TEMPLATE FOR COURSE SYLLABUS FOR NEP IMPLEMENTATION

Discipline: Science [√] Arts, Humanities & Social Science ☐

Commerce ☐ BBA ☐ BCA ☐

Subject Name: COMPUTER SCIENCE

Subject Code: UCMSMAJ36013 (Will be provided by the University)

Semester: Semester I ☐ Semester II ☐ Semester III ☐ Semester IV ☐

Semester V ☐ Semester VI [√] Semester VII ☐ Semester VIII ☐

Course Name: ARTIFICIAL INTELLIGENCE

Course Code: (Will be provided by the University)

Course Credit:	Theoretical	3	Practical/Tutorial	1
Marks Allotted:	Theoretical	40	Practical/Tutorial	20
	Continuing Evaluation	10	Attendance	5

Course Type (tick the correct alternatives):

Major Core	[√]	AEC	☐
Interdisciplinary/ DSE	☐	SEC	☐
Minor / Generic Elective	☐	VAC	☐
Research Project/Dissertation	☐	Vocational	☐

Is the course focused on employability / entrepreneurship? YES [√] NO ☐

Is the course focused on imparting life skill? YES [√] NO ☐

Is the course based on Activity? YES ☐ NO [√]

Remarks by Chairman, UG BOS, if any

UG BOS Meeting Reference Number: Date

Course Code: UCMSMAJ36013

Course Name: ARTIFICIAL INTELLIGENCE

Brief Course Description:

The Artificial Intelligence (AI) course offers a comprehensive introduction to the principles and practices of AI, encompassing various techniques and applications that enable machines to mimic human intelligence. This course covers foundational topics such as machine learning, natural language processing, robotics, and neural networks. It is designed to equip students with the theoretical knowledge and practical skills needed to develop intelligent systems capable of solving complex problems.

Prerequisite(s) and/or Note(s):

Students should have a solid foundation in mathematics, particularly in linear algebra, calculus, and probability. Prior programming experience, especially in languages like Python, is highly recommended. Knowledge of basic algorithms and data structures will be beneficial. This course is suitable for students pursuing studies in computer science, engineering, data science, and related fields.

Course Objectives:

The primary objectives of the Artificial Intelligence course are to provide students with a thorough understanding of the fundamental concepts and techniques used in AI. The course aims to teach students how to apply AI methods to real-world problems, design and implement AI algorithms, and evaluate their effectiveness. By the end of the course, students will be well-versed in the ethical considerations and future trends in AI, preparing them to contribute to advancements in this rapidly evolving field.

Knowledge acquired:

1. Students will gain a deep understanding of various AI methodologies, including supervised learning, unsupervised learning, reinforcement learning, and deep learning.
2. They will learn about the architecture and functioning of neural networks, as well as principles of natural language processing (NLP).
3. Students will explore AI applications in diverse areas such as healthcare, finance, and autonomous systems.
4. The course provides a broad perspective on the impact of AI technologies in various domains, highlighting their significance and potential implications.

Skills gained:

1. Students will develop practical skills in designing, implementing, and evaluating AI algorithms.
2. Gain hands-on experience with TensorFlow, Keras, and scikit-learn for machine learning tasks.
3. Learn to preprocess and analyze data, train and optimize machine learning models, and interpret results.
4. Develop skills in problem-solving, critical thinking, and programming, essential for building intelligent systems.

Competency Developed:

1. Students will have the competency to design and implement AI-driven solutions to complex problems.
2. They will be capable of applying AI techniques across various domains, including developing intelligent assistants, and creating predictive models.
3. This competency prepares students for careers in AI research, data science, software development, and related fields.
4. The course provides a strong foundation for advanced studies and specialization in AI, preparing students to innovate and lead in the development of intelligent technologies.

Detailed Syllabus
3rd Year: Semester 6
UCMSMAJ36013: ARTIFICIAL INTELLIGENCE [Credits:3, Lecture :45]

Unit 1: Introduction (5 Lectures)
Introduction to Artificial Intelligence, Background and Applications, Turing Test and Rational Agent approaches to AI, Introduction to Intelligent Agents, their structure, behavior, and environment.

Unit 2: Problem Solving and Searching Techniques (15 Lectures)
Problem Characteristics, Production Systems, Control Strategies, Breadth First Search, Depth First Search, Hill climbing and its Variations, Heuristics Search Techniques: Best First Search, A* algorithm, Constraint Satisfaction Problem, Means-End Analysis, Introduction to Game Playing, Min-Max and Alpha-Beta pruning algorithms.

Unit 3: Knowledge Representation (15 Lectures)
Introduction to First Order Predicate Logic, Resolution Principle, Unification, Semantic Nets, Conceptual Dependencies, Frames, and Scripts, Production Rules, Conceptual Graphs. Programming in Logic (PROLOG)

Unit 4: Dealing with Uncertainty and Inconsistencies (5 Lectures)
Truth Maintenance System, Default Reasoning, Probabilistic Reasoning, Bayesian Probabilistic Inference, Possible World Representations.

Unit 5: Understanding Natural Languages (5 Lectures)
Parsing Techniques, Context-Free and Transformational Grammars, Recursive and Augmented Transition Nets. Suggested Readings

Suggested Readings:

1. DAN.W. Patterson, Introduction to A.I and Expert Systems – PHI, 2007.
2. Russell &Norvig, Artificial Intelligence-A Modern Approach**,** LPE, Pearson Prentice Hall, 2nd edition, 2005.
3. Rich & Knight, Artificial Intelligence – Tata McGraw Hill, 2nd edition, 1991.
4. W.F. Clocksin and Mellish, Programming in PROLOG, Narosa Publishing House, 3rd edition, 2001.
5. Ivan Bratko, Prolog Programming for Artificial Intelligence, Addison-Wesley, Pearson Education, 3rd edition, 2000.

UCMSMAJ36013: ARTIFICIAL INTELLIGENCE LAB [Credits:1, Lecture Hours:15]

Students are advised to do laboratory/practical practice not limited to, but including the following types of problems:

1. Write a prolog program to calculate the sum of two numbers.
2. Write a prolog program to find the maximum of two numbers.
3. Write a prolog program to calculate the factorial of a given number.
4. Write a prolog program to calculate the nth Fibonacci number.
5. Write a prolog program, insert_nth(item, n, into_list, result) that asserts that result is the list into_list with item inserted as the n,,th element into every list at all levels.
6. Write a Prolog program to remove the Nth item from a list.
7. Write a Prolog program, remove-nth(Before, After) that asserts the After list is the Before list with the removal of every n,,th item from every list at all levels.
8. Write a Prolog program to implement append for two lists.
9. Write a Prolog program to implement palindrome(List).
10. Write a Prolog program to implement max(X,Y,Max) so that Max is the greater of two numbers X and Y.
11. Write a Prolog program to implement maxlist(List,Max) so that Max is the greatest number in the list of numbers List.
12. Write a Prolog program to implement sumlist(List,Sum) so that Sum is the sum of a given list of numbers List.
13. Write a Prolog program to implement two predicates evenlength(List) and oddlength(List) so that they are true if their argument is a list of even or odd length respectively.
14. Write a Prolog program to implement reverse(List,ReversedList) that reverses lists.
15. Write a Prolog program to implement maxlist(List,Max) so that Max is the greatest number in the list of numbers List using cut predicate.

TEMPLATE FOR COURSE SYLLABUS FOR NEP IMPLEMENTATION

Discipline: Science [√] Arts, Humanities & Social Science ☐

Commerce ☐ BBA ☐ BCA ☐

Subject Name: COMPUTER SCIENCE

Subject Code: UCMSMAJ36014 (Will be provided by the University)

Semester: Semester I ☐ Semester II ☐ Semester III ☐ Semester IV ☐

Semester V ☐ Semester VI [√] Semester VII ☐ Semester VIII ☐

Course Name: COMPUTER GRAPHICS

Course Code: (Will be provided by the University)

Course Credit: Theoretical 3 Practical/Tutorial 1

Marks Allotted: Theoretical 40 Practical/Tutorial 20

Continuing Evaluation 10 Attendance 5

Course Type (tick the correct alternatives):

Major Core	[√]	AEC	☐
Interdisciplinary/ DSE	☐	SEC	☐
Minor / Generic Elective	☐	VAC	☐
Research Project/Dissertation	☐	Vocational	☐

Is the course focused on employability / entrepreneurship? YES [√] NO ☐

Is the course focused on imparting life skill? YES [√] NO ☐

Is the course based on Activity? YES [√] NO ☐

Remarks by Chairman, UG BOS, if any

UG BOS Meeting Reference Number: Date:

Prepared by CIRM

Course Code: UCMSMAJ36014

Course Name: COMPUTER GRAPHICS

Brief Course Description:

Computer Graphics is a field of study that explores the creation, manipulation, and representation of visual images using computers. The course covers foundational concepts such as raster graphics, vector graphics, and image processing techniques. Students learn about algorithms and techniques for rendering 2D and 3D graphics, including shading, lighting, and texture mapping. The course also delves into graphical user interface (GUI) design, animation principles, and virtual reality applications. Practical skills in programming graphics applications and using relevant software tools are emphasized, preparing students for careers in game development, animation, simulation, and visual effects industries.

Prerequisite(s) and/or Note(s):

To enroll in this course, students should have a basic understanding of several key areas of mathematics and geometry. This course is suitable for beginners and those looking to expand their knowledge in graphics and designing.

Course Objectives:

The computer graphics course aims to teach students fundamental principles and techniques for creating and manipulating graphical images using computers. It covers algorithms for rendering 2D and 3D graphics, including shading and texture mapping, while emphasizing practical programming skills. Students gain expertise applicable to fields like animation, virtual reality, user interfaces, and interactive simulations, preparing them for careers in game development, animation, and visual effects.

Knowledge acquired:

1. Understanding of fundamental principles and techniques for creating and manipulating graphical images.
2. Knowledge of algorithms for rendering 2D and 3D graphics, including shading, lighting, and texture mapping.
3. Practical skills in programming graphical applications and implementing rendering algorithms.

Skills gained:

1. Proficiency in programming graphical applications and implementing rendering algorithms.
2. Skills in applying rendering techniques such as shading, lighting, and texture mapping to create realistic images.
3. Competence in designing intuitive user interfaces and interactive graphical elements.
4. Capability to solve complex visual problems and effectively communicate concepts through computer-generated imagery.

Competency Developed:

1. Gain proficiency in using graphics software and programming languages to create and manipulate graphical images.
2. Develop skills in designing and integrating visual elements into applications, games, or simulations.
3. Apply problem-solving skills to solve challenges related to visual representation and user interaction.
4. Acquire a foundational understanding of graphics principles and their applications in various domains, such as animation, virtual reality, and user interfaces.

Detailed Syllabus

3rd Year: Semester 6

UCMSMAJ36014: COMPUTER GRAPHICS [Credits: 3, Lectures:45]

Unit 1: Introduction (3 Lectures)
Definition, Basic elements of Computer Graphics, Applications of Computer Graphics.

Unit 2: Graphics Hardware (7 Lectures)
Architecture of Raster and Random scan display devices, input/output devices.

Unit 3: Fundamental Techniques in Graphics (15 Lectures)
Raster scan line, circle and ellipse drawing, thick primitives, Polygon filling, line and polygon clipping algorithms, 2D and 3D Geometric Transformations, 2D and 3D Viewing Transformations (Projections-Parallel and Perspective), Vanishing points.

Unit 4: Geometric Modeling (8 Lectures)
Representing curves & Surfaces.

Unit 5: Visible Surface determination (6 Lectures)
Hidden surface elimination.

Unit 6: Surface rendering (6 Lectures)
Illumination and shading models. Basic color models and Computer Animation.

Suggested Readings:

1. J D Foley, A. Van Dan, Feiner, Hughes Computer Graphics Principles & Practice 2nd edition Publication Addison Wesley 1990.
2. D. Hearn, Baker: Computer Graphics, Prentice Hall of India 2008.
3. D F Rogers Procedural Elements for Computer Graphics, McGraw Hill 1997.
4. D F Rogers, Adams Mathematical Elements for Computer Graphics, McGraw Hill 2nd edition 1989.
5. Salini Govil-Pai, Principles of Computer Graphics, University Press

UCMSMAJ36014: COMPUTER GRAPHICS LAB [Credit:1, Lab Hours: 30]

Students are advised to do laboratory/practical practice not limited to, but including the following types of programs

1. Write a program to implement Bresenham's line drawing algorithm.
2. Write a program to implement mid-point circle drawing algorithm.
3. Write a program to clip a line using Cohen and Sutherland line clipping algorithm.
4. Write a program to clip a polygon using Sutherland Hodgeman algorithm.
5. Write a program to apply various 2D transformations on a 2D object (use homogenous coordinates).
6. Write a program to apply various 3D transformations on a 3D object and then apply parallel and perspectiveprojection on it
7. Write a program to draw Hermite/Bezier curve

TEMPLATE FOR COURSE SYLLABUS FOR NEP IMPLEMENTATION

Discipline: Science [√] Arts, Humanities & Social Science ☐

Commerce ☐ BBA ☐ BCA ☐

Subject Name: COMPUTER SCIENCE

Subject Code: UCMSMAJ36015

Semester: Semester I ☐ Semester II ☐ Semester III ☐ Semester IV ☐

Semester V [√] Semester VI ☐ Semester VII ☐ Semester VIII ☐

Course Name: THEORY OF COMPUTATION

Course Code: (Will be provided by the University)

Course Credit: Theoretical 3 Practical/Tutorial 1

Marks Allotted: Theoretical 60 Practical/Tutorial 0

Continuing Evaluation 10 Attendance 5

Course Type (tick the correct alternatives):

Major Core	[√]	AEC	☐
Interdisciplinary/ DSE	☐	SEC	☐
Minor / Generic Elective	☐	VAC	☐
Research Project/Dissertation	☐	Vocational	☐

Is the course focused on employability / entrepreneurship? YES [√] NO ☐

Is the course focused on imparting life skill? YES ☐ NO [√]

Is the course based on Activity? YES ☐ NO [√]

Remarks by Chairman, UG BOS, if any

UG BOS Meeting Reference Number: Date:

Course Code: UCMSMAJ36015

Course Name: THEORY OF COMPUTATION

Brief Course Description:

The Theory of Computation course explores into the foundational aspects of computer science, focusing on the study of formal languages, automata, and Turing machines. It explores the theoretical underpinnings of what can be computed and how efficiently computations can be performed. This course is crucial for understanding the limits of computation and the computational complexity of various problems.

Prerequisite(s) and/or Note(s):

Students interested in this course should have completed introductory courses in Discrete Mathematics and Algorithms. A solid grasp of mathematical logic and the ability to construct and understand mathematical proofs are essential for success in this course.

Course Objectives:

The primary objectives of the Theory of Computation course are to impart a deep understanding of the basic concepts of computation, to familiarize students with various computational models, and to enable them to analyze the complexity of computational problems. By the end of the course, students will be well-versed in the theoretical aspects of computer science that are fundamental to advanced study and research in the field.

Knowledge acquired:

1. Gain a comprehensive understanding, including finite automata, context-free grammars, and pushdown automata.
2. Learn about the hierarchy of formal languages and their corresponding automata models.
3. Study the theoretical framework of Turing machines and their role in computational theory.
4. Acquire knowledge about the boundaries of computation and the inherent limitations of computational processes.

Skills gained:

1. Students will acquire skills to design and analyze various types of automata and grammars.
2. Develop problem-solving abilities related to computational complexity and algorithmic challenges.
3. Gain proficiency in constructing and proving theorems in the context of automata theory and formal languages.
4. These skills are essential for addressing complex problems in computer science and for conducting theoretical research.

Competency Developed:

1. Students will develop competence in identifying and solving intricate computational problems.
2. The subject will hone abstract thinking and formal reasoning abilities to apply theoretical concepts effectively.
3. Apply theoretical knowledge to real-world scenarios in fields such as algorithm design, software development, and computer science research.
4. This competency prepares students for advancing in diverse fields where complex problem-solving and theoretical understanding are crucial.

Detailed Syllabus

3rd Year: Semester 6

UCMSMAJ36015: THEORY OF COMPUTATION [Credits: 3, Lectures:45]

Unit 1: Languages (8 Lectures)
Alphabets, string, language, Basic Operations on language, Concatenation, Kleene Star

Unit 2: Finite Automata and Regular Languages (15 Lectures)
Regular Expressions, Transition Graphs, Deterministic and non-deterministic finite automata, NFA to DFA Conversion, Regular languages and their relationship with finite automata, Pumping lemma and closure properties of regular languages.

Unit 3: Context free languages (12 Lectures)
Context free grammars, parse trees, ambiguities in grammars and languages, Pushdown automata (Deterministic and Non-deterministic), Pumping Lemma, Properties of context free languages, normal forms.

Unit 4: Turing Machines and Models of Computations (10 Lectures)
RAM, Turing Machine as a model of computation, Universal Turing Machine, Language acceptability, recursively enumerable and recursive languages.

Suggested Readings:

1. Daniel I.A.Cohen, Introduction to computer theory, John Wiley,1996
2. Lewis & Papadimitriou, Elements of the theory of computation , PHI 1997.
3. Hoperoft, Aho, Ullman, Introduction to Automata theory, Language & Computation –3rd Edition, Pearson Education. 2006
4. P. Linz, An Introduction to Formal Language and Automata 4th edition Publication Jones Bartlett, 2006

UCMSMAJ36015: THEORY OF COMPUTATION TUTORIAL [Credit:1, Lab Hours: 30]

Theory of Computation Tutorial as advised and assigned by Teacher(s)

TEMPLATE FOR COURSE SYLLABUS FOR NEP IMPLEMENTATION

Discipline: Science [√] Arts, Humanities & Social Science ☐

Commerce ☐ BBA ☐ BCA ☐

Subject Name: COMPUTER SCIENCE

Subject Code: UCMSMAJ36016 (Will be provided by the University)

Semester: Semester I ☐ Semester II ☐ Semester III ☐ Semester IV ☐

Semester V [√] Semester VI ☐ Semester VII ☐ Semester VIII ☐

Course Name: OPERATION RESEARCH

Course Code: (Will be provided by the University)

Course Credit: Theoretical 3 Practical/Tutorial 1

Marks Allotted: Theoretical 60 Practical/Tutorial 0

Continuing Evaluation 10 Attendance 5

Course Type (tick the correct alternatives):

Major Core	[√]	AEC	☐
Interdisciplinary/ DSE	☐	SEC	☐
Minor / Generic Elective	☐	VAC	☐
Research Project/Dissertation	☐	Vocational	☐

Is the course focused on employability / entrepreneurship? YES [√] NO ☐

Is the course focused on imparting life skill? YES [√] NO ☐

Is the course based on Activity? YES ☐ NO [√]

Remarks by Chairman, UG BOS, if any

UG BOS Meeting Reference Number: Date:

TEMPLATE FOR COURSE SYLLABUS FOR NEP IMPLEMENTATION

Course Code: UCMSMAJ36016

Course Name: OPERATION RESEARCH

Brief Course Description

The course Operations Research is a discipline that applies mathematical methods to optimize complex decision-making processes. This course explores techniques such as linear programming, integer programming, and network optimization to solve problems efficiently. Students learn to model real-world scenarios mathematically, analyze data, and make informed decisions using optimization algorithms.

Prerequisite(s) and/or Note(s):

No prior programming experience is required to enroll in this course, making it suitable for beginners. However, a basic understanding of computer operations and familiarity with high school-level mathematics will be beneficial. This course is ideal for students from diverse backgrounds, including those in computer science, engineering, data science, and other fields that benefit from optimization skills.

Course Objectives:

The objective of an Operations Research course is to teach students how to model and solve complex decision-making problems using mathematical and computational methods. Students learn to formulate real-world problems into mathematical models and apply optimization techniques like linear programming and network optimization to find optimal solutions. By exploring practical applications in fields like logistics, finance, and healthcare, students develop critical problem-solving skills essential for addressing organizational challenges effectively.

Knowledge Acquired:

1. Understanding of optimization techniques such as linear programming and integer programming
2. Skills in decision-making under uncertainty, including decision theory and stochastic modeling.
3. Proficiency in analyzing data to support decision-making processes.
4. Knowledge of practical applications in logistics, supply chain management, finance, healthcare, and other industries where optimization plays a crucial role.

Skills Gained:

1. Ability to translate real-world problems into mathematical models suitable for analysis and optimization.
2. Proficiency in applying optimization techniques such as linear programming, integer programming, and network optimization to find optimal solutions.
3. Skills in analyzing decision-making processes under uncertainty using tools like decision theory and stochastic modeling.
4. Enhanced problem-solving abilities, particularly in complex and dynamic environments.
5. Capability to analyze data effectively to derive insights and support decision-making.

Competency Developed:

1. Ability to analyse complex problems and apply mathematical models for optimization.
2. Competence in making informed decisions based on rigorous analysis of data and uncertainties.
3. Proficiency in formulating real-world problems into mathematical models for optimization.
4. Capability to efficiently resolve organizational challenges through quantitative analysis.
5. Understanding of applying OR principles across various fields for holistic problem-solving.

Detailed Syllabus

3rd Year: Semester 6

UCMSMAJ36016: OPERATION RESEARCH **[Credits: 3, Lectures:45]**

Unit 1: Basics of Operational Research (5 Lectures)

Origin & Development of Operational Research, Definition and Meaning of Operational Research, Different Phases of an Operational Research Study, Scope and Limitations of Operational Research, Mathematical Modeling of Real-Life Problems.

Unit 2: Transportation Model (8 Lectures)

Transportation problem and its variants: Northwest corner method, VAM method, Assignment problem: Hungarian method.

Unit 3: Linear Programming (15 Lectures)

Introduction to Linear algebra. Solution of a system of Linear Equations, Linear independence and dependence of vectors, Concept of Basis, Basic Feasible solution, Problem Formulation, Solution by Graphical Method, Simplex Algorithm, Artificial Starting Solution, Big-M and Two-phase method, Special cases in Simplex Method: Degeneracy, Alternative optima, Unbounded Solution, Infeasible Solution.

Unit 4: Duality (5 Lectures)

Definition of Dual Problem, Primal-Dual relationship, Duality problem, Dual Simplex.

Unit 5: Integer Programming (12 Lectures)

Integer programming and its variants: Branch and bound algorithm, cutting plane algorithm.

Suggested Readings:

1. 1. G. Hadley: Linear Programming. Narosa, 2002 (reprint).
2. A. Ravindran, D. T. Phillips and James J. Solberg: Operations Research-Principles and Practice, JohnWiley & Sons, 2005.
3. Hamdy A. Taha: Operations Research-An Introduction, Prentice Hall, 8th Edition, 2008. 4. F.S. Hillier.G.J.
4. Lieberman: Introduction to Operations Research- Concepts and Cases, 9th Edition, Tata McGraw Hill.2010.

UCMSMAJ36016: OPERATION RESEARCH TUTORIAL **[Credit:1, Lab Hours: 30]**

Operation Research Tutorial as advised and assigned by Teacher(s)