and interactive.

For each topic, links to online materials, interactive platforms, and additional resources for struggling learners are provided. The methodology is also distinguished by its student-centered approach, where teamwork is encouraged through numerous examples of paired and group learning. The table presented below describes the sequence of topic learning across 2.5 academic years.

For the development and assessment of mathematical competencies, an individualized approach is recommended, including exemplar test papers and an assessment system with indicators aligned to the standards. The topics are also connected to real-life examples, contributing to students' motivation and awareness of the applicability of mathematics.

SUBJECT: Algebra and Elements of Mathematical Analysis			
Grade	Semester	Topic	Class hours
10th	1st	<ul style="list-style-type: none"> - Real Numbers - Elements of Trigonometry 	74
	2nd	<ul style="list-style-type: none"> - Numerical Functions - Trigonometric Functions of Numerical Arguments and Trigonometric Equations - Probability Theory and Statistics 	96
			170
11th	1st	<ul style="list-style-type: none"> - Power and Exponential Functions - Logarithmic Functions 	72
	2nd	<ul style="list-style-type: none"> - Numerical Sequences, Limits - Function Continuity: Derivatives - Conditional Probability: Normal Distribution 	98
			170
12th	1st	<ul style="list-style-type: none"> - Equations and Inequalities - Integrals - Combinatorics and Probability Theory, Statistics 	70
Total			410

SUBJECT DETAILED CURRICULUM
COMPUTER SCIENCE WITH PYTHON
PROGRAMMING LANGUAGE

SUBJECT: COMPUTER SCIENCE WITH PYTHON PROGRAMMING LANGUAGE		Grade: 10	Semester: 1
Level: Programming Fundamentals	Lesson hours: 3	Theoretical: 2	Practical: 1
Topic: Introduction to Python (I)			Chapter: 1
Objective		Learning Outcomes	
Develop a general understanding of the nature of programming and the application possibilities of the Python language.		Upon completion of this topic, learners will be able to: 1. Explain what programming is in their own words. 2. Present the main areas of Python language application (analysis, automation, AI, web development, etc.). 3. Identify and correct simple syntactic errors.	
Content	Teaching Activities	Tools/Platforms	
Introduction to the Python language and the concept of programming.	1. Conduct discussions and present real-life examples. 2. View short video clips. 3. Implement mapping of programming domains. 4. Carry out group discussions and written summaries.	Computer, projector, video materials (YouTube or video lessons), classroom pen/notebook.	
Cross-curricular topics: Technology, Informatics.			
Assessment Criteria:			
1. Presents at least 4 examples from Python application domains. 2. Explains what programming is in their own words.			

Prerequisite Knowledge: Has general computer literacy, mouse and keyboard operation skills.

SUBJECT: COMPUTER SCIENCE WITH PYTHON PROGRAMMING LANGUAGE		Grade: 10	Semester: 1
Level: Programming Fundamentals	Lesson hours: 4	Theoretical: 1	Practical: 3
Topic: Syntax (2)			Chapter: 1
Objective		Learning Outcomes	
Learn the basic syntax of the Python language.		Upon completion of this topic, learners will be able to: 1. Interpret and execute a simple Python program. 2. Use the print () command to output text. 3. Use comments to explain code. 4. Save a program as a .py file and execute it. 5. Construct a simple conditional program that produces different outputs based on different input data.	
Content	Teaching Activities	Tools/Platforms	
Python program structure and syntax	1. Create a "Hello, World!" program. 2. Understand program file structure and comments. 3. Write and execute a simple program using comments. 4. Carry out small group work on correcting syntactic errors in code.	IDLE, Thonny, or Profound Academy platform, notebook, whiteboard.	
Cross-curricular topics: Technology, Informatics.			
Assessment Criteria: 1. Executes the program without errors. 2. Uses comments (#) in the code. 3. Saves the file with an appropriate name (e.g., example.py).			
Prerequisite Knowledge: Has basic English vocabulary (print, input, etc.).			

SUBJECT: COMPUTER SCIENCE WITH PYTHON PROGRAMMING LANGUAGE		Grade: 10	Semester: 1
Level: Programming Fundamentals	Lesson hours: 5	Theoretical: 1	Practical: 4
Topic: Inputs, Outputs (3)			Chapter: 1

Objective		Learning Outcomes	
Develop skills in using input () and print () functions for receiving and displaying simple data.		Upon completion of this topic, learners will be able to: 1. Write and interpret a simple interactive program that receives data from the user and prints a response. 2. Explain the effect of variable types in programs with examples (str, int).	
Content	Teaching Activities	Tools/Platforms	
Input data (print (), input ()) printing, output and receiving.	1. Create a program that receives name and age, prints a message. 2. Test with different input data. 3. Conduct discussion about variable types and their effects. 4. Write a simple explanation with examples of variable types (str, int).	IDLE, Thonny, Profound Academy platform, input/output worksheet.	
Cross-curricular Topics: Mathematics (calculations), Language (speech and communication).			
Assessment Criteria: 1. The program correctly accepts two input values (input) and outputs (print) a combined message. 2. Uses variables and expressions (e.g., print ("Hi " + name)).			
Prerequisite Knowledge: Has a concept of what data is, how it is communicated to the computer.			

SUBJECT: COMPUTER SCIENCE WITH PYTHON PROGRAMMING LANGUAGE		Grade: 10	Semester: 1
Level: Programming Fundamentals	Lesson hours: 5	Theoretical: 2	Practical: 3

Topic: Variables (4)			Chapter: 1
Objective		Learning Outcomes	
Learn how to store and use values in programs through variables.		Upon completion of this topic, learners will be able to: 1. Define variables with different types (number, text), apply them in programs and change their values.	
Content	Teaching Activities	Tools/Platforms	
Defining and using variables.	1. Write a simple program that uses variables to store data (e.g., age, name). 2. Test variable value conversions (str to int and vice versa). 3. Conduct group discussion about variable types and assignments. 4. Write a simple explanation about the purposes of using variables.	IDE (IDLE, Thonny), printed manual, whiteboard, markers, pen, A4 paper.	
Cross-curricular Topics: Mathematics (representation with variables), Economics (model variables), Technology.			
Assessment Criteria: 1. Differentiates and applies basic variables (str, int). 2. Changes value and assign to variables. 3. Uses at least two variables in the program.			
Prerequisite Knowledge: Has the skill to differentiate between numbers and text, concept of simple expressions.			

SUBJECT: COMPUTER SCIENCE WITH PYTHON PROGRAMMING LANGUAGE		Grade: 10	Semester: 1
Level: Programming Fundamentals	Lesson hours: 6	Theoretical: 2	Practical: 4
Topic: Types, Strings (5)			Chapter: 2
Objective		Learning Outcomes	
Differentiate between basic data types: integer, float,		Upon completion of this topic, learners will be able to:	

string and boolean.		1. Differentiate between data types: int, float, str and others. 2. Apply basic types to text operations. 3. Apply type conversions with data.
Content	Teaching Activities	Tools/Platforms
Data types: int, float, str. Text concatenation, length determination, input and output. Type conversion from str to int, float to str, etc.	1. Analyse and solve examples using data type applications. 2. Solve small problems using type differentiation and conversion applications.	IDLE, Thonny, Profound Academy platform, notebook, pen/pencil.
Cross-curricular Topics: Languages (text analysis), Data Science.		
Assessment Criteria: 1. Can determine the data type when working with variables. 2. Applies appropriate functions (e.g., len, type). 3. Writes a program that uses type conversion.		
Prerequisite Knowledge: Masters basic concepts of variables, input/output.		

SUBJECT: COMPUTER SCIENCE WITH PYTHON PROGRAMMING LANGUAGE		Grade: 10	Semester: 1
Level: Programming Fundamentals	Lesson hours: 6	Theoretical: 2	Practical: 4
Topic: Numerical Operators (6)			Chapter: 2
Objective	Learning Outcomes		
Apply numerical operators for solving simple problems.	Upon completion of this topic, learners will be able to: 1. Apply +, -, *, /, //, %, ** operators in programs. 2. Write a simple problem-solving program using numerical operators.		
Content	Teaching Activities	Tools/Platforms	

Arithmetic operations in Python. Global and local variables, order of operations.	1. Analyse and solve examples using numerical operators. 2. Solve and discuss competitive problems.	IDLE, Thonny, Profound Academy platform, pen/pencil, worksheet.
Cross-curricular Topics: Mathematics (basic operations), Physics (measurements and calculations).		
Assessment Criteria: 1. Solves simple problems using 2 or more numerical operators. 2. Explains the order of operations.		
Prerequisite Knowledge: Masters basic types and concepts of simple mathematical calculations.		

SUBJECT: COMPUTER SCIENCE WITH PYTHON PROGRAMMING LANGUAGE		Grade: 10	Semester: 1
Level: Programming Fundamentals	Lesson hours: 6	Theoretical: 2	Practical: 4
Topic: Logical Operators (7)			Chapter: 3
Objective		Learning Outcomes	
Apply logical operators for solving simple problems.		Upon completion of this topic, learners will be able to: 1. Differentiate between and, or, not operators and apply them in writing conditions. 2. Construct and use simple logical expressions.	
Content	Teaching Activities		Tools/Platforms
Comparison operators: ==, !=, >, <, >=, <=. Logical operators: and, or, not. Formation of truth tables.	1. Present theoretical material in a simple and structured manner. 2. Solve exercises step-by-step using logical operators. 3. Compare conditions and analyze results.		IDLE, Thonny, Profound Academy platform, pen/pencil, worksheet.
Cross-curricular Topics: Mathematics (logic, parameters), Philosophy (decision making), Engineering.			
Assessment Criteria:			
1. Correctly formulates logical expressions. 2. Can predict the result of expressions.			

Prerequisite Knowledge: Has experience working with numerical operators.

SUBJECT: COMPUTER SCIENCE WITH PYTHON PROGRAMMING LANGUAGE		Grade: 10	Semester: 1
Level: Programming Fundamentals	Lesson hours: 6	Theoretical: 2	Practical: 4
Topic: Loops (8)			Chapter: 3

Objective		Learning Outcomes	
Apply for and while loops to solve multi-step problems		Upon completion of this topic, learners will be able to: 1. Apply loops (for, while) for fixed or conditional repetition 2. Write simple programs using range (), break, continue commands	
Content	Teaching Activities		Tools/Platforms
For, while loop syntax, range () function, break, continue commands for exiting loops	1. Demonstrate exercises with step-by-step explanations 2. Solve problems using different types of loops (for, while, break, continue)	Idle, Thonny, Profound Academy platform, pen/pencil, notebook	
Cross-curricular Topics: Physics (repetitive processes), Biology (cyclic processes), Data Processing Technology			
Assessment Criteria: 1. Can write a loop with specific conditions. 2. Can determine the number of loop iterations 3. Uses break/continue as needed			
Prerequisite Knowledge: Mastery of working with conditions and input data.			

SUBJECT: COMPUTER SCIENCE WITH PYTHON PROGRAMMING LANGUAGE		Grade: 10	Semester: 1
Level: Programming Fundamentals	Lesson hours: 6	Theoretical: 2	Practical: 4
Topic: Conditional Statements: if, if-else, if-elif (9)			Chapter: 3
Objective		Learning Outcomes	

Implement conditional structures (if, if-else, if-elif) for developing decision-based programs.		Upon completion of this topic, learners will be able to: 1. Construct if, if-else, and if-elif conditional structures. 2. Apply logical operators: and, or, not.
Content	Teaching Activities	Tools/Platforms
Conditional structures. Nested if statements. Syntax and error detection.	1. Discuss and compare examples. 2. Analyze code errors. 3. Solve problems using different types of if structures.	IDLE, Thonny, Profound Academy platform, pen/pencil, worksheet.
Cross-curricular Topics: Economics (conditions and scenarios), Geography (conditional analysis), Mathematics.		
Assessment Criteria: 1. Constructs correct if structure according to problem requirements. 2. Uses logical conditions for decision-making purposes. 3. Applies if-else and if-elif structures.		
Prerequisite Knowledge: Demonstrates competency in Python syntactic conventions and operators.		

SUBJECT: COMPUTER SCIENCE WITH PYTHON PROGRAMMING LANGUAGE		Grade: 10	Semester: 1
Level: Programming Fundamentals	Lesson hours: 6	Theoretical: 2	Practical: 4
Topic: Arrays list (10)			Chapter: 4
Objective	Learning Outcomes		
Apply one-dimensional and two-dimensional lists for data storage, editing, and analysis purposes	Upon completion of this topic, learners will be able to: 1. Create and index one-dimensional and two-dimensional lists. 2. Apply basic list methods: append, pop, insert, remove.		
Content	Teaching Activities	Tools/Platforms	
List creation, indexing, iterations, basic methods.	1. Examine and discuss examples. 2. Solve problems by constructing and transforming lists.	IDLE, Thonny, Profound Academy platform, pen/pencil.	

3. Perform exercises applying list methods.
Cross-curricular Topics: Data Science (data sequences), Biology (DNA chains), Mathematics (sets, vectors).
Assessment Criteria:
1. Correctly performs indexing. 2. Implements list element modifications. 3. Applies at least 2 methods.
Prerequisite Knowledge: Demonstrates competency in syntax, loops, and variables.

SUBJECT: COMPUTER SCIENCE WITH PYTHON PROGRAMMING LANGUAGE		Grade: 10	Semester: 1
Level: Programming Fundamentals	Lesson hours: 5	Theoretical: 2	Practical: 3
Topic: Tuple, set (11)			Chapter: 4
Objective		Learning Outcomes	
Differentiate between mutable and immutable collections and apply set structure for storing non-repeating values and efficient searching.		Upon completion of this topic, learners will be able to: 1. Explain and compare the properties of list, tuple, and set structures. 2. Apply appropriate structure based on problem objectives.	
Content	Teaching Activities		Tools/Platforms
Tuple – immutable element syntax. Set – exclusion of duplicates, adding elements (add), removing elements (remove, discard)	1. Present theoretical material and discuss examples. 2. Perform exercises on differentiating structures and their combined usage.		IDLE, Thonny, Profound Academy platform, pen/pencil.
Cross-curricular Topics: Information Technology (data sorting), Chemistry (molecular composition), Mathematics (set comparison).			
Assessment Criteria:			
1. Recognizes the structure. 2. Applies appropriate structure according to the problem.			
Prerequisite Knowledge: Demonstrates competency in lists and data types.			

SUBJECT: COMPUTER SCIENCE WITH PYTHON PROGRAMMING LANGUAGE	Grade: 10	Semester: 1
--	-----------	-------------

Level: Programming Fundamentals	Lesson hours: 6	Theoretical: 2	Practical: 4
Topic: Dictionaries (dict) (12)	Chapter: 4		

Objective		Learning Outcomes	
To teach the use of dictionary structure with key-value pairs for data organization, storage, and manipulation purposes		Upon completion of this topic, learners will be able to: 1. Create and modify dictionary structures. 2. Read data using keywords. 3. Apply methods: get (), keys (), values (). 4. Use complex structures: dictionary of lists, nested dictionaries.	
Content	Teaching Activities	Tools/Platforms	
Dictionary creation, reading values, adding and modifying. Basic methods (get, update, keys, values, items). Complex structures: nested dictionaries and dictionaries containing lists.	1. Discuss examples of different structures and methods. 2. Perform practical exercises with dictionaries for data analysis.	IDLE, Thonny, Profound Academy platform, pen/pencil.	
Cross-curricular Topics: Languages (construction of translation dictionaries), Data Science (key-value structures), Marketing (customer behavior modeling).			
Assessment Criteria: 1. Creates and uses simple dictionaries. 2. Accesses values. 3. Applies at least 2 methods. 4. Creates 2 complex dictionaries, using 1 in problem-solving.			
Prerequisite Knowledge: Demonstrates competency in lists and data structures.			

SUBJECT: COMPUTER SCIENCE WITH PYTHON PROGRAMMING LANGUAGE	Grade: 10	Semester: 2
Level: Intermediate Programming	Lesson hours: 9	Theoretical: 2 Practical: 7

Topic: Functions (13)		Chapter: 5	
Objective		Learning Outcomes	
Teach the concept, application, and purpose of functions in programming		Upon completion of this topic, learners will be able to: 1. Define a function. 2. Call a function with arguments. 3. Use return and parameters. 4. Create and apply functions with unlimited *args and **kwargs parameters.	
Content	Teaching Activities		Tools/Platforms
Function creation, calling, parameter passing, and function scope.	1. Explain through examples. 2. Discuss analytical questions. 3. Solve problems in pairs.		IDLE, Thonny, Profound Academy platform, pen/pencil.
Cross-curricular Topics: Mathematics (functions), Physics (dependent factors), Technical Design (modular approach).			
Assessment Criteria:			
1. Creates a function. 2. Applies at least 3 different types of functions.			
Prerequisite Knowledge: Demonstrates competency in variables, conditions, loops.			

SUBJECT: COMPUTER SCIENCE WITH PYTHON PROGRAMMING LANGUAGE		Grade: 10	Semester: 2
Level: Intermediate Programming	Lesson hours: 4	Theoretical: 1	Practical: 3
Topic: Files (14)			Chapter: 5
Objective		Learning Outcomes	
Teach file input/output mechanisms for data storage, processing, and exchange		Upon completion of this topic, learners will be able to: <ol style="list-style-type: none"> 1. Open and read files. 	

	2. Perform file writing operations. 3. Use with construction. 4. Handle errors.	
Content	Teaching Activities	Tools/Platforms
Basic file operations: opening, reading, writing, and closing files using open (), read (), write (), append (), with, file modes functions.	1. Present exercises. 2. Process files, analyze errors.	Thonny, IDLE, Terminal.
Cross-curricular Topics: Information Technology (file system), Business (file storage, documentation), Library Science.		
Assessment Criteria:		
1. Clearly reads and writes to files. 2. Controls file closing.		
Prerequisite Knowledge: Knows Python language input/output.		

SUBJECT: COMPUTER SCIENCE WITH PYTHON PROGRAMMING LANGUAGE		Grade: 10	Semester: 2
Level: Intermediate Programming	Lesson hours: 3	Theoretical: 0	Practical: 3
Topic: Module Introduction Colab (15)			Chapter: 5
Objective	Learning Outcomes		
Introduce basic data analysis tools (NumPy, Matplotlib)	Upon completion of this topic, learners will be able to: <ol style="list-style-type: none"> 1. Import and use modules. 2. Build arrays with NumPy. 3. Create graphs with Matplotlib. 4. Run programs in Google Colab and Jupyter Notebook environment. 		
Content	Teaching Activities	Tools/Platforms	
Module import, basics of arrays and	1. Demonstrate and explain with real examples.	Google Colab, Jupyter Notebook, NumPy,	

graphs, using Colab, Jupyter environment.	2. Implement educational mini projects.	Matplotlib.
Cross-curricular Topics: Mathematics (recursive definitions), Physics (induction), Design (pattern repetition).		
Assessment Criteria: 1. Applies modules purposefully. 2. Creates visualizations.		
Prerequisite Knowledge: Demonstrates competency in Python programming fundamentals and data structures.		

SUBJECT: COMPUTER SCIENCE WITH PYTHON PROGRAMMING LANGUAGE		Grade: 10	Semester: 2
Level: Intermediate Programming	Lesson hours: 10	Theoretical: 1	Practical: 9
Topic: Advanced Functions (16)			Chapter: 6
Objective		Learning Outcomes	
Teach advanced skills while using functions.		Upon completion of this topic, learners will be able to: 1. Use lambda, nested functions. 2. Differentiate between positional, default, keyword arguments. 3. Analyze code organization efficiency.	
Content	Teaching Activities		Tools/Platforms
Lambda, closure, decorators, and functional programming elements.	1. Perform exercises step-by-step and explain. 2. Solve problems in groups and pairs.	IDLE, Thonny, Profound Academy platform, pen/pencil.	
Cross-curricular Topics: Data Science, Software Engineering, Logic (abstraction, composition).			
Assessment Criteria: 1. Uses different arguments. 2. Creates modular functions.			
Prerequisite Knowledge: Knows basic functions and applies them in problems.			

SUBJECT: COMPUTER SCIENCE WITH PYTHON PROGRAMMING LANGUAGE	Grade: 10	Semester: 2
--	-----------	-------------

Level: Intermediate Programming		Lesson hours: 10	Theoretical: 1	Practical: 9
Topic: Recursion (17)				Chapter: 6
Objective		Learning Outcomes		
Introduce the principle of recursion, its applicability and limitations		Upon completion of this topic, learners will be able to: 1. Define recursion. 2. Write simple recursive functions. 3. Compare loops and recursion. 4. Use recursion in problem solving.		
Content	Teaching Activities		Tools/Platforms	
Principle of function calling itself, concept of recursive calls.	1. Explain verbally. 2. Demonstrate examples. 3. Perform exercises.		IDLE, Thonny, Profound Academy platform, pen/pencil.	
Cross-curricular Topics: Mathematics (Fibonacci, factorial), binary trees.				
Assessment Criteria:				
1. Analyzes recursive structure. 2. Applies in problem solving.				
Prerequisite Knowledge: Knowledge of advanced functions.				

SUBJECT: COMPUTER SCIENCE WITH PYTHON PROGRAMMING LANGUAGE		Grade: 11	Semester: 1
Level: Intermediate Programming	Lesson hours: 12	Theoretical: 2	Practical: 10
Topic: Class (18)			Chapter: 1
Objective		Learning Outcomes	
Introduce the student to object-oriented programming concepts through construction and use of classes and		Upon completion of this topic, learners will be able to: 1. Explain basic OOP concepts (class, object, attribute, method).	

objects	2. Build a class and create an object with attributes and methods. 3. Use classes for data modeling and problem solving. 4. Apply object-oriented thinking in simple problems.	
Content	Teaching Activities	Tools/Platforms
Class and object, init() method, attributes and methods, self-reference, data organization with classes.	1. Present theoretical-visual description and explain. 2. Perform Class code writing exercises. 3. Complete object-building assignments, analyze examples.	Thonny, IDLE, Jupyter, Google Colab, Profound Academy platform.
Cross-curricular Topics: Object modeling (engineering), Business (system management), Graphic Design (objects).		
Assessment Criteria:		
1. Creates classes, objects, and applies different methods.		
Prerequisite Knowledge: Knowledge of functions, lists, and dictionaries and experience in their application.		

SUBJECT: COMPUTER SCIENCE WITH PYTHON PROGRAMMING LANGUAGE		Grade: 11	Semester: 1
Level: Intermediate Programming	Lesson hours: 10	Theoretical: 2	Practical: 8
Topic: Inheritance (19)			Chapter: 1
Objective	Learning Outcomes		
Introduce the student to the concept of class inheritance, its application, and the importance of code reusability for extending program functionality.	Upon completion of this topic, learners will be able to: 1. Explain the basic concept of inheritance. 2. Create parent and child classes using inheritance. 3. Modify and extend inherited attributes and methods. 4. Analyze when it is appropriate to use inheritance in building multiple classes.		
Content	Teaching Activities	Tools/Platforms	

Class inheritance (class Child (Parent)), super methods (super ()), inheritance structure composition, code reusability principle.	<ol style="list-style-type: none"> 1. Perform method overriding assignments. 2. Conduct case and example analysis. 3. Perform individual exercises, group assignments. 	Thonny, IDLE, Jupyter, Google Colab, Profound Academy platform.
Cross-curricular Topics: Biology (genetic inheritance), IT (code reusability), Sociology (hereditary systems).		
Assessment Criteria:		
1. Creates class hierarchy. 2. Applies inheritance and adapts methods according to needs.		
Prerequisite Knowledge: Knows, applies class and object construction.		

SUBJECT: COMPUTER SCIENCE WITH PYTHON PROGRAMMING LANGUAGE		Grade: 11	Semester: 1
Level: Intermediate Programming	Lesson hours: 14	Theoretical: 2	Practical: 12
Topic: Modules and Repetition (20)			Chapter: 2
Objective		Learning Outcomes	
Introduce students to the principles of using Python modules and packages (NumPy, matplotlib, sklearn and built-in modules) for developing initial practical skills in scientific computing, data analysis, and artificial intelligence.		Upon completion of this topic, learners will be able to: <ol style="list-style-type: none"> 1. Explain the purpose of using modules and packages 2. Use simple built-in modules: math, random, time 3. Import and apply external modules: numpy, matplotlib, sklearn 4. Find and apply appropriate functions for solving specific problems 	
Content	Teaching Activities	Tools/Platforms	
Import and from...import structures. Numpy arrays, matplotlib graphs, sklearn functions. Common functions: np.array(), plt.plot(), train_test_split()	<ol style="list-style-type: none"> 1. Present interactive demonstration on importing and using modules 2. Perform step-by-step exercises and demonstrations 	Thonny, IDLE, Jupyter, Google Colab, Profound Academy platform, Numpy, Matplotlib, Sklearn.	

	3. Conduct group work for data analysis	
Cross-curricular Topics: Data Science (analytical tools), Economics (model analysis)		
Assessment Criteria:		
1. Correctly imports simple (built-in) modules 2. Correctly imports external modules 3. Selects appropriate functions and applies them in different problems.		
Prerequisite Knowledge: Knows variables, conditions, lists, and loops		

SUBJECT: COMPUTER SCIENCE WITH PYTHON PROGRAMMING LANGUAGE		Grade: 11	Semester: 2
Level: Intermediate Programming	Lesson hours: 7	Theoretical: 1	Practical: 6
Topic: Libraries and Environments (21)			Chapter: 3
Objective	Learning Outcomes		
Introduce students to the integration and application possibilities of libraries in various environments, with a focus on Artificial Intelligence (AI) applications	<p>Upon completion of this topic, learners will be able to:</p> <ol style="list-style-type: none"> 1. Distinguish main AI/ML cloud environments (Kaggle, Anaconda, Docker, Hugging Face). 2. Use the pandas, pytorch, and tensorflow libraries. 3. Create and manage Python environments using Anaconda or venv, including tools such as numpy, matplotlib, pandas, sklearn, pytorch, tensorflow, pip, and conda. 		
Content	Teaching Activities	Tools/Platforms	
Types of cloud environments: Kaggle, Google Colab, Anaconda, Docker, Hugging Face. Library management tools: pip, conda, venv. Libraries: numpy, matplotlib, pandas, sklearn, pytorch, tensorflow.	<ol style="list-style-type: none"> 1. Create a Notebook in Kaggle or Colab. 2. Install libraries using Anaconda or venv. 3. Perform test AI data analysis using the libraries. 	Thonny, IDLE, Jupyter, Google Colab, Terminal, Kaggle, pip, conda, Anaconda, venv, pandas, pytorch, tensorflow.	

Cross-curricular Topics: Artificial Intelligence, Data Science, Cloud Technology Environments.				
Assessment Criteria: 1. Sets up and organizes virtual environments (venv, Anaconda) and libraries. 2. Creates an error-free working notebook in Kaggle or Colab using at least two libraries. 3. Performs complete data analysis (data loading, filtering, visualization plotting, basic interpretation).				
Prerequisite Knowledge: Has intermediate Python knowledge and basic experience using core libraries.				
SUBJECT: COMPUTER SCIENCE WITH PYTHON PROGRAMMING LANGUAGE			Grade: 11	Semester: 2
Level: Intermediate Programming		Lesson hours: 5	Theoretical: 0	Practical: 4
Topic: Debugging and Error Resolution (22)				Chapter: 5
Objective		Learning Outcomes		
Develop students' ability to systematically analyse, identify, and resolve programming errors through appropriate classification and application of debugging techniques and error-handling mechanisms.		Upon completion of this topic, learners will be able to: 1. Distinguish between syntax, runtime, logical, and user-defined errors. 2. Implement robust error handling using try / except / finally structures. 3. Design and raise custom exceptions in response to specific problem requirements. 4. Utilize breakpoints, tracebacks, and debugging tools to detect and resolve errors efficiently.		
Content		Teaching Activities		Tools/Platforms
Types of errors: SyntaxError, NameError, IndexError, ZeroDivisionError, ValueError. Error-handling structures: try / except / finally, custom exceptions. Debugging tools: pdb module, breakpoint () function, IDE debuggers.		1. Practise step-by-step debugging techniques using various IDEs (IDLE, Thonny, VS Code). 2. Analyse and discuss real-world examples containing errors. 3. Design and implement clear, meaningful custom exceptions. 4. Conduct peer reviews and group discussions on debugging faulty code snippets.		Thonny, IDLE, Jupyter, Google Colab, VS Code Debugger, Python pdb module, Terminal.
Cross-curricular Topics: Computing (debugging techniques); Mathematics (error analysis); Engineering (fault prevention).				

Assessment Criteria:

1. Accurately classifies and explains various error types. 2. Correctly implements try / except structures according to error type. 3. Creates and communicates custom exceptions appropriately. 4. Demonstrates clear application and explanation of the debugging process across different examples.

Prerequisite Knowledge: Proficiency in working with conditions, loops, functions, and lists.

SUBJECT: COMPUTER SCIENCE WITH PYTHON PROGRAMMING LANGUAGE		Grade: 11	Semester: 2
Level: Intermediate Programming	Lesson hours: 2	Theoretical: 0	Practical: 2
Topic: Version Control and Dashboard Usage (Git, Terminal) (23)			Chapter: 4
Objective	Learning Outcomes		
Strengthen students' ability to analyse, present, and manage their own projects by effectively employing collaborative tools and practices (Git).	Upon completion of this topic, students will be able to: 1. Present their project with clear and logical structure. 2. Use Git for effective version control. 3. Apply basic terminal commands (cd, mkdir, git init, git commit) for project management.		
Content	Teaching Activities	Tools/Platforms	
Fundamentals of Git and version control. Managing collaborative projects and integrating GitHub. Peer review and feedback processes.	1. Create a Git repository and publish it on GitHub. 2. Practise the Git workflow (init, add, commit, push). 3. Perform version control exercises using terminal commands. 4. Engage in team discussions and peer reviews.	Thonny, IDLE, Jupyter, Google Colab, VS Code, Terminal (bash/zsh), Git CLI, GitHub Desktop.	
Cross-curricular Topics: Computing (project management); Organizational Culture (collaborative work); Engineering (doc. control).			
Assessment Criteria:			

1. Presents project content logically and coherently. 2. Communicates ideas clearly and provides constructive feedback. 3. Demonstrates self-reflection through structured evaluation of their own work.

Prerequisite Knowledge: Capable of performing data analysis, familiar with Python libraries, understands Git concepts, and elementary experience with the terminal.

SUBJECT: COMPUTER SCIENCE WITH PYTHON PROGRAMMING LANGUAGE		Grade: 11	Semester: 2
Level: Intermediate Programming	Lesson hours: 4	Theoretical: 2	Practical: 2
Topic: Summary and Reflection (24)			Chapter: 4
Objective	Learning Outcomes		
Support students in consolidating the material covered, articulating their developed skills, and participating effectively in technical discussions.	<p>Upon completion of this topic, students will be able to:</p> <ol style="list-style-type: none"> 1. Analyse the key topics studied and connect them to real-world problems. 2. Present the structure and core solutions of their own project. 3. Provide constructive feedback on the projects of peers. 4. Carry out self-assessment and identify areas for further learning and development. 		
Content	Teaching Activities	Tools/Platforms	
<p>Brief review of key topics: data types, functions, loops, libraries, error handling.</p> <p>Oral project presentation. Team-based</p>	Deliver a presentation, take part in discussions, and conduct self-reflection.	Google Slides, oral presentations (or on paper), reflection matrices.	

reflection on problem-solving experiences.	Prepare a short pitch covering "My Solutions", "My Challenges", and "My Next Steps".	
Cross-curricular Topics: Social Sciences (discussion skills); Communication (presentation skills, self-assessment, project-based and team collaboration).		
Assessment Criteria: 1. Presents content clearly and coherently. 2. Articulates ideas effectively and responds constructively. 3. Performs self-assessment with clear criteria.		
Prerequisite Knowledge: Has a sound understanding of project finalization, functions, and libraries		

SUBJECT DETAILED CURRICULUM

ARTIFICIAL INTELLIGENCE

SUBJECT: ARTIFICIAL INTELLIGENCE		Grade: 10	Semester: 1
Level: AI Experience	Lesson hours: 3	Theoretical: 2	Practical: 1
Topic : AI Literacy and Ethics (I)			Chapter: 1

Objective	Learning Outcomes
Enable students to explore AI tools, understand their ethical implications, and recognize the significance of data in AI systems	Upon completion of this topic, learners will be able to: <ol style="list-style-type: none">1. Identify and explain AI tools used in everyday life.2. Detect and describe examples of bias in machine learning models.3. Propose methods to improve fairness and effectiveness in AI systems.4. Analyse shortcomings in datasets.

Content	Teaching Activities	Tools/Platforms
Fundamental principles of AI. Types of data bias. Approaches to ethical AI development	1. Carry out practical demonstrations and exercises using AI tools. 2. Facilitate group debates on ethical dilemmas in AI. 3. Analyse real-life case studies and biased algorithms.	Videos, slide presentations, sample datasets, data bias visualization tools.
Cross-curricular Topics: Sociology (impact of technology on society); Data Science (bias detection and analysis).		
Assessment Criteria: 1. Applies AI tools meaningful and appropriately. 2. Effectively analyses and discusses ethical dilemmas. 3. Actively participates in group discussions, sharing relevant examples of bias.		
Prerequisite Knowledge: Demonstrates algorithmic thinking and information literacy.		

SUBJECT: ARTIFICIAL INTELLIGENCE		Grade: 10	Semester: 1
Level: AI Experience	Lesson hours: 8	Theoretical: 5	Practical: 3
Topic: Introduction to AI and Related Technologies (2)			Chapter: 1

Objective	Learning Outcomes
Enable students to explore the fundamental principles of Artificial Intelligence and its applications within modern technologies.	Upon completion of this topic, learners will be able to: 1. Distinguish between the concepts of Artificial Intelligence (AI) and Machine Learning (ML). 2. Explain the role of Machine Learning and Deep Learning (DL) as subfields of AI. 3. Describe examples of simple ML models, particularly for data classification tasks.

Content	Teaching Activities	Tools/Platforms
Understanding 'Intelligence': human vs. machine. Definitions of AI and its everyday applications and limitations. Narrow AI vs. General AI. AI subfields and current examples. Demonstrations and practical illustrations.	<ol style="list-style-type: none"> 1. Attend theoretical lessons and participate in class discussions. 2. Analyse and reflect on videos illustrating AI concepts. 3. Undertake small group projects applying basic ML tools. 	Videos, slide presentations, Google Teachable Machine.
Cross-curricular Topics: Computer Science (algorithms, basic programming); Mathematics (fundamentals of data visualization).		
Assessment Criteria: 1. Presents a simple ML model created with Teachable Machine. 2. Accurately differentiates types of AI and explains their applications.		
Prerequisite Knowledge: Possesses basic computer skills and logical thinking abilities.		

SUBJECT: ARTIFICIAL INTELLIGENCE		Grade: 10	Semester: 1
Level: AI Experience	Lesson hours: 5	Theoretical: 3	Practical: 2
Topic: AI Careers, Future Disciplines and Professions (3)			Chapter: 2

Objective	Learning Outcomes
Help students explore career prospects in the field of Artificial Intelligence.	Upon completion of this topic, learners will be able to: <ol style="list-style-type: none"> 1. Describe the core responsibilities of AI professionals (e.g. data engineering). 2. Plan their own learning pathway within the AI sector. 3. Identify how AI is applied at both local and global levels.

Content	Teaching Activities	Tools/Platforms
History and applications of AI across various sectors. Emerging future professions. Visual tools and posters.	<ol style="list-style-type: none"> 1. Participate in discussions with AI professionals. 2. Take part in matching games that connect professions to AI tools. 3. Create an "My AI Journey" poster. 	AI profession cards; career pathway research using LinkedIn/Glassdoor; examples of local and international AI start-ups.
Cross-curricular Topics: Economics: labor market trends; Geography: understanding international opportunities and best practices.		
Assessment Criteria: 1. Presents a clear "My AI Journey" poster. 2. Completes an AI career quiz or matching activity.		
Prerequisite Knowledge: Demonstrates basic research skills and experience with project-based learning.		

SUBJECT: ARTIFICIAL INTELLIGENCE		Grade: 10	Semester: 2
Level: Elements of Machine Learning (Supervised Learning)	Lesson hours: 6	Theoretical: 3	Practical: 3
Topic: Perception (4)			Chapter: 3

Objective	Learning Outcomes
Introduce students to the concept of perception in AI, distinguishing it clearly from sensation, and to explore how textual, audio, and visual data are digitally represented within	<p>Upon completion of this topic, learners will be able to:</p> <ol style="list-style-type: none"> 1. Explain the distinction between human sensory perception (e.g., vision, hearing) and machine perception (e.g., sensors, programming). 2. Describe key challenges in machine perception (e.g., noise, data error) and the role of

computer systems.	abstraction in simplifying and analyzing data. 3. Identify and describe various data representation formats: images (RGB), audio (waveforms), and text (ASCII encoding).	
Content	Teaching Activities	Tools/Platforms
Human vs. machine perception. Sensation vs. Perception. Examples of machine sensing (visual, auditory, localization, thermal, motion, distance). Representation across modalities (image, sound, text).	1. Explore human and machine examples of perception using visual/audio input and sensor-based data. 2. Present different modalities and formats of data representation.	Visual materials (image, sound, text samples); simplified data demos (RGB channels, sound waves, ASCII); discussion charts and activity sheets.
Cross-curricular Topics: Biology (sensory systems); Physics (light, sound); Computer Science (data formats); Electronics (sensors).		
Assessment Criteria: 1. Distinguishes between sensation and perception. 2. Describes different modes of machine perception. 3. Explains basic principles of digital representation (RGB, waveforms, ASCII). 4. Presents perception challenges with concrete examples.		
Prerequisite Knowledge: Understands basic human sensory functions; knows data units (bit, byte); has foundational understanding of light and sound waves.		

SUBJECT: ARTIFICIAL INTELLIGENCE		Grade: 10	Semester: 2
Level: Elements of Machine Learning (Supervised Learning)	Lesson hours: 11	Theoretical: 6	Practical: 5

Objective	Learning Outcomes	
Provide students with a general understanding of machine learning and supervised learning, highlighting key methods, applications, and common challenges.	<p>Upon completion of this topic, learners will be able to:</p> <ol style="list-style-type: none"> 1. Explain the principles of machine learning and supervised learning, their differences, and areas of application. 2. Describe linear classification and the k-NN (k-nearest neighbors) method and outline more complex classification tasks. 3. Understand the concepts of underfitting and overfitting and explain their impact on models. 4. Implement a simple k-NN classifier in Python using appropriate libraries and test it on a small dataset. 	
Content	Teaching Activities	Tools/Platforms
Introduction to data-driven learning (machine learning). Supervised learning for classification problems. Training data and feature representation. Rule-based vs. linear classification examples. k-nearest neighbors' classification. Model selection challenges.	<ol style="list-style-type: none"> 1. Compare traditional and machine learning approaches using tables and examples. 2. Develop visual representations of features, labels, and classification processes. 3. Analyse and construct simple decision trees. 4. Perform linear classification and identify decision boundaries. 5. Implement a k-NN classifier, covering both theoretical and practical aspects. 6. Test prepared models in Python or Colab on small datasets. 	Python, Jupyter Notebook or Google Colab, Scikit-learn, NumPy, Matplotlib, Teachable Machine; printed charts, cards, sample datasets (e.g. Iris, basic weather data).

Cross-curricular Topics: Mathematics (distance metrics, linear equations); Data Science and Statistics; Computer Science (Python programming, algorithms); Biology (Iris dataset); Physics (data modelling concepts).

Assessment Criteria:

1. Explains the essence of supervised learning and classification. 2. Identifies different classification algorithms and their contexts. 3. Understands and explains underfitting and overfitting phenomena. 4. Implements k-NN in Python, describes, tests, and compares the results.

Prerequisite Knowledge: Has basic programming skills (Python, variables, functions); understands fundamental mathematical concepts (points, perpendicular bisectors, means); can present and perform basic data analysis (CSV tables, labelling); has experience creating tree diagrams (classification logic).

SUBJECT: ARTIFICIAL INTELLIGENCE		Grade: 10	Semester: 2
Level: Elements of Machine Learning (Supervised Learning)	Lesson hours: 19	Theoretical: 10	Practical: 9
Topic: ML Components: Data (6)			Chapter: 5

Objective	Learning Outcomes
Introduce students to the fundamental steps of data representation, analysis, and preprocessing for supervised machine learning.	<p>Upon completion of this topic, learners will be able to:</p> <ol style="list-style-type: none"> 1. Apply basic programming techniques for data visualization (scatter plots, line graphs, tabular and other diagram types) using Matplotlib and NumPy. 2. Explain what data preprocessing entails and understand its steps (normalization, data augmentation, feature engineering). 3. Describe the roles of training, validation, and testing datasets in the data pipeline and model development.

	4. Apply data preprocessing steps in Python to prepare small datasets using NumPy or other libraries.	
Content	Teaching Activities	Tools/Platforms
Data representation with vectors and matrices. Basic vector operations using NumPy. Descriptive statistics and visualization with Matplotlib. Data preprocessing: scaling, normalization, and data splitting. Cross-validation for model evaluation. Underfitting and overfitting causes and prevention.	<ol style="list-style-type: none"> 1. Identify and collect examples of different data types. 2. Present data in tables, arrays, and vectors. 3. Identify features and labels in practical examples. 4. Perform statistical analysis: mean, median, standard deviation. 5. Implement data preprocessing: cleaning, scaling, encoding. 6. Split data for training, validation, and testing. 7. Create data visualizations: 1D/2D plots, boxplots. 8. Practice using Python (NumPy, Matplotlib) in Google Colab or Jupyter Notebook. 	Google Colab, Jupyter Notebook, Python (NumPy, Matplotlib, pandas), Excel, Google Sheets; online visualization tools (Plotly, Desmos); data presentation templates (recording tables, boxplot templates).
Cross-curricular Topics: Mathematics (statistics, vectors); Technology (use of digital data); Biology/Science (measurement data); Data Science (theory and applications).		
Assessment Criteria:		
1. Explains the essence of supervised learning and classification. 2. Identifies and contextualizes different classification algorithms. 3. Understands and describes underfitting and overfitting phenomena. 4. Applies k-NN in Python, interprets, and compares results.		
Prerequisite Knowledge: Has basic knowledge of Python programming, vectors, and arrays; understands core statistical concepts (mean, deviation); possesses elementary computational thinking skills.		

SUBJECT: ARTIFICIAL INTELLIGENCE		Grade: 11	Semester: 1
Level: Machine Learning Algorithms	Lesson hours: 5	Theoretical: 2	Practical: 3
Topic: ML Methods: Decision Trees (7)			Chapter: 1

Objective		Learning Outcomes	
Explore the decision tree method as a simple and interpretable supervised learning algorithm, understanding its construction principles and how to mitigate overfitting.	Upon successful completion of this topic, learners will be able to: 1. Explain the concept of decision trees and their application in classification problems. 2. Construct a simple decision tree in Python using basic splitting rules. 3. Describe the relationship between tree depth and overfitting. 4. Analyse splitting criteria, impurity measures (e.g. Gini impurity), and principles of early stopping and pruning.		
Content	Teaching Activities	Tools/Platforms	
Decision trees as a classification algorithm. Data splitting based on features. Splitting criteria and impurity measures (e.g. Gini impurity). Effect of tree depth on model performance. Overfitting and generalization challenges. Early stopping and pruning techniques.	1. Build decision trees manually and implement them in Python. 2. Practice selecting and analyzing splitting rules. 3. Evaluate models through practical exercises, Colab testing, and group discussions.	Python, NumPy, Matplotlib, Scikit-learn, Jupyter Notebook, Google Colab, printed data tables.	
Cross-curricular Topics: Mathematics (logical reasoning, data analysis); Computer Science (algorithms, coding, data structures); Data Science (classification tasks, model building); Medicine/Finance (diagnostic and risk decision trees).			

Assessment Criteria:

1. Creates a decision tree model in Python and visualizes at least one branching solution. 2. Correctly calculates Gini impurity for at least two split options and selects the optimal split. 3. Compares accuracy of models with varying depths and explains differences. 4. Applies pruning or early stopping within defined limits and interprets results.

Prerequisite Knowledge: Able to represent data in terms of features and labels; has basic Python programming skills; understands basic decision logic.

SUBJECT: ARTIFICIAL INTELLIGENCE		Grade: 11	Semester: 1
Level: Machine Learning Algorithms	Lesson hours: 18	Theoretical: 6	Practical: 12
Topic: ML Methods: Linear and Logistic Regression (8)			Chapter: 2

Objective		Learning Outcomes	
Introduce learners to linear methods for regression and classification, and to build conceptual understanding of how parametric approaches function in practice.		Upon successful completion of this topic, learners will be able to: <ol style="list-style-type: none"> 1. Distinguish clearly between regression and classification tasks. 2. Explain the role of a loss function, specifically Mean Squared Error (MSE). 3. Describe the principles underlying linear and logistic regression. 4. Implement a simple linear or logistic model in Python and test it on a dataset. 	
Content	Teaching Activities	Tools/Platforms	
Linear vs. non-linear functions. Regression and classification: definitions and	Discuss the differences between linear and non-linear functions.	Python, Matplotlib, NumPy, Scikit-learn, Jupyter Notebook, Google Colab.	

<p>differences. Building and evaluating linear models. Loss functions: MSE, cross-entropy; the role of gradient descent. Logistic regression algorithm.</p> <p>- Practical implementation with sklearn and NumPy.</p>	<p>2. Examine and plot different function graphs.</p> <p>3. Calculate MSE in Colab on sample datasets.</p> <p>4. Build linear and logistic models using sklearn.</p> <p>5. Create simple simulations with variable data.</p>	
<p>Cross-curricular Topics: Mathematics (derivatives, squared error, gradient descent); Physics (energy minimization models, slopes and gradients); Data Science (functional modelling, loss analysis).</p>		
<p>Assessment Criteria</p> <p>1. Accurately distinguishes between regression and classification tasks.2. Analyses how the loss function affects model quality.3. Implements linear/logistic models using clear logical steps.4. Explains causes of overfitting and underfitting in the context of their models.</p>		
<p>Prerequisite Knowledge: Masters Python fundamentals (variables, lists, functions, loops), applies NumPy arrays, matplotlib graphics, sklearn module.</p>		

SUBJECT: ARTIFICIAL INTELLIGENCE		Grade: 11	Semester: 1
Level: Machine Learning Algorithms	Lesson hours: 17	Theoretical: 5	Practical: 12
Topic: ML Components: Evaluation (9)			Chapter: 3
Objective		Learning Outcomes	

Familiarize learners with core methods and metrics for evaluating machine learning models, while emphasizing the contextual and ethical implications of metric choice and interpretation.	Upon successful completion of this topic, learners will be able to: <ol style="list-style-type: none"> 1. Construct and interpret a confusion matrix. 2. Calculate and explain key evaluation metrics: accuracy, precision, recall, and F1-score. 3. Apply evaluation techniques using the sklearn library in Python. 4. Identify possible sources of evaluation errors and propose realistic improvements. 	
Content	Teaching Activities	Tools/Platforms
Building and interpreting a confusion matrix (TP, TN, FP, FN). Mathematical basis and use cases of accuracy, precision, recall, and F1-score. Analysis of evaluation errors and biases. Hands-on practice with sklearn.metrics in Python.	<ol style="list-style-type: none"> 1. Build a confusion matrix manually and interpret results for given datasets. 2. Compute evaluation metrics using sklearn. 3. Analyse potential evaluation errors and biases and discuss improvement strategies. 4. Reflect on real-world ethical considerations when interpreting metrics. 	Jupyter Notebook, Google Colab (Evaluation_Practice.ipynb), Python, sklearn.metrics module, simple labelled datasets in matrix form.
Cross-curricular Topics: Biology (medical diagnosis models); Mathematics (probabilities, percentages); ICT (use of computational evaluation tools); Civics/Ethics (discussion of bias and consequences in automated decisions).		
Assessment Criteria: <ol style="list-style-type: none"> 1. Correctly differentiates TP, TN, FP, FN cases in examples. 2. Accurately calculates and interprets the four key metrics. 3. Critically assesses a model's limitations across scenarios. 4. Applies evaluation tools effectively, justifying choices with clear indicators. 		
Prerequisite Knowledge: Solid understanding of basic classification tasks; mathematical literacy in calculating ratios and percentages; foundational Python skills (lists, functions); initial experience in building simple models and handling data in sklearn.		

SUBJECT: ARTIFICIAL INTELLIGENCE		Grade: 11	Semester: 1
Level: Machine Learning Algorithms	Lesson hours: 6	Theoretical: 3	Practical: 3
Topic: Introduction to Advanced Topics (10)			Chapter: 4

Objective	Learning Outcomes	
Introduce learners to the key concepts of bias, variance, regularization, and the curse of dimensionality as essential foundations for evaluating and improving machine learning models.	<p>Upon successful completion of this topic, learners will be able to:</p> <ol style="list-style-type: none"> 1. Explain how bias and variance affect model performance and interpret the trade-off. 2. Apply basic regularization methods (L1, L2, pruning, early stopping) to reduce overfitting. 3. Describe the 'curse of dimensionality' and implement basic dimensionality reduction techniques. 4. Integrate regularization in linear models using sklearn. 	
Content	Teaching Activities	Tools/Platforms
Understanding bias and variance and their trade-off. Overfitting vs underfitting review. Regularization techniques: L1 (Lasso), L2 (Ridge), tree pruning, early stopping. Curse of dimensionality: challenges in high-dimensional spaces. Dimensionality reduction using PCA and t-SNE. Implementing and comparing models in practice.	<ol style="list-style-type: none"> 1. Explore bias and variance patterns with guided examples and visualizations. 2. Conduct numerical exercises calculating MSE, bias, and variance. 3. Compare models of varying complexity; interpret regularization impact. 4. Apply PCA and t-SNE for low-dimensional representation of sample datasets. 	Google Colab, Jupyter, Python (sklearn for Ridge, Lasso, PCA, t-SNE), matplotlib/seaborn for visualization, worksheets for guided exercises.
Cross-curricular Topics: Mathematics (sums of squares, distance, vectors); Computer Science (algorithms, classifier comparison);		

Physics/Natural Sciences (sources of measurement error, variance); Engineering/Data Science (simplification of measurements, complexity control).

Assessment Criteria:

1. Correctly identifies bias and variance in sample scenarios. 2. Selects and applies appropriate regularization method for a given modelling issue. 3. Implements PCA or t-SNE on small datasets and explains results. 4. Calculates MSE, bias and variance, drawing reasoned conclusions from the values.

Prerequisite Knowledge: Basic grasp of linear model operations; prior practice with sklearn basics (model.fit, model.predict); ability to represent data as vectors and matrices; intuitive sense of multi-dimensional space, without advanced mathematical formalism

SUBJECT: ARTIFICIAL INTELLIGENCE		Grade: 11	Semester: 2
Level: Neural Networks	Lesson hours: 10	Theoretical: 6	Practical: 4
Topic: Introduction to Neural Networks (II)			Chapter: 1

Objective	Learning Outcomes
Develop a foundational understanding of the principles, structure, and practical applications of neural networks, bridging traditional machine learning with deep learning methods.	<p>Upon successful completion of this topic, learners will be able to:</p> <ol style="list-style-type: none"> 1. Recap key machine learning concepts and relate them to neural networks. 2. Describe the biological inspiration behind neural networks. 3. Explain the role of the perceptron and activation functions. 4. Identify key structural components of deep neural networks. 5. Implement and run a simple neural network using Python. 6. Distinguish between CNNs, RNNs, and Transformers and describe their typical use cases.

Content	Teaching Activities	Tools/Platforms
Biological neurons and the inspiration for artificial neurons. The perceptron model and its mathematical basis. Activation functions (e.g., Sigmoid, ReLU, Softmax). Basic structure of deep neural networks (layers, weights, biases). Overview of CNNs, RNNs, and Transformers: when and why to use them. Simple NN implementation in Python	<ol style="list-style-type: none"> 1. Compare biological and artificial neurons through diagrams and discussion. 2. Build a perceptron model step-by-step. Explore activation functions visually and mathematically. 3. Use NN Playground for interactive experimentation. 4. Implement and test a simple neural network in Colab using TensorFlow or PyTorch basics. 	NN Playground, Google Colab/Jupyter Notebook, Python libraries (NumPy, TensorFlow or PyTorch), sample datasets (e.g., MNIST subsets).
Cross-curricular Topics: Biology (neurons, nervous systems); Mathematics (vectors, matrix operations, functions); Computer Science (algorithms, training loops); Physics (sensor data); Medicine/Health (medical imaging with neural networks).		
Assessment Criteria: 1. Clearly explains the perceptron structure and the role of weights and bias. 2. Demonstrates correct application of activation functions in a simple model. 3. Builds and runs a working simple neural network in Python. 4. Identifies and distinguishes CNN, RNN, and Transformer architectures and matches them to suitable tasks.		
Prerequisite Knowledge: Solid understanding of core machine learning concepts, Python intermediate programming skills, familiarity with vectors and basic linear algebra.		

SUBJECT: ARTIFICIAL INTELLIGENCE

Grade: 11

Semester: 2

Level: Neural Networks		Lesson hours: 7	Theoretical: 4	Practical: 3
Topic: Backpropagation (12)				Chapter: 2
Objective		Learning Outcomes		
Introduce and deepen understanding of two key components in training neural networks: loss functions and the gradient descent process, making the concepts of learning rate, gradient flow, and backpropagation accessible through theory and practice.		<p>Upon successful completion of this topic, learners will be able to:</p> <ol style="list-style-type: none"> 1. Define and explain the role of a loss function in training neural networks. 2. Apply MSE and Cross-Entropy functions in simple examples. 3. Explain the concept of gradient descent theoretically and visually. 4. Build a simple neural network in Python, using gradient descent and backpropagation. 5. Analyse the impact of learning rate and select appropriate values. 6. Describe the weight update process and the role of gradients in each training cycle. 		
Content	Teaching Activities	Tools/Platforms		
<p>Loss functions: MSE, MAE, Cross-Entropy (CE), Binary Cross-Entropy (BCE). Concept and math of gradient descent.</p> <p>Backpropagation algorithm: chain rule and weight updates. Learning rate, local minima and saddle points. Visual metaphors: 'ball rolling down a hill', 'mountain climber'. Simple neural network implementation with step-by-step backpropagation.</p>	<ol style="list-style-type: none"> 1. Interactive discussion on loss function goals, comparing different types (MSE vs. CE). 2. Pair activity: manual MSE computation with toy data. 3. Analyse real examples with BCE/CCE and interpret results. 4. Code a simple neural network from scratch in Python, demonstrate weight updates with backpropagation. 	<p>NN Playground</p> <p>Google Colab/Jupyter Notebook ("Backpropagation Step-by-Step").</p> <p>Python (NumPy, Matplotlib).</p> <p>Printable worksheets for manual calculations.</p>		

	5. Visualize gradient descent steps and test the impact of learning rate.	
Cross-curricular Topics: Mathematics (derivatives, quadratic error, gradients); Physics (energy minimization, slopes, forces); Data Science (loss function design, evaluation metrics).		
Assessment Criteria: 1. Correctly explains the function and significance of loss functions in NNs. 2. Accurately computes MSE, MAE, BCE, CCE for simple examples. 3. Builds a functional Python NN model that runs backpropagation correctly. 4. Demonstrates reduction of loss using test data, showing understanding of learning rate impact		
Prerequisite Knowledge: Understanding of perceptron structure and limitations; familiarity with activation functions (Sigmoid, Softmax); working knowledge of Python; ability to apply basic MSE/CE formulas.		

SUBJECT: ARTIFICIAL INTELLIGENCE		Grade: 11	Semester: 2
Level: Neural Networks	Lesson hours: 14	Theoretical: 6	Practical: 8
Topic: Practical Training of Neural Networks (13)			Chapter: 3

Objective	Learning Outcomes
Build, configure and train neural networks in practice using TensorFlow/Keras or PyTorch libraries, understanding the role of neurons, layers and hyperparameters.	Upon successful completion of this topic, learners will be able to: 1. Build neural networks with dense and flattened layers (TensorFlow / PyTorch). 2. Tune hyperparameters and evaluate model quality (accuracy, loss). 3. Apply regularization (L1, L2, Dropout) and normalization.

	4. Build training and validation pipelines. 5. Build multi-layer NNs for different tasks (digit/image classification). 6. Analyse and select optimal architecture (shallow/deep).	
Content	Teaching Activities	Tools/Platforms
Downloading and displaying MNIST digit data. Data preprocessing (normalize). Building layers with nn.Linear, ReLU, Softmax. Loss function: CrossEntropyLoss. Optimizer: SGD, Adam. Training loop: loss reduction, accuracy tracking. Model architecture design and testing. Mention of GPU acceleration concept.	1. Perform flattened image reshaping. 2. Build a Torch model by inheriting torch.nn.Module. 3. Implement reading of loss and accuracy plots. 4. Modify hyperparameters, e.g., learning rate, batch size. 5. Do final testing on new data (predict).	PyTorch, Google Colab, torchvision.datasets (MNIST), matplotlib.
Cross-curricular Topics: Mathematics (linear combination, vectors). Data Science (classification tasks). Artificial Intelligence (image recognition). Engineering (system acceleration, GPU).		
Assessment Criteria: Builds a working multi-layer neural network (MLP). Correctly applies loss and optimizer functions. Recognizes data structure and maps it to the torch model. Analyses the effect of architecture changes on accuracy.		
Prerequisite Knowledge: Has deep knowledge of Python programming, experience with data input/output (I/O), understands the construction and dimensions of Torch tensors.		

SUBJECT: ARTIFICIAL INTELLIGENCE		Grade: 11	Semester: 2
Level: Neural Networks	Lesson hours: 7	Theoretical: 3	Practical: 4
Topic: Convolutional Neural Networks, CNNs (14)			Chapter: 4

Objective		Learning Outcomes	
Provide a general understanding of the function of convolutional layers and their role in image-related tasks.	Upon successful completion of this topic, learners will be able to: 1. Explain the structure and role of convolutional layers in image analysis. 2. Build and train a simple CNN model. 3. Compare CNN models with fully connected models.		
Content	Teaching Activities	Tools/Platforms	
2D convolution principles, zero padding, stride, filters, and image processing concepts. Pooling methods and CNN architectures. Overview of the VGG-16 model. Practical implementation of image classification using CNNs.	1. Perform manual 2D convolutions on matrices. 2. Explore the impact of different filters on images. 3. Implement pooling and normalization steps. 4. Build and train a CNN model using Python. 5. Apply a pre-trained VGG-16 model to a simple classification task.	Python, PyTorch / TensorFlow, NumPy, Matplotlib, scikit-learn.	
Cross-curricular Topics: 1. Design (image visualization and analysis). 2. Technology (image processing systems). 3. Biomedical applications (medical image classification). 4. Physics (filter-based algorithm modeling).			
Assessment Criteria: 1. Accurately defines concepts such as padding, strides, and pooling. 2. Can manually apply convolution on small matrices. 3. Correctly implements CNN-based classification on simple data. 4. Compares CNN results with a basic MLP.			

Prerequisite Knowledge: Understands basic neural network principles. Familiar with matrix operations (dot product) and activation functions.

SUBJECT: ARTIFICIAL INTELLIGENCE		Grade: 11	Semester: 2
Level: Neural Networks	Lesson hours: 5	Theoretical: 2	Practical: 3
Topic: Transfer Learning (15)			Chapter: 5

Objective		Learning Outcomes	
Introduce students to the concept and stages of transfer learning, explaining how pre-trained models can be applied when data is limited.		<p>Upon successful completion of this topic, learners will be able to:</p> <ol style="list-style-type: none"> 1. Explain the idea of transfer learning and why it is needed. 2. Define source and target domains and understand layer freezing and fine-tuning. 3. Implement transfer learning using VGG-16 or another pre-trained model with data augmentation. 4. Compare the performance of models trained from scratch vs. pre-trained models on simple tasks. 	
Content	Teaching Activities	Tools/Platforms	
Transfer learning concepts, source and target domains. Freezing layers, trainable layers, data augmentation, gender classification Colab example, popular CNN models: VGG-16, VGG-Face.	<ol style="list-style-type: none"> 1. Discuss the idea of transfer learning and review real-life applications. 2. Experiment independently/in pairs with gender classification using transfer learning. 3. Download data, apply data augmentation, and train the model. 4. Discuss the applicability of pre-trained models to different tasks. 	Google Colab, PyTorch/TensorFlow libraries, VGG-16 pre-trained models, Gender Dataset.	

Cross-curricular Topics: Biology (face recognition), Ethics (biases, ethical data use), Visual Arts (image analysis), Medical Data Analysis (facial recognition).

Assessment Criteria:

1. Clearly explains the stages of transfer learning (oral/written). 2. Implements fine-tuning of a pre-trained model. 3. Accurately identifies frozen layers. 4. Compares model results (accuracy/F1).

Prerequisite Knowledge: Solid Python programming skills, understanding of neural network architectures, experience with basic visual data preparation, training, splitting, and normalization.

SUBJECT: ARTIFICIAL INTELLIGENCE		Grade: 11	Semester: 2
Level: Neural Networks	Lesson hours: 9	Theoretical: 6	Practical: 3
Topic: Alternative Learning Concepts (16)			Chapter: 5

Objective	Learning Outcomes	
Provide a clear understanding of the differences between supervised, unsupervised, and reinforcement learning by applying practical examples such as clustering (k-means, DBSCAN) and generative AI.	<p>Upon successful completion of this topic, learners will be able to:</p> <ol style="list-style-type: none"> 1. Distinguish between supervised, unsupervised, and reinforcement learning approaches. 2. Perform k-means clustering on both synthetic and irregular datasets. 3. Define and apply the elbow method and compare k-means with DBSCAN. <p>Describe the concept of generative AI and formulate effective prompts for generating images, songs, and stories.</p> <ol style="list-style-type: none"> 4. Explain the basic principles of reinforcement learning. 	
Content	Teaching Activities	Tools/Platforms

<p>Differences between supervised, unsupervised, and reinforcement learning. k-means algorithm, DBSCAN, elbow method.</p> <p>Generative AI and prompt engineering. Zero-shot, few-shot prompting, chain-of-thought prompting. AI-generated music, images, text.</p> <p>History of generative AI. Reinforcement learning with examples.</p>	<ol style="list-style-type: none"> 1. Perform k-means clustering manually and using Colab. 2. Compare DBSCAN and k-means on different datasets. 3. Discuss reinforcement learning examples like autonomous vehicles and games. 4. Explore generative AI platforms, write and evaluate different prompts. 5. Create an image, story, or song using an LLM 	<p>Google Colab, Python (Scikit-learn), Stable Diffusion / DALL-E / LLM Playground, Teachable Machine, Colab 1: k-means clustering, Colab 2: DBSCAN vs k-means.</p>
<p>Cross-curricular Topics: Mathematics (centroids, distance, grouping), Machine Learning, Music/Arts (song and image generation), Language Processing (prompt engineering), Media (AI transparency and copyright).</p>		
<p>Assessment Criteria:</p> <p>1. Correctly distinguishes types of learning. 2. Performs k-means and the elbow method. 3. Compares DBSCAN results with k-means. 4. Generates an image or text and evaluates prompt effectiveness. 5. Explains reinforcement learning through a real example.</p>		
<p>Prerequisite Knowledge: Familiarity with ML clustering and training, practical understanding of neural networks, ability to present and analyze digital images.</p>		

SUBJECT: ARTIFICIAL INTELLIGENCE		Grade: 12	Semester: 1
Level: Linguistic Artificial Intelligence	Lesson hours: 10	Theoretical: 7	Practical: 3
Topic: Natural Language Processing (17)			Chapter: 1
Objective		Learning Outcomes	

Introduce students to the main challenges of natural language processing and the ways of representing text data; explore basic approaches to language modeling and their use in human-computer interaction.	<p>Upon successful completion of this topic, learners will be able to:</p> <ol style="list-style-type: none"> 1. Present the levels of linguistic abstraction: phonetics, grammar, semantics (syntax, grammar, semantics). 2. Explain the complexities of natural language understanding: ambiguity, context, irregular structures. 3. Perform text representation using bag-of-words and calculate term frequencies. 4. Present the concept of representing words as vectors through embeddings. 5. Define language modeling (sentence completion, word prediction) and describe its possible applications. 	
Content	Teaching Activities	Tools/Platforms
Levels of structure in natural language. Text ambiguity. NLP tasks: classification, sentiment analysis, QA. Text representation: Bag of Words, One-hot encoding, Word Embeddings. Concept of language modeling. Chatbots, DeepSeek, models like ChatGPT.	<ol style="list-style-type: none"> 1. Analyze sentence structures and ambiguity in groups. 2. Implement bag-of-words representation and calculate term frequencies. 3. Explore pre-trained embeddings (e.g., Word2Vec). 4. Play a "Next Word Prediction" game to demonstrate language response. 5. Study a simple chatbot example in a pre-prepared notebook. 	sklearn CountVectorizer, TfidfVectorizer, gensim or spacy for embedding examples, Google Colab, matplotlib or wordcloud for frequency visualization.
Cross-curricular Topics: Linguistics (grammar, word meaning), Psychology (language understanding, communication), Ethics (chatbot responsibility and transparency).		
Assessment Criteria: <ol style="list-style-type: none"> 1. Correctly presents levels of linguistic abstraction. 2. Performs BoW representation and explains its limitations. 3. Describes embeddings and gives comparative examples. 		

Prerequisite Knowledge: Understands the structural elements of text (sentence, word); performs data vectorization; uses Python to count word frequency and clarity.

SUBJECT: ARTIFICIAL INTELLIGENCE		Grade: 12	Semester: 1
Level: Linguistic Artificial Intelligence	Lesson hours: 9	Theoretical: 4	Practical: 5
Topic: Sequence Modeling (Recurrent Neural Networks and Transformers) (18)			Chapter: 2

Objective		Learning Outcomes	
Provide a general understanding of how Recurrent Neural Networks (RNNs) and Transformers work and teach their application on sequential data.		Upon successful completion of this topic, learners will be able to: <ol style="list-style-type: none"> 1. Understand the architectural differences between RNNs and Transformers. 2. Explain the concept of self-attention at a general level. 3. Present how sequential models are applied in sentiment classification and language modeling. 4. Implement a basic causal language modeling experiment using a Transformer-based model. 	
Content	Teaching Activities	Tools/Platforms	
What is sequential data? RNNs and LSTMs. Temporal dependencies in data. Basics of self-attention and Transformers. Sentiment classification using sequential models. Language modeling with causal Transformers.	<ol style="list-style-type: none"> 1. Present a simple explanation and example of the RNN concept. 2. Visualize and explain the Transformer's self-attention mechanism. 3. Conduct a sentiment classification experiment with a sequential model. 4. Implement causal language modeling using Transformers. 	Python, PyTorch, HuggingFace Transformers.	

Cross-curricular Topics: Linguistics (spoken/written sequences), Psychology (memory, sequential thinking), Data Science (time series), Literacy (context, syntax).

Assessment Criteria:

1. Can describe RNN and Transformer architectures.
2. Can explain the self-attention concept.
3. Implements a basic causal language modeling task.
4. Uses pre-trained Transformer models.

Prerequisite Knowledge: Understands language modeling and natural language structure; has skills in using PyTorch or the HuggingFace library.

SUBJECT: ARTIFICIAL INTELLIGENCE		Grade: 12	Semester: 1
Level: Linguistic Artificial Intelligence	Lesson hours: 5	Theoretical: 4	Practical: 1
Topic: Applications of Natural Language Processing (19)			Chapter: 3

Objective	Learning Outcomes	
Teach how NLP technologies are applied across various domains, evaluate the role of large language models, and use AI tools to solve creative and analytical tasks.	<p>Upon successful completion of this topic, learners will be able to:</p> <ol style="list-style-type: none"> 1. Present practical NLP applications, from translation to chatbots and code generation. 2. Explain what a large language model (LLM) is and why it matters. 3. Use a model like ChatGPT to solve real-world problems (generation, Q&A, discussion). 4. Brainstorm how AI language tools can be applied in educational, creative, or research contexts. 	
Content	Teaching Activities	Tools/Platforms

Grammatical analysis (part-of-speech tagging, syntax trees). Machine translation, question answering, logical inference, conversational agents (chatbots), code generation, LLMs as multifunctional tools. Few-shot, zero-shot, in-context learning, chain-of-thought prompting.	1. Discuss practical examples of NLP use cases (Google Translate, Siri, Grammarly). 2. Develop a simple GPT-based task and solve it using AI tools.	ChatGPT or other LLM interfaces (HuggingFace, Claude, Bard). Google Docs, Slides. Etherpad, Padlet, Jamboard.
Cross-curricular Topics: Languages and Literature (composition, critique), Media and Ethics (trustworthiness, copyright), Arts/Music (story or song generation), Law (document summarization, legal text analysis).		
Assessment Criteria: 1. Clearly presents at least 5 domains of NLP applications. 2. Accurately defines the role of LLMs and differences from other models. 3. Uses an AI tool purposefully for a given task. 4. Develops a realistic idea for NLP tool application.		
Prerequisite Knowledge: Understands text structure, basic NLP concepts (BoW, embeddings), knows language modeling and prompting.		

SUBJECT: ARTIFICIAL INTELLIGENCE		Grade: 12	Semester: 1
Level: Linguistic Artificial Intelligence	Lesson hours: 3	Theoretical: 2	Practical: 1
Topic: Neural Network Biases (20)			Chapter: 4

Objective	Learning Outcomes
Teach how biases present in data can influence neural network decisions, the potential consequences of these	Upon successful completion of this topic, learners will be able to: 1. Explain what bias is and how it manifests in data and models. 2. Present examples where neural networks reproduce or amplify biases in data.

biases, and ways to identify and reduce them.	3. Describe possible approaches or strategies for detecting and mitigating biases. 4. Discuss fairness in machine decisions and the challenges it poses.	
Content	Teaching Activities	Tools/Platforms
Definition of bias. Bias in diverse data (gender, race, language). Examples of biased AI systems in face recognition, automated hiring evaluation. Fairness vs. accuracy. Concept of model alignment. Data balancing, retraining with new conditions, human-in-the-loop considerations.	1. Discuss real-life bias examples using media materials or tests. 2. Analyze the consequences of AI system errors (social, legal). 3. Discuss different types of fairness (equity, group fairness, individual fairness). 4. Pair work: propose ways to fix or validate data during collection or modeling.	Videos and articles (e.g., Coded Bias documentary). Google Slides. Pre-trained biased models (distilGPT2, EleutherAI).
Cross-curricular Topics: Political Science and Law (decision fairness). Ethics (impact of technology on society). Psychology (unconscious biases). Technology (data cleaning).		
Assessment Criteria: 1. Clearly defines what bias is in data and how it appears in models. 2. Pays attention to social consequences. 3. Proposes practical ways to reduce bias in a specific model. 4. Clearly explains the concept of fairness and gives concrete examples.		
Prerequisite Knowledge: Understands fair data collection principles. Has advanced knowledge of machine learning.		

SUBJECT: ARTIFICIAL INTELLIGENCE	Grade: 12	Semester: 1
Level: AI Creator and Innovator	Lesson hours: 32	Theoretical: 2 Practical: 30

Objective	Learning Outcomes	
Teach students how to implement a complete AI-powered applied project based on a real-world problem, applying previously (during ~ 2.5 academic years) acquired theoretical and practical skills to develop a working model, while strengthening research, teamwork, and innovation skills.	<p>Upon successful completion of this topic, learners will be able to:</p> <ol style="list-style-type: none"> 1. Present a project idea and justify its relevance to a real or scientific problem. 2. Develop a project document including objectives, action plan, role distribution, and toolkit. 3. Build, test, and improve an AI model through data analysis and Python programming. 4. Analyse the effectiveness of the developed solution. 5. Disseminate the project by presenting its practical relevance and teamwork results. 6. Discuss opportunities for improvement, refinement, and innovative approaches based on feedback. 	
Content	Teaching Activities	Tools/Platforms
Transforming real-world problems into AI projects. Selecting and analysing data sources; ethics of data use. Main phases of AI model building: data preparation, algorithm selection, training and testing. Practical implementation in Python using sklearn, pandas, matplotlib, pytorch, tensorflow. Measuring effectiveness and fairness: bias, fairness, explainability. Teamwork, public presentation, and feedback culture.	<ol style="list-style-type: none"> 1. Form teams and select leadership roles. 2. Analyse the problem and collect data. 3. Test and connect with partner organizations. 4. Develop presentation skills. 5. Analyse feedback and improve the solution. 6. Defend and discuss the project with a review committee. 	Python, Scikit-learn, Numpy, Matplotlib, Pandas, PyTorch, TensorFlow/Keras, Google Colab, Git, Google Forms, Excel, Jupyter Notebooks, Canva, PowerPoint, Google Slides, Google Docs, Trello, Hugging Face, Streamlit, Flask, API endpoint tools.
Cross-curricular Topics: Data Science, Machine Learning, Algorithms and Statistics, Python Programming, Algebra and Calculus, AI Ethics, Project Management and Development Methodology.		

Assessment Criteria:

The student's work is assessed throughout the full project development cycle. Criteria include the quality of the planning and research phase, accuracy in applying algorithms and algebraic models, coding skills in Python, as well as creativity and innovative solutions. Additionally, the effectiveness of teamwork, technical and public presentation clarity, advanced AI knowledge, and the practical applicability of the project to real scientific or professional problems are assessed.

Prerequisite Knowledge: The student should have a strong grasp of fundamental AI concepts, skills in data preparation and analysis, and an understanding of core machine learning and deep learning methods. They should be able to use the Python programming language and relevant libraries for model building, training, and testing. They should understand problem formulation, project design, model evaluation (accuracy, precision, recall, fairness), and improvement principles. Experience with implementing projects is also required.