Curriculum for UG Degree Course
in
Computer Science Engineering & Artificial Intelligence

SAMMAKKA SARAKKA CENTRAL TRIBAL UNIVERSITY

सम्मक्का सारक्का केंद्रीय जनजातीय विश्वविद्यालय

(A Central University established by an Act of Parliament - Act 36 of 2023)

Jakaram Village, Mulugu District, Telangana State, Phone: 08715 - 293905

B.Tech Computer Science Engineering & Artificial Intelligence

Semester-I | 23 Credits

Semester-I | 23 Credits							
Semester	**Course Code**	**Course Title**	**Category**	**Credits**	**L**	**T**	**P**
I		Computer Science and IT Fundamentals	Major-Core	4	3	0	2
I		Engineering Maths	Major-Core	4	3	1	0
I		Problem Solving and Programming in C	Major-Core	4	3	0	2
I		Python Programming	SEC	4	3	0	2
I		Digital Twins	IDE	3	3	0	0
I		Communicative English	AEC	2	2	0	0
I		Environment Studies	VAC	2	2	0	0

Note: Course Curriculum is Dynamic and Subject to change as per the requirements.

SEMESTER I

				Credit Distribution			
Course Code	Course Title	Category	Credits	L	T	P	Remarks
	Computer Science and IT Fundamentals	Major-Core	4	3	0	2	

Course Objectives

1. Understand the fundamentals of computer systems, hardware components, and computing environments.
2. Learn software tools, development environments, and automation techniques used in computing workflows.
3. Introduce modern computing architectures and system platforms.
4. Provide an overview of theory, systems, AI courses that students will encounter during their B.Tech program.

Course Outcomes

Upon successful completion of this course, students will be able to:

1. Identify and explain the major hardware and software components of a computer system.
2. Use essential development tools, Linux utilities, version control systems, and scripting environments for productivity.
3. Describe modern computing architectures including CPU, GPU, cloud, and distributed platforms.
4. Explain the purpose and applications of core computer science theory and systems subjects.
5. Recognize AI, data-driven technologies, and emerging computing domains relevant to future studies and careers.

Course Content

UNIT I: Computer Components

- Theory: Introduction to computing systems and digital computing. What Computer Science is all about. History & basics: hardware, software, mathematics and applications.

Functional components of a computer system: CPU, motherboard, memory (RAM, ROM), storage devices (HDD, SSD), power supply, buses, I/O devices, peripherals, and ports. Desktop computer organization and component interactions. Overview of laptops, mobile systems, embedded devices, and data storage fundamentals.

- Lab/Practical: Hands-on demonstration of desktop computer assembly/disassembly. Identification of hardware components, peripheral setup, and system configuration.

UNIT II: Computing Tools, Linux Environment and Developer Productivity

- Theory: Development environments and software productivity tools. Introduction to VS Code and debugging concepts: breakpoints, watch variables, and step execution. Linux environment and command-line computing: file system navigation, permissions, file operations, and commonly used Linux commands/utilities. Command-line productivity tools: introduction to vi/vim, code navigation using ctags and cscope, and debugging concepts using gdb. Configuration management and collaboration: introduction to Git, repositories, commits, branching, and collaborative workflows. Technical documentation: introduction to README.md, documentation basics, Doxygen, and LaTeX for technical writing. Scripting and automation: shell scripting (bash, awk, sed, grep, cron jobs), introduction to Python scripting, Jupyter Notebook, Google Colab, and basics of build systems using Makefile.
- Lab/Practical: Hands-on exercises using Linux commands, VS Code debugging, Git repositories, shell scripting, LaTeX report writing, Jupyter notebooks, and simple automation tasks.

UNIT III: Modern Computing Architectures and Platforms

- Theory: Introduction to computing architectures: CPU, GPU, multicore systems, parallel computing basics, cloud computing, edge computing, virtualization, and data centers. Overview of system software: operating systems, compilers, interpreters, applications, and middleware. Internet infrastructure and digital ecosystems.
- Lab/Practical: Hands-on exploration of operating system utilities, cloud platforms, system monitoring tools, and computing environments.

UNIT IV: Overview of Theoretical Foundations in Computer Science

- Theory: Faculty-led overview lectures introducing foundational theory-oriented subjects in the curriculum: Programming for Problem Solving, Discrete Mathematics, Data Structures and Algorithms, Design and Analysis of Algorithms etc. Overview of the purpose, applications, and interconnections among theoretical computer science subjects.
- Lab/Practical: Simple algorithm demonstrations, computational puzzles, introductory coding activities, and problem-solving exercises.

UNIT V: Overview of Systems and Computing Infrastructure

- Theory: Faculty-led overview lectures introducing systems-oriented subjects: Computer Organization and Architecture, Operating Systems, Database Management Systems, Computer Networks etc. Overview of system design, hardware-software interaction, and computing infrastructure.
- Lab/Practical: Demonstrations of operating system tools, networking concepts, database interfaces, and system software.

UNIT VI: Overview of AI, Data and Emerging Technologies

- Theory: Faculty-led overview lectures introducing AI and data-driven subjects: Artificial Intelligence, Machine Learning, Natural Language Processing etc. Real-world applications, societal impact, and future directions in intelligent systems.
- Lab/Practical: Demonstrations of AI tools, introductory data analysis workflows, and intelligent application examples.

Textbooks

1. Brookshear, J. Glenn, & Brylow, Dennis. Computer Science: An Overview. 12th Edition, Pearson.
2. Morley, Deborah, & Parker, Charles S. Understanding Computers: Today and Tomorrow. Cengage Learning.

Reference Books

1. Patterson, David A., & Hennessy, John L. Computer Organization and Design: The Hardware/Software Interface. Morgan Kaufmann.
2. Tanenbaum, Andrew S. Structured Computer Organization. Pearson.
3. Kernighan, Brian W., & Pike, Rob. The Unix Programming Environment. Pearson.
4. Chacon, Scott, & Straub, Ben. Pro Git. Apress.

****

				Credit Distribution			
Course Code	Course Title	Category	Credits	L	T	P	Remarks
	Engineering Maths	Major-Core	4	3	1	0	

Course Objectives

This course covers a wide range of mathematical concepts that are useful in almost all engineering core subjects.

1. Strengthen students' understanding of single-variable and multivariable calculus from a computing perspective.
2. Introduce gradients, Jacobians, Hessians, Taylor approximations, and their applications.
3. Develop familiarity with matrix and vector calculus used in advanced courses in AI/DS.
4. Teach numerical methods for root finding, interpolation, numerical differentiation, integration, and ODE solving.
5. Sensitize students to floating-point error, conditioning, stability, convergence, and discretization error.

Course Content

UNIT – I: Calculus, Taylor Approximation and Integral Foundations

Functions, limits, continuity and differentiability; first- and higher-order derivatives; geometric and computational interpretation; Taylor polynomials; local linear/quadratic approximation and approximation error; common functions: exponential, logarithm, sigmoid, tanh, ReLU, softplus, and smooth approximations; definite and indefinite integrals; Fundamental Theorem of Calculus; substitution and integration by parts; improper integrals and continuous approximations to discrete sums.

UNIT – II: Multivariable Calculus

Functions of several variables; geometry of surfaces and level sets; partial derivatives, directional derivatives, and gradient vector; tangent planes and local linearization; Jacobian matrix for vector-valued functions; Hessian matrix, curvature, and second-order Taylor expansion; multivariable chain rule for scalar and vector functions; scalars, vectors, matrices, and tensors as variables.

UNIT – III: Vector Calculus

Introduction; scalar point function and vector point function; directional derivative; gradient; double and triple integrals (Cartesian and polar); change of order of integration in double integrals; change of variables (Cartesian to polar); divergence, curl, and their related properties;

Laplacian operator; line integral – work done; surface integrals; volume integral; Green's theorem, Stokes' theorem, and Gauss's Divergence theorem (statement and verification); first-order ordinary differential equations: separable and linear ODEs; systems of linear ODEs: basic idea.

UNIT – IV: Numerical Methods

Introduction to numerical computation; approximate solutions; round-off error; truncation error; absolute error; relative error; conditioning of problems; and stability of numerical procedures.

Numerical solution of nonlinear equations: bisection method, fixed-point iteration, Newton-Raphson method, and secant method; convergence, stopping criteria, and comparison of methods.

Curve fitting by the method of least squares: fitting a straight line, second-degree curve, exponential curve, and power curve; residual error and interpretation of fitted models.

Numerical integration: generalized quadrature, trapezoidal rule, Simpson's 1/3 rule, Simpson's 3/8 rule, and comparison of accuracy.

Numerical solution of ordinary differential equations: Taylor's series method, Euler's method, modified Euler's method, and fourth-order Runge-Kutta method; step-size selection, truncation error, and stability considerations.

Suggested Text Books

1. Gilbert Strang, *Calculus*.
2. Richard L. Burden and J. Douglas Faires, *Numerical Analysis*.

Suggested Reference Books

1. Erwin Kreyszig, *Advanced Engineering Mathematics*.

******

				Credit Distribution			
Course Code	Course Title	Category	Credits	L	T	P	Remarks
	Problem Solving and Programming in C	Major-Core	4	3	0	2	

Course Objectives:

1. To learn how to solve a given problem.
2. To illustrate the basic concepts of C programming language.
3. To discuss the concepts of Functions, Arrays, Pointers and Structures.
4. To gain a comprehensive understanding of dynamic memory allocation .
5. To apply concepts of structures and files to solve real word problems.

Course Outcomes: At the end of the course, the student should be able to

1. Demonstrate the ability to analyze complex problems and apply appropriate problem-solving techniques to devise effective solutions.
2. Apply control structures to solve programming problems effectively
3. Design efficient algorithms involving arrays, demonstrating a clear understanding of array data structures.
4. Solve programming problems that require the use of pointers, including pointer arithmetic and manipulation
5. Demonstrate the ability to declare structure variables and define their member data types.

Course Content

UNIT -I:

Introduction to Problem Solving: Problem Solving Aspect, Problem Identification, Problem Understanding, Algorithm Development, Solution Planning, Flowcharts, flow agorithm. Overview of C: History of C, C Language Elements, Basic Structure of C Program, C Tokens-Variables and Data Types, Operators, Expressions and Type Conversions.

UNIT -II:

Control Statements: Selection Statements- if and switch statements. Iterative Statements: for, while and do-while statements. Jump Statements: break and continue statements.

UNIT -III:

Arrays: Declaration, accessing array elements, Storing values, Operations on arrays, Multi-dimensional arrays. Functions: Introduction, Using Functions, Function declaration, Function definition and Function call, Parameter passing, Passing arrays to functions, Recursion, Storage classes.

UNIT-IV:

Pointers: Declaration and Initialization of pointer variables, Pointer arithmetic, Pointers and arrays, Pointer to pointer, Array of pointers, Pointers and functions, Dynamic Memory Allocation. Strings: Introduction to Strings, String handling functions, Preprocessor Directives.

UNIT-V:

Structures: Introduction, Nested Structures, Array of Structures, Structures and Functions, Unions. Command-Line Arguments: Command-line Arguments Files: Introduction, File Operations

Text Books:

1.B. A. Forouzan and R. F. Gilberg, Computer Science: A Structured Programming Approach Using C, 3/e, Cengage Learning, 2007.

2. Reema Thareja, Programming in C, Oxford University Press, AICTE Edition, 2018.

3. R.G. Dromey, "How to Solve it by Computer". 2014, Pearson.

Reference Books:

1. Jeri R. Hanly, Ellot B. Koffman, Problem Solving and Program Design in C, 5/e, Pearson

2. Brian W Kernighan and Dennis M Ritchie, The C Programming Language, Second Edition, Prentice Hall Publication.

3. Paul Deitel, Harvey Deitel -C How to Program with an introduction to C++,Eighth Edition

******

				Credit Distribution			
Course Code	Course Title	Category	Credits	L	T	P	Remarks
	Python Programming	SEC	4	3	0	2	

Objectives

The main objective is to teach Computational thinking using Python.

1. To know the basics of Programming
2. To convert an algorithm into a Python program
3. To construct Python programs with control structures.
4. To structure a Python Program as a set of functions
5. To use Python data structures-lists, tuples, dictionaries.
6. To do input/output with files in Python.
7. To construct Python programs as a set of objects.

Outcomes: On completion of the course, students will be able to:

1. Develop algorithmic solutions to simple computational problems.
2. Develop and execute simple Python programs.
3. Develop simple Python programs for solving problems.
4. Structure a Python program into functions.
5. Represent compound data using Python lists, tuples, and dictionaries.
6. Read and write data from/to files in Python Programs

UNIT-I

Introduction to Computing and Problem Solving: Fundamentals of Computing – Computing Devices – Identification of Computational Problems – Pseudo Code and Flowcharts – Instructions – Algorithms – Building Blocks of Algorithms.

Introduction to Python Programming: Python Interpreter and Interactive Mode– Variables and Identifiers – Arithmetic Operators – Values and Types – Statements, Reading Input, Print Output, Type Conversions, The type() Function and Is Operator, Dynamic and Strongly Typed Language.

Control Flow Statements: The if, The if…else, The if…elif…else Decision Control Statements, Nested if Statement, The while Loop, The for Loop, The continue and break Statements.

UNIT-II

Functions: Built-In Functions, Commonly Used Modules, Function Definition and Calling the Function, The return Statement and void Function, Scope and Lifetime of Variables, Default Parameters, Keyword Arguments, *args and **kwargs, Command Line Arguments. Strings: Creating and Storing Strings, Basic String Operations, Accessing Characters in String by Index Number, String Slicing and Joining, String Methods, Formatting Strings.

UNIT-III

Lists: list operations, list slices, list methods, list loop, mutability, aliasing, cloning lists, list Parameters; Tuples: tuple assignment, tuple as return value; Dictionaries: operations and methods; advanced list processing - list comprehension; Illustrative programs: selection sort, insertion sort, merge sort, histogram.

Files and exception: text files, reading and writing files, format operator; command line arguments, errors and exceptions, handling exceptions, modules, packages; Illustrative programs: word count, copy file.

UNIT-IV

Object-Oriented Programming: Classes and Objects, Creating Classes in Python, Creating Objects in Python, The Constructor Method, Classes with Multiple Objects, Class Attributes versus Data Attributes, Encapsulation, Inheritance The Polymorphism.

Functional Programming: Lambda. Iterators, Generators, List Comprehensions.

Text Books:

1. Introduction to Python Programming. Gowrishankar S, Veena A. CRC Press, Taylor & Francis Group, 2019
2. Allen B. Downey, "Think Python: How to Think Like a Computer Scientist", 2nd edition, Updated for Python 3, Shroff/O'Reilly Publishers, 2016 (http://greenteapress.com/wp/think- python/)

Reference Books:

1. Learning To Program With Python. Richard L. Halterman. Copyright © 2011
2. Python for Everybody, Exploring Data Using Python 3. Dr. Charles R. Severance. 2016.

******

				Credit Distribution			
Course Code	Course Title	Category	Credits	L	T	P	Remarks
	Digital Twins	IDE	3	3	0	0	

Course Objectives

1. Understand Digital Twin concepts and applications.
2. Learn IoT-based data acquisition and virtual modelling.
3. Integrate AI/ML for Digital Twin analytics.
4. Design and implement Digital Twin systems.
5. Explore future trends, sustainability, and Industry 4.0.

Course Outcomes

1. Explain Digital Twin concepts and applications.
2. Apply IoT and real-time data processing techniques.
3. Integrate virtual models with physical systems.
4. Design and implement Digital Twin solutions.
5. Analyze trends and applications of Digital Twins.

Unit – I Introduction to Digital Twins

Introduction - Overview and evolution of Digital Twin technology - Key components: physical systems, digital systems, and data integration - Applications in industries like manufacturing, healthcare, and smart cities - Benefits of Digital Twins in predictive maintenance and operational efficiency - Challenges and ethical considerations in Digital Twin implementation.

Unit – II Foundations of Digital Twins

Data acquisition from physical systems using IoT and sensors - Real-time data streaming and processing techniques - Creating virtual models using CAD and simulation tools - Communication protocols for Digital Twin systems - Integration of AI/ML for predictive and prescriptive analytics.

Unit – III Building and Implementing Digital Twins

Steps to design and deploy a Digital Twin system - Tools and platforms for Digital Twin creation - Synchronizing real-time data with virtual models - Case studies of successful Digital Twin implementations - Troubleshooting and maintaining Digital Twin systems.

Unit - IV – Future Trends

Exploring advancements in Digital Twin technologies - Scalability and interoperability in large-scale systems

Unit – V Innovation in Digital Twins

Potential impact on Industry 4.0 and beyond - Sustainability and environmental applications - Emerging opportunities and career pathways in Digital Twin development.

Reference Books

1. Vijay Raghunathan, Digital Twin: A Complete Guide For The Complete Beginner,2019.
2. Lavanya Sharma, Pradheep Kumar Garge, Digital Twins and Simulation Technology: Concepts and Applications:1st Edition, Chapman and Hall/CRC.
3. Shyam Varan Nath, Pieter van Schalkwyk, Dan Isaacs, Building Industrial Digital Twins, Packt Publishing,2021.
4. Gopal Chaudary, Manju Khari, Mohamed Elhoneny, Digital Twin Technology: 1st Edition, CRC Press,2021.
5. Andrew Yeh Chris Nee, Fei Tao, and Meng Zhang, "Digital Twin Driven Smart Manufacturing", Elsevier Science., United States, 2019.
6. Sharma, Angira, et al." digital twins: state of the art Theory and practice ,challenges, and open Research Questions." Journal of Industrial Engineering.
7. Fuller, aidan, et al. "Digital Twin: Enabling Technologies, challenges, and open Research"IEEE,2022.
8. Yao, jun-Feng, et al. "Systematic Review of Digital Twin Technology and Applications. "Visual computing for Industry, Biomedicine, and art, vol.6, Article no.10,2023. DOI:10.1186/s42492-023-00072-7.
9. Fuller, Aidan, et al "Digital Twin: Enabling Technologies, Challenges open and Research". IEEE Access, Vol,PP,no.99,may 2020,pp.1-1 DOI:10.1109/ACCESS.2020.license:CC by 4.0, keele University, Zhong Fan, Charles Day, Christ Barlow.

******

				Credit Distribution			
Course Code	Course Title	Category	Credits	L	T	P	Remarks
	Communicative English	AEC	2	2	0	0	

******

				Credit Distribution			
Course Code	Course Title	Category	Credits	L	T	P	Remarks
	Environment Studies	VAC	2	2	0	0	

******