# YOGI VEMANA UNIVERSITY::KADAPA

## Syllabus for 4-Year UG Honours in B.Sc. (Quantum Technologies) as Major in consonance with Curriculum framework w.e.f. AY 2025-26 COURSE STRUCTURE (for Semester I to VI)

| Year | Semester | Course | Title of the Course | No.of Hrs /Week | No. of Credits |
|---|---|---|---|---|---|
| I | I | 1 | Computer Fundamentals and Office Automation | 3 | 3 |
| | | | Computer Fundamentals and Office Automation-Practical | 2 | 1 |
| | | 2 | Mathematical Physical and Computing foundations of Quantum Computing | 3 | 3 |
| | | | Mathematical Physical and Computing foundations of Quantum Computing -practical | 2 | 1 |
| | II | 3 | Problem Solving using C | 3 | 3 |
| | | | Problem Solving using C - Practical | 2 | 1 |
| | | 4 | Numerical Methods for Quantum Computing | 3 | 3 |
| | | | Numerical Methods for Quantum Computing - Practical | 2 | 1 |
| II | III | 5 | Data Structures using Python | 3 | 3 |
| | | | Data Structures using Python-Practical | 2 | 1 |
| | | 6 | Operating Systems | 3 | 3 |
| | | | Operating Systems-Practical | 2 | 1 |
| | | 7 | Foundations of Quantum Technologies | 3 | 3 |
| | | | Foundations of Quantum Technologies-Practical | 2 | 1 |
| | IV | 8 | Python for Quantum Computing | 3 | 3 |
| | | | Python for Quantum Computing-Practical | 2 | 1 |
| | | 9 | Quantum Computing using Qiskit | 3 | 3 |
| | | | Quantum Computing using Qiskit -Practical | 2 | 1 |
| | | 10 | Quantum Computing Algorithms | 3 | 3 |
| | | | Quantum Computing Algorithms -Practical | 2 | 1 |

| | | | | | |
|---|---|---|---|---|---|
| **III** | **V** | **11** | Quantum Computing Applications | **3** | **3** |
| | | | Quantum Computing Applications -Practical | **2** | **1** |
| | | | | | |
| **Year** | **Semester** | **Course** | **Title of the Course** | **No.of Hrs /Week** | **No. of Credits** |
| | | **12 A** | Computer Networks | **3** | **3** |
| | | | Computer Networks-Practical | **2** | **1** |
| | | **OR** | | | |
| | | **12 B** | Mathematics for Machine Learning | **3** | **3** |
| | | | Mathematics for Machine Learning-Practical | **2** | **1** |
| | | | | | |
| | | **13 A** | Classical Cryptography & Network Security | **3** | **3** |
| | | | Classical Cryptography & Network Security-Practical | **2** | **1** |
| | | **OR** | | | |
| | | **13 B** | Python for Machine Learning | **3** | **3** |
| | | | Python for Machine Learning-Practical | **2** | **1** |
| | | | | | |
| | | **14 A** | Quantum Cryptography Protocols | **3** | **3** |
| | | | Quantum Cryptography Protocols-Practical | **2** | **1** |
| | | **OR** | | | |
| | | **14 B** | Deep Learning Fundamentals | **3** | **3** |
| | | | Deep Learning Fundamentals-Practical | **2** | **1** |
| | | | | | |
| | **VI** | **15 A** | Quantum Simulation in Cyber Security | **3** | **3** |
| | | | Quantum Simulation in Cyber Security-Practical | **2** | **1** |
| | | **OR** | | | |
| | | **15 B** | Quantum Machine Learning | **3** | **3** |
| | | | Quantum Machine Learning-Practical | **2** | **1** |

**Note:** In the III Year (during the V and VI Semesters), students are required to select a pair of electives from one of the **TWO** specified domains. **For example: if set 'A' is chosen, courses 12 to 15 to be chosen as 12A, 13A, 14A and 15A.** To ensure in-depth understanding and skill development in the chosen domain, students must continue with the same domain electives in both the V and VI Semesters.

**SEMESTER-I**

**COURSE 1: COMPUTER FUNDAMENTALS AND OFFICE AUTOMATION**

**Theory** **Credits: 3** **3 hrs/week**

---

**Course Objectives**

1. **Understand foundational computing concepts**, including number systems, the evolution of computers, block diagrams, and generational progress.
2. **Develop knowledge of computer architecture**, focusing on system organization and networking fundamentals.
3. **Acquire practical skills in document creation**, formatting, and digital presentations using word processing tools.
4. **Gain proficiency in spreadsheet operations**, such as data entry, formulas, functions, and charting techniques.
5. **Introduce data visualization and basic modelling principles**, fostering analytical thinking in structuring and interpreting data sets.

**Course Outcomes**

1. At the End of the Course, The Students will be able to **explain different number systems**, the historical evolution of computers, and identify key components in a block diagram.
2. Learners will demonstrate **basic blocks of a computer and fundamental networking knowledge.**
3. Learners will create professional-level documents and **design visually appealing presentations** using word processing software and presentation software.
4. Learners will manipulate data within spreadsheets, apply formulas, and **generate accurate summaries and visualizations.**
5. Learners will apply data modelling techniques to **analyze, organize, and represent data effectively** in various scenarios.

**Syllabus:**

**Unit 1. Number Systems, Evolution, Block Diagram and Generations:**

**Number Systems:** Binary, Decimal, Octal, Hexadecimal; conversions between number systems.

**Evolution of Computers:** History from early mechanical devices to modern-day systems.

**Block Diagram of a Computer:** Components like Input Unit, Output Unit, Memory, CPU (ALU + CU).

**Generations of Computers:** First to Fifth Generation with technologies, characteristics, examples.

**Unit 2. Basic organization and N/W fundamentals:**

**Computer Organization:** Functional components –Storage types, Memory hierarchy.

**Types of Computers:** Micro, Mini, Mainframe, and Supercomputers.

**Networking Fundamentals:** Definition, need for networks, types (LAN, WAN, MAN), topology (Star, Ring, Bus).

**Internet Basics:** IP Address, Domain Name, Web Browser, Email, WWW.

**Unit 3. Word Processing and presentations:**

**Word Processing Basics: Using MS Word/Google Docs** – formatting, styles, tables, mail merge. Applications - Creating resumes, reports. Keyboard Shortcuts- File Operations, Editing Operations, Formatting Shortcuts, Navigation and Selection.

**Presentation Tools:** Using PowerPoint/Google Slides – slide design, animations, transitions. Applications - Brochures, and presentations.

**Unit 4. Spreadsheet Basics:**

**Spreadsheet Concepts:** Understanding rows, columns, cells in tools like MS Excel/Google Sheets, cell referencing.

**Functions and Formulae:** SUM, AVERAGE, IF, COUNT.

**Charts and Graphs:** Creating visual representations, Types of Charts.

**Data Handling:** Sorting, filtering, conditional formatting.

**Text Functions:** LEFT, RIGHT, MID, LEN, TRIM, CONCAT, TEXTJOIN

**Advanced Functions:** Logical: IF, AND, OR, IFERROR, Lookup

**Unit 5. Data Analysis and Visualization:**

**Conditional Formatting:** Custom rules, Color scales, Icon sets, Data bars

**Data Analysis Tools:** Pivot Tables and Pivot Charts, Data Validation (Drop-downs, Input Messages, Error Alerts).

**What-If Analysis:** Goal Seek, Scenario Manager, Data Tables

**Charts and Dashboards:** Creating Interactive Dashboards, Using slicers with Pivot Tables, Combo Charts and Sparklines

**Textbooks:**

1. **Fundamentals of Computers,** Reema Thareja, Oxford University Press, Second Edition
2. **Fundamentals of Computers**, V. Rajaraman – PHI Learning
3. **Introduction to Computers** by Peter Norton – McGraw Hill
4. **Microsoft Office 365 In Practice** by Randy Nordell – McGraw Hill Education

**References:**

1. **Excel 2021 Bible** by Michael Alexander, Richard Kusleika – Wiley
2. **Networking All-in-One For Dummies** by Doug Lowe – Wiley
3. **Microsoft Official Docs and Training**: https://learn.microsoft.com
4. **Google Workspace Learning Center**: https://support.google.com/a/users/

**Activities:**

**Outcome:** At the End of the Course, The Students will be able to **explain different number systems**, the historical evolution of computers, and identify key components in a block diagram.

**Activity:** Create a digital poster or info graphic comparing number systems (binary, decimal, octal, hexadecimal) and illustrating the timeline of computer generations with key innovations.

**Evaluation Method:** Rubric-based assessment of the poster presentation on a 10-point scale focusing on:

- Accuracy of number system conversions
- Correct identification of block diagram components
- Visual organization and creativity

**Outcome:** Learners will demonstrate **basic blocks of a computer and fundamental networking knowledge.**

**Activity:** Design a concept map showing the internal architecture of a computer and types of networks (LAN, WAN, MAN), including devices and topologies.

**Evaluation Method:** Checklist-based peer review and instructor validation:

- Completeness of the map

- Correctness of networking concepts
- Use of appropriate terminology
- Logical flow and structure of the map

**Outcome:** Learners will create professional-level documents and **design visually appealing presentations** using word processing software and presentation software.

**Activity:** Prepare a formal report (e.g., project proposal) in a word processor and present it using a slide deck with transitions, embedded media, and design elements.

**Evaluation Method:** Performance-based evaluation using a 10-point scoring scale:

- Formatting and structure of the document
- Presentation aesthetics and clarity
- Communication skills during presentation

**Outcome:** Learners will manipulate data within spreadsheets, apply formulas, and **generate accurate summaries and visualizations.**

**Activity:** Analyze a dataset (e.g., student scores or sales data) using spreadsheet software. Apply formulas (SUM, AVERAGE, IF, VLOOKUP) and create relevant charts.

**Evaluation Method:** Practical test with a rubric:

- Correct use of formulas
- Accuracy of data summaries

**Outcome:** Learners will apply data modeling techniques to **analyze, organize, and represent data effectively** in various scenarios.

**Activity:** Prepare an interactive dashboard for a given data set using EXCEL.

**Evaluation Method:** Evaluation of the dashboard on a 10-point scoring scale:

- Presentation aesthetics and clarity
- Inter activeness
- Communication skills during presentation

**SEMESTER-I**

**COURSE 1: COMPUTER FUNDAMENTALS AND OFFICE AUTOMATION**

**Practical** **Credits: 1** **2 hrs/week**

---

**List of Experiments:**

1. Demonstration of Assembling and Dissembling of Computer Systems.
2. Identify and prepare notes on the type of Network topology of your institution.
3. Prepare your resume in Word.
4. Using Word, write a letter to your higher official seeking 10-days leave.
5. Prepare a presentation that contains text, audio and video.
6. Using a spreadsheet, prepare your class Time Table.
7. Using a Spreadsheet, calculate the Gross and Net salary of employees(Min 5) considering all the allowances.
8. Generate the class-wise and subject-wise results for a class of 20 students. Also generate the highest and lowest marks in each subject.
9. Using IF, AND, OR, and IFERROR to Automate Grade Evaluation.
    a. Create a table of student scores in different subjects.
    b. Use IF to assign grades (A/B/C/Fail).
    c. Use IFERROR to handle missing scores or invalid data.
10. Employee Database Search Using VLOOKUP, HLOOKUP, XLOOKUP, INDEX, and MATCH
    a. Create a database of employees (Name, ID, Department, Salary).
    b. Implement VLOOKUP to search by employee ID.
    c. Use HLOOKUP to extract department heads by role.
    d. Apply XLOOKUP for more flexible searches.
    e. Use INDEX + MATCH as an alternative to VLOOKUP.
11. Sales Report Analysis Using Pivot Tables and Charts
    a. Use a dataset of product sales (Product, Region, Date, Quantity, Revenue).
    b. Create Pivot Tables to summarize data by region/product.
    c. Insert Pivot Charts for visual analysis (e.g., bar, line).
    d. Add slicers to make the dashboard interactive.
12. Designing a Data Entry Form with Drop-downs and Input Rules
    a. Create a student registration form.
    b. Add drop-down lists for course selection using Data Validation.
    c. Add input messages to guide users.
    d. Add error alerts for wrong entries.
13. Monthly Budget Planning using Goal Seek and Scenario Manager
    a. Create a simple personal budget (income, expenses, savings).
    b. Use Goal Seek to determine income needed to save a desired amount.
    c. Use Scenario Manager to compare different budgeting scenarios (best/ worst/ realistic case).
    d. Create a one-variable Data Table to analyze how different expenses affect savings.
14. Dashboard Creation Using Combo Charts, Sparklines & Slicers
    a. Use existing sales or attendance data.
    b. Insert combo charts (e.g., column + line).
    c. Add spark lines to show trends.
    d. Use slicers with Pivot Tables to control dashboard elements.
    e. Finalize and format for interactivity.

**SEMESTER-I**

**COURSE 2: Mathematical Physical and Computing foundations of Quantum Computing**

**Theory** **Credits: 3** **3 hrs/week**

---

**Course Objectives:**

By the end of the course, students will be able to:

1. Understand foundational mathematical concepts such as vectors, matrices, and linear transformations in quantum computing.
2. Explore the mathematical representation of quantum gates and qubits using linear algebra.
3. Apply complex numbers, eigenvalues, and eigenvectors to quantum computational models.
4. Relate abstract mathematical ideas to practical quantum computing operations.
5. Develop problem-solving skills for quantum mechanics and quantum circuit analysis.
6. To introduce the evolution of computer science and its mathematical basis.

**Learning Outcomes:**

Students will be able to:

1. Apply complex number operations in quantum computations.
2. Compute eigenvalues and eigenvectors for quantum state matrices.
3. Interpret the Bloch sphere for single-qubit representation.
4. Identify Hermitian and unitary operators as foundational elements in quantum mechanics.
5. Understand the evolution and fundamental experiments leading to quantum mechanics.
6. Explain the mathematical and physical foundations behind superposition and entanglement.
7. Explain classical computational models and their historical development.

**Unit I: Foundations of Quantum Mathematics (9 Periods)**

**Vectors and Linear Combinations:** Euclidean vectors, magnitude and direction, scalar multiplication, vector addition.

**Superposition and Measurement:** Understanding superposition states and measurement in quantum systems.

**Introduction to Matrices:** Matrix definition, notation, simple operations (addition, transpose, scalar multiplication).

**Matrix Multiplication and Properties:** Matrix-vector and matrix-matrix multiplication.

**Quantum Gates and Circuit Models:** Logic gates vs quantum gates, matrix representation of quantum gates (NOT, AND, OR, Pauli-X).

**Unit II: Elementary Linear Algebra for Quantum Systems (9 Periods)**

**Sets, Functions, and Vector Spaces:** Sets, Cartesian product, functions, fields, and vector spaces.

**Subspaces, Basis, and Dimension:** Linear independence, span, and basis of a vector space.

**Linear Transformations:** Definition, properties, and matrix representation.

**Transformations Inspired by Euclid:** Translation, rotation, and projection.

**Linear Operators and Functionals:** Matrix dependence on basis, commutator operations.

**Unit III: Complex Numbers and Eigen Concepts (9 Periods)**

**Complex Number System:** Cartesian, polar, and exponential forms; conjugates and modulus.

**Complex Numbers in Quantum Mechanics:** Bloch sphere representation and Euler's formula.

**Eigenvalues and Eigenvectors:** Definitions, computation, and physical significance.

**Determinants and Invertibility:** Matrix inverse and the invertible matrix theorem.

**Applications in Quantum State Analysis:** Diagonalization, Hermitian and unitary operators.

**Unit 4 – Quantum Mechanics and Its Principles**

**History of Quantum Mechanics:**Classical Physics foundations (Newton, Maxwell, Galileo),Failures of classical theory and emergence of quantum concepts,Key contributors: Planck, Einstein, Bohr, Heisenberg, Schrödinger, Dirac

**Atoms and Thermodynamics:** Atomic theory, kinetic theory, and laws of thermodynamics,**Statistical Mechanics:** Maxwell–Boltzmann distribution and entropy (Boltzmann's constant),**Photoelectric Effect:** Einstein's explanation of quantized energy, **Wave–**

**Particle Duality:** de Broglie's matter waves and Bohr's model, **Quantum Models:** Schrödinger's wave equation, Heisenberg's matrix mechanics, **Copenhagen Interpretation:** Bohr's complementarity and probabilistic measurement

**Key Principles for Quantum Computing:**Linear Algebra in Quantum Systems,Superposition and Interference, Dirac (Bra–Ket) Notation ,Quantum Uncertainty (Heisenberg Principle), Entanglement and Non-locality, Postulates of Quantum Mechanics

**Unit-5: Introduction to Computer Science,** Historical Foundations, Turing Machines, Logic Circuits and Gates, Computational Complexity and Efficiency,Energy, Reversibility, and Computation

Text Books:


1. ***Essential Mathematics for Quantum Computing*** **by Leonard S. Woody III (Packt, 2022)**.
2. ***Quantum Computing from the Ground Up by Riley Tipton Perry, by World Scientific Publishing Co. Pte. Ltd.***

**SEMESTER-I**

**COURSE 2: Mathematical Physical and Computing foundations of Quantum Computing**

**Practical** **Credits: 1** **2 hrs/week**

---

**List of Experiments:**

**Tools:** GeoGebra, Desmos, Python (NumPy), or any simple matrix calculator.

| No. | Experiment Title | Concepts Covered | Tool/Method |
|---|---|---|---|
| 1 | **Vector Addition and Superposition** | Represent two 2D vectors and visualize their superposition (quantum state analogy). | GeoGebra / Desmos |
| 2 | **Matrix Multiplication Basics** | Multiply 2×2 matrices and visualize the transformation of a 2D point. | Python NumPy / GeoGebra |
| 3 | **Quantum NOT Gate Simulation** | Implement NOT gate matrix ( X = \begin{bmatrix}0 & 1 \ 1 & 0\end{bmatrix} ) and apply to | 0⟩ and |
| 4 | **Matrix Transpose and Symmetry Check** | Create any 3×3 matrix and verify (A = A^T) (symmetric) or (A \neq A^T). | Simple math tool |
| 5 | **Vector Space Verification** | Verify if given vectors form a subspace under addition and scalar multiplication. | GeoGebra / manual |
| 6 | **Linear Transformation Visualization** | Use a 2×2 matrix to rotate or project points in 2D space. | GeoGebra |
| 7 | **Complex Number Plane Plotting** | Plot complex numbers in Cartesian and Polar forms; verify Euler's relation (e^{iθ} = \cosθ + i\sinθ). | Desmos / Python |
| 8 | **Bloch Sphere Projection (Intro)** | Represent | 0⟩ and |

**Tools:** Logisim or Logisim Evolution

| No. | Experiment Title | Concept | Tool/Method |
|---|---|---|---|
| 1 | **Binary to Decimal Converter** | Implement conversion circuit using XOR, AND, and OR gates. | Logisim |
| 2 | **Half Adder and Full Adder Circuits** | Design classical addition logic for two bits, then extend to full adder. | Logisim |
| 3 | **NOT, AND, OR, XOR Gates Demonstration** | Build and verify truth tables for basic logic gates. | Logisim |
| 4 | **Turing Machine Emulator (Basic)** | Simulate tape read/write using a simple state transition circuit. | Logisim |
| 5 | **Binary Pattern Recognizer** | Design finite automata circuit that outputs high for a specific binary pattern (like "101"). | Logisim |
| 6 | **Logic-to-Quantum Transition Concept** | Compare Boolean logic gates to unitary matrix equivalents (e.g., NOT ↔ X gate). | Logisim + explanation |

**SEMESTER-II**

**COURSE3: PROBLEM SOLVING USING C**

**Theory** **Credits:3** **3 hrs/week**

**Course Objectives:**

1. Understand the fundamentals of computer programming, Apply structured problem-solving approaches using algorithms, flowcharts, and C programming constructs.
2. Develop efficient logic using decision-making, loop, and jump control statements.
3. Utilize derived data types like arrays and strings for modular program design.
4. Design and implement modular solutions using functions, recursive logic, pointer operations, and dynamic memory management.
5. Handle complex data structures including structures, unions, and text file operations.

**Course Outcomes:**

At the end of the course, students will be able to:

1. Understand basic computing concepts, programming paradigms and write structured C programs.
2. Apply control flow statements to solve logical and repetitive tasks in C.
3. Implement arrays and string operations to manage and manipulate data efficiently.
4. Design modular code using functions, recursion, and appropriate parameter passing.
5. Utilize pointers and memory operations for effective data handling. Demonstrate competence in dynamic memory allocation and text file processing.

**Unit 1. Introduction to computer programming:**

Introduction, Types of software, Compiler and interpreter, Concepts of Machine level, Assembly level and high-level programming, Flowcharts and Algorithms, Fundamentals of C: History of C, Features of C, C Tokens-variables and keywords and identifiers, constants and Data types, Rules for constructing variable names, Operators, Structure of C program, Input /output statements in C-Formatted and Unformatted I/O

**Unit 2. Control statements:**

Decision making statements: if, if else, else if ladder, switch statements. Loop control statements: while loop, for loop and do-while loop. Jump Control statements: break,continue and goto.

**Unit 3. Derived datatypes in C:**

Arrays: One Dimensional arrays - Declaration, Initialization and Memory representation; Two Dimensional arrays - Declaration, Initialization and Memory representation. Strings: Declaring & Initializing string variables; String handling functions, Character handling functions

**Unit 4. Functions:**

Pointers: Pointer data type, Pointer declaration, initialization, accessing values using pointers. Pointer arithmetic, Pointers and arrays.

Function Prototype, definition and calling. Return statement. Nesting of functions. Categories of functions. Recursion (Basic Concept only). Parameter Passing by address & by value. Local and Global variables. Storage classes: automatic, external, static and register.

**Unit 5. Dynamic Memory Management:**

Introduction, Functions - malloc, calloc, realloc, free Structures: Basics of structure, structure members, accessing structure members, nested structures, array of structures, structure and functions, structures and pointers. Unions - Union definition; difference between Structures and Unions. Working with text files - modes: opening, reading, writing and closing text files.

**Text Books:**

1. Programming in ANSI C, E. Balagurusamy, Tata McGraw Hill, 6th Edn
2. Computer fundamentals and programming in C, Reema Theraja, Oxford University Press

**Reference Books:**

1. Let us C, Y. Kanetkar, BPB publications
2. Head First C: A Brain-Friendly Guide, David Griffiths, Dawn Griffiths

**Activities:**

Outcome: Understand basic computing concepts, programming paradigms and write structured C programs.

**Activity:** Create a concept map of computing fundamentals and programming paradigms (procedural, structured, object-oriented). Then, they write a structured C program (e.g., a calculator or student grade system) using proper syntax, indentation, and modular design.

Evaluation Method: Rubric-based Code Review & Viva to check the ○ The correctness of the concept map
○ Correct use of structure (main + functions)

○ Identification of paradigm used
○ Code readability and documentation

**Outcome:** Apply control flow statements to solve logical and repetitive tasks in C.

**Activity:** Implement a program that solves a logic puzzle (e.g., number guessing game, pattern

generation, or prime number finder) using if, switch, for, while, and do-while.
**Evaluation Method:** Automated Test Cases + Peer Review to check the
○ Correct use of control statements
○ Logical correctness of output
○ Efficiency and edge case handling
○ Peer feedback on clarity and logic
**Outcome:** Implement arrays and string operations to manage and manipulate data efficiently.
**Activity:** Build a program that stores and arranges student marks in ascending and descending order using arrays and performs string operations like concatenation, comparing, and formatting names.
**Evaluation Method:** Functional Demonstration + Code Walkthrough to check the
○ Correct array and string usage
○ Memory efficiency
○ Handling of invalid inputs
○ Explanation of sorting/searching logic
**Activity:**
● Recursive Problem Solver
Students write a modular program to solve a recursive problem (e.g., factorial, Fibonacci, or Tower of Hanoi) using functions with parameters and return values.
**Evaluation Method:**
● Code Trace + Written Quiz
○ Correct function decomposition
○ Proper parameter passing (by value/reference)
○ Recursion depth and base case handling
○ Quiz on tracing recursive calls

**Outcome:** Utilize pointers and memory operations for effective data handling. Demonstrate competence in dynamic memory allocation and text file processing.
**Activity:** Create a program that dynamically stores user input (e.g., survey responses) using pointers and writes/reads the data to/from a text file.
**Evaluation Method:** Memory Debugging + File I/O Assessment to check the ○ Proper use of malloc, calloc, realloc, and free
○ Pointer arithmetic and dereferencing
○ File creation, reading, writing, and error handling
○ Use of tools like Valgrind or manual memory trace (Optional for Unix flavours)

**SEMESTER-II**

**COURSE3:PROBLEM SOLVING USING C**

**Practical** **Credits:1** **2 hrs/week**

## List of Experiments

1. Write a program to check whether the given number is Armstrong or not.
2. Write a program to find the sum of individual digits of a positive integer.
3. Write a program to generate the first *n* terms of the Fibonacci sequence.
4. Write a program to find both the largest and smallest number in a list of integer values.
5. Write a program to demonstrate change in parameter values while swapping two integer variables using Call by Value & Call by Address.
6. Write a program to perform various string operations.
7. Write a program to search an element in a given list of values.
8. Write a program that uses functions to add two matrices.
9. Write a program to calculate factorial of a given integer value using recursive functions.
10. Write a program for multiplication of two N × N matrices.
11. Write a program to sort a given list of integers in ascending order.
12. Write a program to calculate the salaries of all employees using the Employee (ID, Name, Designation, Basic Pay, DA, HRA, Gross Salary, Deduction, Net Salary) structure.
    a. DA is 30% of Basic Pay
    b. HRA is 15% of Basic Pay
    c. Deduction is 10% of (Basic Pay + DA)
    d. Gross Salary = Basic Pay + DA + HRA
    e. Net Salary = Gross Salary – Deduction
13. Write a program to read/write the data from/to a file.
14. Write a program to reverse the contents of a file and store in another file.
15. Write a program to create Book (ISBN, Title, Author, Price, Pages, Publisher) structure and store book details in a file and perform the following operations:
    a. Add book details
    b. Search a book details for a given ISBN and display book details, if available
    c. Update a book details using ISBN
    d. Delete book details for a given ISBN and display list of remaining books.

**SEMESTER-II**

**COURSE 4: Numerical Methods for Quantum Computing**

**Theory** **Credits: 3** **3 hrs/week**

---

Course Objectives:

1. To understand the foundations of numerical computation and approximation.
2. To analyze different types and sources of numerical errors.
3. To apply numerical methods for solving algebraic and linear systems.
4. To compute numerical differentiation and integration using appropriate algorithms.
5. To solve ordinary differential equations using iterative and Runge-Kutta methods.

Course Learning Outcomes

After completing this course, students will be able to:

1. Explain the significance and limitations of numerical methods in solving engineering problems.
2. Identify and minimize different sources of numerical errors in computation.
3. Apply root-finding and linear system solving methods to practical problems.
4. Perform numerical differentiation and integration accurately.
5. Implement numerical algorithms to solve ordinary differential equations with controlled error.

**Unit–I: Introduction to Numerical Methods and Errors (9 Periods)**

Aim of Numerical Methods, Concept of Numerical Approximation,Measuring Errors: Absolute, Relative, and Percentage Errors, Important Definitions and Theorems Related to Numerical Methods, Round-off and Truncation Errors, Error Propagation and Machine Epsilon, Trade-off between Round-off and Truncation Errors

**Unit–II: Solution of Algebraic and Transcendental Equations (9 Periods)**

Classification of Equations, Bracketing Methods: Bisection and False Position Methods, Open Methods: Fixed-Point Iteration, Newton-Raphson Method, Convergence Analysis and Order of Convergence, Estimation of Errors and Convergence Rate, Backward and Forward Error Analysis

**Unit–III: Systems of Linear Equations (9 Periods)**

Matrix Representation of Linear Systems, Gaussian Elimination and Gauss-Jordan Methods, LU Decomposition and Pivoting, Iterative Methods: Jacobi and Gauss-Seidel Methods, Convergence Criteria and Error Estimation

**Unit–IV: Numerical Differentiation and Integration (9 Periods)**

Numerical Differentiation using Finite Differences, Polynomial Interpolation and Lagrange Interpolation, Newton-Cotes Quadrature Formulas (Trapezoidal, Simpson's Rules), Romberg Integration and Richardson Extrapolation, Gauss Quadrature Methods and Error Estimation

**Unit–V: Numerical Solutions of Differential Equations (9 Periods)**

Initial Value Problems (IVPs), Euler's Method – Stability and Convergence, Runge-Kutta Methods (Second and Fourth Order), Adaptive Methods for IVPs, Systems and Higher-Order Differential Equations, Global and Local Truncation Errors

Text Books:


1. Numerical Methods for Engineering and Data Science (Wuthrich & El Ayoubi, CRC Press, 2025
2. Numerical Methods Fundamentals by R.V.Dukkipati

**SEMESTER-II**

**COURSE 4: Numerical Methods for Quantum Computing**

**Practical** **Credits: 1** **2 hrs/week**

---

**List of Experiments:**

| Exp. No. | Title of Experiment | Objective | Major Tasks / Activities | Software / Tools |
|---|---|---|---|---|
| **1** | Study of Round-off and Truncation Errors | To understand the effect of finite precision and floating-point arithmetic on numerical results. | Perform addition/subtraction of small & large numbers, observe round-off errors, compute machine epsilon (`eps`). | Octave / Python (`sys.float_info.epsilon`) |
| **2** | Error Propagation and Significant Digits | To study how input data errors propagate through calculations. | Evaluate propagation of input errors using addition/subtraction; demonstrate loss of significance. | Octave / Python |
| **3** | Bisection Method for Nonlinear Equations | To find the root of a nonlinear equation using a bracketing method. | Implement Bisection Method for ( f(x)=x^3 - 4x + 1 ); plot error reduction per iteration. | Octave / Python |
| **4** | Newton-Raphson and Fixed Point Iteration | To solve nonlinear equations and compare convergence speed. | Apply Newton-Raphson and Fixed-Point Iteration to ( f(x)=e^{-x}-x ); compare iteration count and error. | Octave / Python |
| **5** | Solving Linear Systems using Gaussian Elimination | To solve a linear system using the direct elimination approach. | Input 3×3 system; perform forward elimination and back substitution; verify with built-in solver. | Octave / Python (`numpy.linalg.solve()`) |
| **6** | LU Decomposition Method | To solve linear systems efficiently using LU factorization. | Decompose matrix ( A = LU ); solve for ( X ) given ( B ); compare with direct methods. | Octave (`lu()`), Python (`scipy.linalg.lu()`) |
| **7** | Polynomial and Lagrange Interpolation | To approximate a function from discrete data points. | Input (x, y) values; generate interpolation polynomial; plot interpolated curve and compare with true function. | Octave / Python (`numpy.polyfit`, `matplotlib`) |

| Exp. No. | Title of Experiment | Objective | Major Tasks / Activities | Software / Tools |
|---|---|---|---|---|
| **8** | Numerical Integration using Trapezoidal and Simpson's Rules | To compute definite integrals numerically and compare accuracy. | Implement both rules for ( f(x)=\sin(x) ) on [0, π]; compare with analytical result. | Octave / Python (`scipy.integrate`) |
| **9** | Euler's Method for Solving First-Order ODE | To solve an initial value problem numerically using Euler's method. | Solve ( \frac{dy}{dx}=y-x^2+1 ); compare results for step sizes h=0.1, 0.05; plot error vs. iteration. | Octave / Python |
| **10** | Runge-Kutta (RK4) Method for IVPs | To apply the 4th order Runge-Kutta method and compare with Euler's. | Solve ( \frac{dy}{dx}=x+y, y(0)=1 ); compute RK4 solution; compare accuracy & convergence with Euler's method. | Octave / Python |

# Yogi Vemana University::Kadapa

## B.Sc. Honours(Quantum Technologies)

**w.e.f. 2025-26 admitted batch**

Recommended Format of Question Paper for all Courses

*Time: 3 Hours* *Max. Marks : 70*

***Section-A***

Answer any FIVE of the following questions. **5X4=20**

1. From unit-I
2. From Unit-I
3. From Unit-II
4. From Unit-II
5. From Unit-III
6. From Unit-III
7. From Unit-IV
8. From Unit-IV
9. From Unit-V
10. From Unit-V

***Section-B***

Answer all the questions from below. **5X10=50**

11. From Unit-I

    or

12. From Unit-I
13. From Unit-II

    (or)

14. From Unit-II
15. From Unit-III

    (or)

16. From Unit-III
17. From Unit-IV

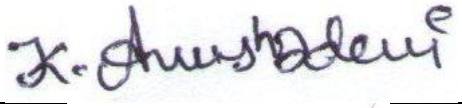
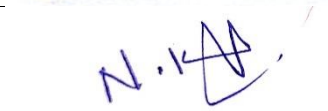
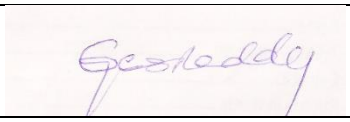
    (or)

18. From Unit-IV
19. From Unit-V

    (or)

20. From Unit-V

# Board of Studies for Computer Science and Allied Courses, Computer Applications

| | | |
|---|---|---|
| Chair Person | Sri.G.Dayanandam |  |
| Member | Smt.K.Anusha Devi |  |
| Member | Smt.N.Kiranmai |  |
| Member | Sk.Abjal Jeelani Basha |  |
| Member | Sri.Chandra Sekhar Reddy |  |
| University Nominee | Dr.M.Reddaiah |  |