Scheme of Examination for B.CA. (AL/ML)

Semester: 2nd (w.e.f. Session 2025-26)

Sr. No.	Course/ Paper Code	Title of the Course	Mode of Learning	Course Type	Credit			Contact Hours per Week			Examination Scheme				Total Marks
					Theory	Practical / Seminar	Total	Theory	Practical /Seminar	Total	End Semester Exam	Internal Assessment	Practical (External)	Practical (Internal)	
1	25BCAIML-N-201	Database Management system	CL/ BL	CC	3	1	4	3	2	5	50	20	20	10	100
2	25BCAIML-N-202	Object Oriented Programming Using C++	CL/ BL	CC	3	1	4	3	2	5	50	20	20	10	100
3	25BCAIML-N-203	Software Engineering	CL/ BL	CC	4	..	4	4	..	4	70	30	..	..	100
4	25BCAIML-N-204	Mathematical Foundation of Computer Science	CL/ BL	MIC	3	..	3	3	..	3	50	20	..	..	70
5	24UN-BBA-MDC201	Introduction to Entrepreneur ship Development	CL/ BL	MDC	3	..	3	3	..	3	50	20	..	..	70
6	24UN-ENG-AEC201	Business Communication	CL/BL	AEC	2	..	2	2	..	2	35	15	..	..	50
7	25BCAIML-SEC-201	Front End Engineering	CL/ BL	SEC	2	1	3	2	2	4	35	15	15	5	70
8	24UN-PSY-VAC101	Human Values and Ethics	CL/ BL	VAC	2	..	2	2	..	2	35	15	..	..	50
	Total						**25**			**28**					**610**

25 BCAIML-N-202	**Object Oriented Programming Using C++**	3L:0T:0P	3 Credits

Maximum Marks:7 0
External Examination: 50
Internal Assessment: 20
Max. Time: 3 Hrs

Course Objectives

- To learn about OOPS concepts and linking C++ as a powerful OOPS language.

Note: There shall be nine questions in all. Question no. 1 shall be compulsory, consisting of eight short answer type questions covering the entire syllabus. Two questions will be asked from each unit. Students will have to attempt one question from each unit. Each question shall carry equal marks.

Unit I

Introduction to Object Oriented Programming - Basic concept of OOP, Comparison of Procedural Programming and OOP, Benefits of OOP, C++ compilation, Abstraction, Encapsulation, Inheritance, Polymorphism, Review of C, Difference between C and C++ - cin, cout, new, delete, operators. Program Structure of the C++ program, Namespace.

Warm-up with C++ Syntax: Decision and Control Structures: if statement, if-else statement, switch statement, Loop: while, do-while, for; Jump statements: break, continue, go to.

Functions - main() function, components of function: prototype, function call, definition, parameter; passing arguments; types of function, inline function, function overloading.

Unit II

Introduction to Classes and Objects - Classes in C++, class declaration, declaring objects, Defining Member functions, Inline member function, Array of objects, Objects as function argument, Static data member and member function, Friend function and friend class.

Constructors and Destructors - Constructors, Instantiation of objects, Default constructor, Parameterized constructor, Copy constructor and its use, Destructors, Constraints on constructors and destructors, Dynamic initialization of objects.

Unit III

Operator Overloading - Overloading unary operators: Operator keyword, arguments and return value; overloading unary and binary operators: arithmetic operators, manipulation of strings using operators; Type conversions.

Inheritance - Derived class and base class: Defining a derived class, Accessing the base class member, Inheritance: multilevel, multiple, hierarchical, hybrid; Virtual base class, Abstract class

Virtual Functions and Polymorphism - Virtual functions, pure virtual functions; Polymorphism, Categorization of polymorphism techniques: Compile time polymorphism, Runtime polymorphism

Unit IV

File Handling - File classes, Opening and Closing a file, File modes, Manipulation of file pointers, Functions for I/O operations.

Exception handling- Try, throw, and catch.

Standard Template Library- Fundamental idea about string, iterators, hashes, iostreams and other types.

Course Outcomes

A student will be able to:

- To understand basic concept of C++
- To acquire the knowledge of C++ operators, hierarchy and precedence and various control structures.
- To learn to use arrays and string in C++ programs.
- To get familiar with OOPS concepts with C++.

Suggested Readings:

1. Bjarne Stroustrup, The C++ Programming Language, Pearson Education
2. Balaguruswami, E., Object Oriented Programming In C++, Tata McGraw-Hill.
3. Richard Johnson, An Introduction to Object-Oriented Application Development, Thomson Learning.

Lab- I: (Based on 25BCAI-N-202) Object Oriented Programming Using C++	0L:0T:2P	1 Credits

Maximum Marks:30
External Examination: 20
Internal Assessment: 10
Max. Time: 3 Hrs

Course Outcomes: By the end of the course the student will be able to:

- Develop problem solving skills coupled with top-down design principles.
- Develop programs in C++ by using various conditional and looping structures.
- Develop programs in C++ using Class and Objects Concepts.
- Work on some real projects used in Industry nowadays.

List of Practical:

1. Raising a number n to a power p is the same as multiplying n by itself p times. Write a function called power( ) that takes a double value for n and an int value for p, and returns the result as double value. Use a default argument of 2 for p, so that if this argument is omitted, the number will be squared. Write a main ( ) function that gets values from the user to test this function.
2. A point on the two dimensional plane can be represented by two numbers: an X coordinate and a Y coordinate. For example, (4,5) represents a point 4 units to the right of the origin along the X axis and 5 units up the Y axis. The sum of two points can be defined as a new point whose X coordinate is the sum of the X coordinates of the points and whose Y coordinate is the sum of their Y coordinates. Write a program that uses a structure called point to model a point. Define three points, and have the user input values to two of them. Then set the third point equal to the sum of the other two, and display the value of the new point. Interaction with the program might look like this:

   Enter coordinates for P1: 3 4

   Enter coordinates for P2: 5 7

   Coordinates of P1 + P2 are : 8, 11
3. Create the equivalent of a four function calculator. The program should request the user to enter a number, an operator, and another number. It should then carry out the specified arithmetical operation: adding, subtracting, multiplying, or dividing the two numbers. (It should use a switch statement to select the operation). Finally it should display the result. When it finishes the calculation, the program should ask if the user wants to do another calculation. The response can be 'Y' or 'N'. Some sample interaction with the program might look like this.

   Enter first number, operator, and second number: 10/ 3

   Answer = 3.333333

   Do another (Y/ N)? Y

   Enter first number, operator, second number 12 + 100

Answer = 112

Do another (Y/ N) ? N

4. A phone number, such as (212) 767-8900, can be thought of as having three parts: the area code (212), the exchange (767) and the number (8900). Write a program that uses a structure to store these three parts of a phone number separately. Call the structure phone. Create two structure variables of type phone. Initialize one, and have the user input a number for the other one. Then display both numbers. The interchange might look like this:
   - Enter your area code, exchange, and number: 415 555 1212
   - My number is (212) 767-8900
   - Your number is (415) 555-1212

5. Create two classes DM and DB which store the value of distances. DM stores distances in metres and centimeters and DB in feet and inches. Write a program that can read values for the class objects and add one object of DM with another object of DB. Use a friend function to carry out the addition operation. The object that stores the results maybe a DM object or DB objects, depending on the units in which the results are required. The display should be in the format of feet and inches or metres and cenitmetres depending on the object on display.
6. Create a class rational which represents a numerical value by two double values- NUMERATOR and DENOMINATOR. Include the following public member Functions:
   constructor with no arguments (default).
   constructor with two arguments.
   void reduce( ) that reduces the rational number by eliminating the highest common factor between the numerator and denominator.
   Overload + operator to add two rational number.
   Overload >> operator to enable input through cin.
   Overload << operator to enable output through cout.
   Write a main ( ) to test all the functions in the class.

7. Consider the following class definition

```
class father {
protected : int age;
public;
father (int x) {age = x;}
virtual void iam ( )
{ cout<< "I AM THE FATHER, my age is : "<< age<< end1:}
};
```

   Derive the two classes son and daughter from the above class and for each, define iam ( ) to write our similar but appropriate messages. You should also define suitable constructors for these classes. Now, write a main ( ) that creates objects of the three classes and then calls iam ( ) for them. Declare pointer to father. Successively, assign addresses of objects of the two derived classes to this pointer and in each case, call iam ( ) through the pointer to demonstrate polymorphism in action.

8. Write a program that creates a binary file by reading the data for the students from the terminal. The data of each student consist of roll no., name (a string of 30 or lesser no. of characters) and marks.
9. A hospital wants to create a database regarding its indoor patients. The information to store include
   Name of the patient
   Date of admission
   Disease

Date of discharge

Create a structure to store the date (year, month and date as its members). Create a base class to store the above information. The member function should include functions to enter information and display a list of all the patients in the database. Create a derived class to store the age of the patients. List the information about all the to store the age of the patients. List the information about all the pediatric patients (less than twelve years in age).

10. Make a class Employee with a name and salary. Make a class Manager inherit from Employee. Add an instance variable, named department, of type string. Supply a method to String that prints the manager's name, department and salary. Make a class Executive inherits from Manager. Supply a method to String that prints the string "Executive" followed by the information stored in the Manager superclass object. Supply a test program that tests these classes and methods.
11. Imagine a tollbooth with a class called toll Booth. The two data items are a type unsigned int to hold the total number of cars, and a type double to hold the total amount of money collected. A constructor initializes both these to 0. A member function called payingCar ( ) increments the car total and adds 0.50 to the cash total. Another function, called nopayCar ( ), increments the car total but adds nothing to the cash total. Finally, a member function called displays the two totals. Include a program to test this class. This program should allow the user to push one key to count a paying car, and another to count a nonpaying car.
12. Write a function called reversit ( ) that reverses a string (an array of char). Use a for loop that swaps the first and last characters, then the second and next to last characters and so on. The string should be passed to reversit ( ) as an argument. Write a program to exercise reversit ( ). The program should get a string from the user, call reversit ( ), and print out the result. Use an input method that allows embedded blanks. Test the program with Napoleon's famous phrase, "Able was I ere I saw Elba)".
13. Create some objects of the string class, and put them in a Deque. Some at the head of the Deque and some at the tail. Display the contents of the Deque using the forEach ( ) function and a user written display function. Then search the Deque for a particular string, using the first That ( ) function and display any strings that match. Finally remove all the items from the Deque using the getLeft ( ) function and display each item. Notice the order in which the items are displayed: Using getLeft ( ), those inserted on the left (head) of the Deque are removed in "last in first out" order while those put on the right side are removed in "first in first out" order. The opposite would be true if getRight ( ) were used.
14. Assume that a bank maintains two kinds of accounts for customers, one called as savings account and the other as current account. The savings account provides compound interest and withdrawal facilities but no cheque book facility. The current account provides cheque book facility but no interest. Current account holders should also maintain a minimum balance and if the balance falls below this level, a service charge is imposed.

    Create a class account that stores customer name, account number and type of account. From this derive the classes

    cur_acct and sav_acct to make them more specific to their requirements. Include necessary member functions in order to achieve the following tasks:

    Accept deposit from a customer and update the balance.

    Display the balance.

    Compute and deposit interest.

    Permit withdrawal and update the balance.

    Check for the minimum balance, impose penalty, necessary and update the balance.

    Do not use any constructors. Use member functions to initialize the class members.
15. Create a base class called shape. Use this class to store two double type values that could be used to compute the area of figures. Derive two specific classes called triangle and rectangle from the base shape. Add to the base class, a member function get_data( ) to initialize baseclass data members and another member function display_area( ) to compute and display the area of figures. Make display_area ( ) as a virtual function and redefine this function in the derived classes to suit their requirements. Using these

three classes, design a program that will accept dimensions of a triangle or a rectangle interactively and display the area.
Remember the two values given as input will be treated as lengths of two sides in the case of rectangles and as base and height in the case of triangles and used as follows:

Area of rectangle = x * y

Area of triangle = ½ * x * y

24BCAIML - SEC -201	Front End Engineering	2L:0T:0P	2 Credits

Maximum Marks:50
External Examination: 35
Internal Assessment: 15
Max. Time: 3 Hrs

Course Objectives

- To understand advanced tools of web technologies and develop front end applications using JavaScript.

Note: There shall be nine questions in all. Question no. 1 shall be compulsory, consisting of eight short answer type questions covering the entire syllabus. Two questions will be asked from each unit. Students will have to attempt one question from each unit. Each question shall carry equal marks.

Unit I

Introduction to Web, Static and dynamic Pages, Introduction and fast revision to html tags and their usages, HTML Tags.

CSS: Selectors, Specificity, Selector Chaining, Box Model- Height & Width, Margin, Padding, Borders, Margin Collapse, Overflow, visibility.

JavaScript: Introduction and its advantages and disadvantages, Internal and external JS, Introduction to Variables and basics programming concepts in JavaScript, Scope of variables,
conditional statements, loops Arrays accessing array elements using different methods.

Unit II

JSON and localstorage : Functions: Type of function and their scope anonymous functions and arrow functions

Arrays Revisited: Iterators, foreach, filter, map ad creating custom iterators.

Objects: Introduction to Object, managing Objects.

JSON and localstorage: Nested JSON and Different JSON methods, Data transfer/exchange methods using JSON, Advantages and drawbacks of Using localstorage and other possible ways, JSON handling different properties, Different ways of storing and retrieving the JSON

Unit III

DOM: Introduction to DOM, Accessing Elements of HTML using Dom, Various functions for handling DOM, Event Handling, ways of handling the events, Dynamic handling of DOM Elements, Dynamic Creation of different elements of DOM

Async Programming Introduction and its usage, JavaScript methods to perform background tasks, Timeout and Interval,

Async and await, Promises and handling promises, Callback and exceptions.

Unit IV

GIT Version Control System (Cycle, Clone, GIT log, diff, commit, push, pull, Branches, and contributions)

Introduction to React, FED in JavaScript and React, developing code in VS Code and introduction to webpack, Managing webpack and other components, Components in React

Course Outcomes

A student will be able to:

- Students will be able to understand and apply HTML5, CSS3, JavaScript to develop front End.
- Students will have sound practical know-how of using arrays, objects, JSON and local Storage.
- They will be able to manipulate the front-end using DOM manipulation.
- Understand the concept of Front-end development, API's, Async Programming.
- Students will be able to develop Single Page Applications using the component-based approach.

Suggested Readings:

1. Head First HTML and CSS – Elisabeth Robson & Eric Freeman
2. Web Design Playground: HTML & CSS the Interactive Way – Paul McFedries
3. Bootstrap 5 Foundations – Daniel Foreman

Lab-2 :Front End Engineering Lab (Based on 24BCAIML - SEC -201)	0L:0T:4P	2 Credits

Maximum Marks:20
External Examination: 15
Internal Assessment: 5

Course Outcomes: By the end of the course the student will be able to:

- To practically apply HTML5, CSS3, JavaScript to develop front End.
- Practically know-how of using arrays, objects, JSON and local Storage.
- To manipulate the front-end using DOM manipulation.
- Use the concept of Front-end development, API's, Async Programming.
- To develop Single Page Applications using the component-based approach of React.

List of Practical:

1. Book Review Blog, you have to create a simple blog page to display the contents of blog with pictures, you have to complete the required tasks.
2. Html Table- basic table generation to get into the understanding, you have to complete the task as defined
3. College Admission Form, create a basic admission form using html forms to capture the admission information from the user.
4. Share Your Recipe, your task is to display the contents of the Food Recipe with various components like ingredients, preparations time.
5. Chess Board Write a program in js that creates a chessboard that represent a grid using
 to separate lines. At each position of the grid there is either a " * " or a "#".
6. Calculate Bill, You have been given 2 arrays of objects (menu and categories) and another json to have bill, from these various object we need to calculate the Bill and generate a JSON.
7. To Do App – Basic Application, you have to build a basic to do app to manage the tasks and complete the user story as defined.
8. Discussion App - D1 Basic Discussion app to include questions and responses to all the questions, application will display questions on the left pane and their response on right pane on the click of question, till then displays the new question form to add new question.
9. Build Pomodoro Clock – Pomodoro is a time management technique developed in the 1980s which uses a timer to break down work into intervals, traditionally 25 minutes in length, separated by short breaks. In this assignment, you'll know how to create such a timer in the browser with JavaScript.
10. Web API, in this assignment, you need Build a basic app to fetch the records using API's with the help of AJAX. This assignment uses the GIThubapi to access data using API.
11. Build a basic app to compile coding questions using API's with the help of AJAX. In this assignment you need to create a compiler app using the codequotientapi and follow the tasks.

25BCAIML-N-201	**Database Management System**	3L:0T:0P	3 Credits

Maximum Marks:70
External Examination: 50
Internal Assessment: 20
Max. Time: 3 Hrs

Course Objectives

- Understand the relational database design principles. Familiar with the basic issues of transaction processing and concurrency control.
- Familiar with SQL queries, database storage structures and access techniques.

Note: There shall be nine questions in all. Question no. 1 shall be compulsory, consisting of eight short answer type questions covering the entire syllabus. Two questions will be asked from each unit. Students will have to attempt one question from each unit. Each question shall carry equal marks.

UNIT – I

Basic Concepts – Data, Information, Records and files. Traditional file – based Systems-File Based Approach-Limitations of File Based Approach, Database Approach-Characteristics of Database Approach, Database Management System (DBMS), Components of DBMS Environment, DBMS Functions and Components, Advantages and Disadvantages of DBMS, Roles in the Database Environment - Data and Database Administrator, Database Designers, Applications Developers and Users.

Data Models: Records- based Data Models, Object-based Data Models, Physical Data Models and Conceptual Modeling.

UNIT – II

Database System Architecture – Three Levels of Architecture, External, Conceptual and Internal Levels, Schemas, Mappings and Instances, Data Independence – Logical and Physical Data Independence, Normalization with various normal forms.

Entity-Relationship Model – Entity Types, Entity Sets, Attributes Relationship Types, Relationship Instances and ER Diagrams.

Relational Data Model - Terminology in Relational Data Structure, Relations, Properties of Relations, Keys, Domains, Integrity Constraints over Relations, Base Tables and Views, Basic Concepts of Hierarchical and Network Data Model.

UNIT – III

SQL fundamentals, data types, DDL, DML, DCL, Data Constraints, working with Tables, Defining different constraints on the table, Defining Integrity Constraints in the ALTER TABLE Command, Select Command, Logical Operator, Range Searching, Pattern Matching, Oracle Function, Grouping data from Tables in SQL, Manipulation Data in SQL.

UNIT – IV

Joining Multiple Tables (Equi Joins), Joining a Table to itself (self Joins), Sub queries Union, intersect & Minus Clause, Creating view, Renaming the Column of a view

Basics of PL/SQL: PL/SQL - Introduction and advantages Understanding, PL/SQL Block structure, Fundamentals of PL/SQL Language - data types, variables, constants and expressions, Operators, Conditional statement, Controlling loop iterations

Course Outcomes

A student will be able to:

- Gain knowledge of database systems and database management systems software.
- Ability to model data in applications using conceptual modeling tools such as ER Diagrams and design database schemas based on the model.
- Formulate, using SQL, solutions to a broad range of query and data update problems.
- Demonstrate an understanding of normalization theory and apply such knowledge to the normalization of a database.
- Be acquainted with the basics of transaction processing and concurrency control.

Suggested Readings:

1. Database System Concepts, 7th Edition, Silberschatz, Korth and Sudarshan, McGraw-Hill. Indian Edition released 2021 .
2. Fundamentals of Database Systems, 7th Edition, Elmasri and Navathe, Pearson Pubs, 2017 .
3. Principles of Database Management, Lemahieu, Broucke and Baesens, Cambridge University Press, 2018.

Lab-2 :Database Management Lab (Based on 25BCAIML-N-201)	0L:0T:2P	1 Credits

Maximum Marks:30
External Examination: 20
Internal Assessment: 10

Course Outcomes: By the end of the course the student will be able to:

- To practically work with SQL Server and Oracle Database Server.
- To work with PL/SQL
- Implement triggers and stored procedures to handle complex business data.

List of Practical:

1. To install the SQL Server.
2. The STUDENT detail database has a table with the following attributes. The primary keys are underlined. STUDENT(regno: int, name: string, dob: date, marks: int, city: string)
   i) Create the above table.
   ii) Remove the city attribute from the table.
   iii) Change the date type of regno from integer to string.
   iv) Add a new attribute phoneno to the existing table.
   v) Enter five tuples into the table.
   vi) Display all the tuples in the student table.
3. A LIBRARY database has a table with the following attributes.
   LIBRARY(bookid:int, title:string, author:string, publication:string, yearpub:int, price:real)
   i) Create the above table.
   ii) Enter the five tuples into the table
   iii) Display all the tuples in the library table.
   iv) Display the different publishers from the list.
   v) Arrange the tuples in the alphabetical order of the book titles.
   vi) List the details of all the books whose price ranges between Rs. 100 and Rs. 300
4. The SALARY database of an organization has a table with the following attributes.
   EMPSALARY(empcod:int, empname:string, dob:date, department:string, salary:real)
   i) Create the above table.
   ii) Enter the five tuples into the table
   iii) Display all the number of employees working in each department.
   iv) Find the sum of the salaries of all employees.
   v) Find the sum and average of the salaries of employees of a particular department.
   vi) Find the least and highest salaries that an employee draws.
5. Consider the insurance database given below. The primary keys are underlined and the data types are specified.
   PERSON(driver-id-no: string, name: string, address:strong)
   CAR(regno: string, model: string, year: int)
   ACCIDENT(report-no: int, date: date, location: String)
   OWNS(driver-id-no: string, regno: string)
   PARTICIPATED(driver-id-no: string, regno: string, report-no: int, damage-amount: int)
   i) Create the above tables by properly specifying the primary keys and the foreign keys
   ii) Enter at least five tuples for each relation.
   iii) Demonstrate how you
   a) Update the damage amount for the car with a specific regno in the accident with report no 12 to 25000.
   b) Add a new accident to the database.
   iv) Find total number of people who owned cars that were involved in accidents in 2002

v) Find the number of accidents in which cars belonging to a specific model were involved

6. Consider the following database of students enrollment in courses and books adopted for each course.
   STUDENT(regno: string, name: string, major: strong, bdate: date)
   COURSE(course-no: intcname: string, dept: string)
   ENROLL(reg-no: string, course-no: int, sem: int, marks: int)
   BOOK-ADOPTION(course-no: int, sem: int, book-isbn: int)
   TEXT(book-isbn: int, book-title: string, publisher: string, author: string)
   i) Create the above tables by properly specifying the primary keys and the foreign keys
   ii) Enter atleast five tuples for each relation.
   iii) Demonstrate how you add a new text book to the database and make this book be adopted by some department.
   iv) Produce a list of textbooks (include Course-no, book-isbn, book-title) in the alphabetical order for courses offered by the 'Compute Science' department that use more than two books.
   v) List any department that has all its adopted books published by a specific publisher.

7. The following tables are maintained by a book dealer
   AUTHOR(author-id: int, name: string, city: string, country: string)
   PUBLISHER(publisher-id: int name: string, city: string, country: string)
   CATLOG(book-id: int, title : string, author-id: int, publisher-id: int, category: int, year: int, price: int)
   CATEGORY(category-id: int, description: string)
   ORDER-DETAILS(order-no: int, book-id: int, quantity: int)
   i) Create above tables by properly specifying the primary keys and the foreign keys.
   ii) Enter atleast five tuples for each relation.
   iii) Give the details of the authors who have 2 or more books in the catalog and the price of the books is greater than the average price of the books in the catalog and the year of publication is after 2010.
   iv) Find the author of the book which has maximum sales.
   v) Demonstrate how to increase price of books published by specific publisher by 10%

8. Consider the following database for BANK.
   BRANCH(branch-name: string, branch-city: string, assets: real)
   ACCOUNT(accno: int, branch-name: string, balance: real)
   DEPOSITOR(customer-name: string, accno: int)
   CUSTOMER(customer-name: string, customer-street: string, customer-city: string)
   LOAN(loan-no: int, branch-name: string, amount: real)
   BORROWER(customer-name: string, loan-no: int)
   i) Create the above tables by properly specifying the primary keys and foreign keys.
   ii) Enter atleast five tuples for each relation.
   iii) Find all the customers who have atleast two accounts at the main branch.
   iv) Find all customers who have an account at all the branches located in a specific city.
   v) Demonstrate how t0 delete all account tuples at every branch located in a specific city.

9. Consider the following database for ORDER PROCESSING.
   CUSTOMER(cust-no: int, cname: string, city: string)
   ORDER(orderno: int, odate: date, ord-amt: real)
   ORDER_ITEM(orderno: int, itemno:int, qty: int)
   ITEM(itemno: int, unitprice: real)
   SHIPMENT(orderno: int, warehouseno: int, ship-date: date)
   WAREHOUSE(warehouseno: int, city: string)
   i) Create the above tables by properly specifying the primary keys and the foreign keys
   ii) Enter atleast five tuples for each relation.
   iii) List the order number and ship date for all orders shipped from particular warehouse
   iv) Produce a listing: customer name, no of orders, average order amount
   v) List the orders that were not shipped within 30 days of ordering

10. Create two tables: motherboard (motherboard_Name, Price) and processor (processor_Name, Price) and insert 3 records in each table. Now write a query to display all maximum combinations of motherboard and processor with their price.
11. Write a PL/SQL program to find the largest of three numbers
12. Write a PL/SQL program to find the factorial of a number..

25BCAIML-N-203	Software Engineering	3L:0T:0P	3 Credits

Maximum Marks:100
External Examination: 70
Internal Assessment: 30
Max. Time: 3 Hrs

Course Objectives

- The objective of this course is to educate the students about the different models of software development and metrics used in software engineering.
- Students will get knowledge of Software Testing.

Note: There shall be nine questions in all. Question no. 1 shall be compulsory, consisting of eight short answer type questions covering the entire syllabus. Two questions will be asked from each unit. Students will have to attempt one question from each unit. Each question shall carry equal marks.

UNIT – I

Introduction: Program vs. Software, Software Engineering, Programming paradigms, Software Crisis – problem and causes, Phases in Software development: Requirement Analysis, Software Design, Coding, Testing, Maintenance, Software Development Process Models: Waterfall, Prototype, Evolutionary and Spiral models, Role of Metrics.

UNIT – II

Feasibility Study, Software Requirement Analysis and Specifications: SRS, Need for SRS, Characteristics of an SRS, Components of an SRS, Problem Analysis, Information gathering tools, Organizing and structuring information, Requirement specification, validation and Verification.

UNIT – III

Structured Analysis and Tools: Data Flow Diagram, Data Dictionary, Decision table, Decision trees, Structured English, Entity-Relationship diagrams, Cohesion and Coupling. Gantt chart, PERT Chart, Software Maintenance: Type of maintenance, Management of Maintenance, Maintenance Process, maintenance characteristics.

UNIT – IV

Software Project Planning: Cost estimation: COCOMO model, Project scheduling, Staffing and personnel planning, team structure, Software configuration management, Quality assurance plans, Project monitoring plans, Risk Management. Software testing strategies: unit testing, integration testing, Validation testing, System testing, Alpha and Beta testing.

Course Outcomes

A student will be able to:

- Understand concept of Software Engineering and various SDLC; DSE-III(III).
- Learn to calculate the cost of a software project for an Enterprise; DSE-III(III).
- Gather basic information of an enterprise to design a software; DSE-III(III).
- Understand the fundamentals of Software Testing and Software Maintenance.

Suggested Readings:

1. Pressman S. Roger, Software Engineering, Tata McGraw Hill.
2. Jalote Pankaj, An Integrated Approach to Software Engineering, Narosa Publ. House.
3. K. K. Aggarwal, Yogesh Singh, Software Engineering, New Age International.
4. Sommerville, Software Engineering, Pearson Education.
5. Fairley Richard, Software Engineering Concepts, Tata Mc-Graw Hill Ed.
6. Rajib Mall, Fundamentals of Software Engineering, PHI Learning.

25BCAIML-N-204	Mathematical Foundation of Computer Science	3L:0T:0P	3 Credits

Maximum Marks:70
External Examination: 50
Internal Assessment: 20
Max. Time: 3 Hrs

Course Objectives

- This course helps the students to demonstrate various principles involved in solving mathematical problems and thereby reducing the time taken for performing job functions and solving problems.

Note: There shall be nine questions in all. Question no. 1 shall be compulsory, consisting of eight short answer type questions covering the entire syllabus. Two questions will be asked from each unit. Students will have to attempt one question from each unit. Each question shall carry equal marks.

Unit I

Heights & Distances: Interest, Heights and Distances, Profit and Loss, Partnership, Mixture and Allegation, Time and distance Series, Time & Work, The Data Interpretation part covers Tabulation., Calendar, Clocks, Heights & Distances

Unit II

Number Problems: Numbers, H.C.F. & L.C.M. of Numbers, Decimal Fractions, Simplification, Square Roots & Cube Roots, Whole numbers problems, Decimals problems, Problems on Trains

Unit III

Database Modeling: Fractions problems, Numbers and Ages, Percentage problems, Boats and Streams, Ratio & Proportion, Square roots, Averages.

Unit IV

Statistics and Correlation: Meaning, Scope and Limitations of Statistics, Types and Sources of Data, Methods and Problems of Collection of Primary and Secondary Data

Measures of Central Tendency (Arithmetic Mean, Median, Partition Values, Mode),Measure of Dispersion (Absolute and Relative Measures Range, Quartile Deviation, Mean Deviation, Standard Deviation and Coefficient of Variation)

Correlation: Definition, Scatter diagram, Karl Pearson's coefficient of correlation, Numerical Problems for determination of Correlation coefficient. Regression: Definition, dependent and independent variables, least Square method only, Numerical problems.

Permutations & Combinations, Probability,

Course Outcomes

A student will be able to:

- Common methodologies of solving questions related to Quantitative Aptitude and basics of Statistics.
- Learn the various question types of aptitude and their basic approaches.
- Create a bridge for the advanced levels of aptitude and prepare a base for problem solving and employability of the student.

Suggested Readings:

1) Quantitative Aptitude for Competitive Examinations by R. S. Aggarwal.
2) Business Statistics by TR Jain and SC Aggarwal