SCHEME OF EXAMINATION

B. Tech- 2nd Year — Semester: 3th

Branch: Computer Science and Business Systems

S.N.	Subject Name	Subject Code	Periods per week			Scheme of Exam			Total Marks	Credit L+(T+P)/2
						Theory/Practical				
			L	T	P	ESE	CT	TA		
1	Formal Language and Automata Theory	CS112301	3	1	-	80	20	25	125	3
2	Computer Organization and Architecture	CS112302	3	1	-	80	20	25	125	3
3	Object Oriented Programming	CS112303	3	1	-	80	20	25	125	3
4	Computational Statistics	AM112304	3	1	-	80	20	25	125	3
5	Database Management Systems	CS112305	3	0	-	80	20	25	125	2
6	Computer Organization and Architecture Lab	CS112391		-	2	40	-	20	60	1
7	Object Oriented programming Lab	CS112392		-	2	40	-	20	60	1
8	Computational Statistics Lab	CS112393		-	2	40	-	20	60	1
9	Database Management Systems Lab	CS112394		-	2	40	-	20	60	1
10	Mini Project	CS112396	-	-	-	100		25	125	2
11	Indian constitution	HM100395	-	-	-	-	-	10	10	-
Total			15	4	8	660	100	240	1000	20

Note:

L-Lecture T-Tutorial P-Practical

CT-ClassTest TA-TeachersAssessment ESE-EndSemesterExam

Subject Code CS112301	Formal Language and Automata Theory	L =	T = 0	P = 0	Credits =
Examination Scheme	ESE	CT	TA	Total	ESE Duration
	100	20	30	150	3 Hours
	Minimum number of class tests to be conducted=02			Minimum Assignments=02	

Course Objectives	Course Outcomes
Formal languages and automata theory deals with the concepts of automata, formal languages, Grammar, computability and decidability. The reasons to study Formal Languages and Automata Theory are Automata Theory provides a simple, elegant view of the complex machine that we call a computer. More precisely, the objectives are: • To give an overview of the theoretical foundations of computer science from the perspective of formal languages. • To illustrate finite state machines to solve problems in computing. • To explain the hierarchy of problems arising in the computer sciences. • To familiarize Regular grammars, context frees grammar. • To solve various problems of applying normal form techniques, push down automata and Turing Machines	On successful completion of the course, the student will be able to: **CO1**.Design finite automata to accept a set of strings of a language. **CO2**.Determine whether the given language is regular or not. . **CO3**.Design context free grammars to generate strings of context free language. **CO4**.Design push down automata and the equivalent context free grammars and Design Turing machine. **CO5**.Distinguish between computability and non-computability, Decidability and un-decidability.

UNIT – I Introduction: Alphabet, languages and grammars, productions and derivation, Chomsky hierarchy of languages. **Regular languages and finite automata:** Regular expressions and languages, deterministic finite automata (DFA) and equivalence with regular expressions, nondeterministic finite automata (NFA) and equivalence with DFA, regular grammars and equivalence with finite automata, properties of regular languages, *Kleene's theorem,* pumping lemma for regular languages, *Myhill-Nerode theorem and its uses,* minimization of finite automata.

UNIT – II Context-free languages and pushdown automata: Context-free grammars (CFG) and languages (CFL), Chomsky and Greibach normal forms, nondeterministic pushdown automata (PDA) and equivalence with CFG, parse trees, ambiguity in CFG, pumping lemma for context-free languages, deterministic pushdown automata, closure properties of CFLs. **Context-sensitive languages:** Context-sensitive grammars (CSG) and languages, linear bounded automata and equivalence with CSG.

UNIT – III: Turing machines: The basic model for Turing machines (TM), Turing recognizable(recursively enumerable) and Turing-decidable (recursive) languages and their closure properties, variants of Turing machines, nondeterministic TMs and equivalence with deterministic

TMs, unrestricted grammars and equivalence with Turing machines, TMsas enumerators.

UNIT – IV Undecidability: Church-Turing thesis, universal Turing machine, the universal and diagonalization languages, reduction between languages and Rice s theorem, undecidable problems about languages.

UNIT – V: Basic Introduction to Complexity: Introductory ideas on Time complexity of deterministic and nondeterministic Turing machines, P and NP, NP- completeness, Cook's Theorem, other NP -Complete problems.

Text Books:

S. No.	Title	Author(s)	Publisher
1	Introduction to Automata Theory, Languages, and Computation	John E. Hopcroft, Rajeev Motwani and Jeffrey D. Ullman.	Addison-Wesley

Reference Books:

S. No.	Title	Author(s)	Publisher
1	Elements of the Theory of Computation	Harry R. Lewis and Christos H. Papadimitriou.	
2	Automata and Computability	Dexter C. Kozen	
3	Introduction to the Theory of Computation	Michael Sipser	

Subject Code CS112302	COMPUTER ORGANIZATION & ARCHITECHTURE	L =	T = 0	P = 0	Credits =
Examination Scheme	ESE	CT	TA	Total	ESE Duration
	100	20	30	150	3 Hours
	Minimum number of class tests to be conducted=02			Minimum Assignments=02	

Course Objectives	Course Outcomes
1. To understand the structure, function and characteristics of computer systems. 2. To understand the design of the various functional units and components of computers. 3. To identify the elements of modern instructions sets and their impact on processor design. 4. To explain the function of each element of a memory hierarchy 5. To identify and compare different methods for computer I/O.	1. Identify the basic hardware components of a computer system. 2. Familiarize themselves with binary and hexadecimal number systems including computer arithmetic. 3. Familiarize themselves with functional units of the processor such as the register file and arithmetic logical unit. 4. Understand basics functionality of systems: parallel, pipelined, superscalar and RISC/CISC architectures. **5.** Represent system design in appropriate formats; addressing modes, an instruction sets as per the system configuration requirements.

UNIT – I Introduction: Basics in Boolean logic and Combinational/Sequential Circuits. **Functional blocks of a computer:** CPU, memory, input-output subsystems, control unit. **Instruction set architecture of a CPU:** Registers, instruction execution cycle, RTL interpretation of instructions, addressing modes, instruction set. Outlining instruction sets of some common CPUs. **Data representation:** Signed number representation, fixed and floating point representations, character representation.

UNIT – II Computer arithmetic: Integer addition andsubtraction, ripple carry adder, carry look-ahead adder, etc. multiplication – shift-and-add,Booth multiplier, carry save multiplier, etc. Division restoring and non-restoring techniques, floating point arithmetic, IEEE 754 format.

UNIT – III: Introduction to x86 architecture. **CPU control unit design:** Hardwired and micro-programmed design approaches, design of a simple hypothetical CPU. **Memory system design:** Semiconductor memory technologies, memory organization

UNIT –IV Peripheral devices and their characteristics: Input-output subsystems, I/O deviceinterface, I/O transfers – program controlled, interrupt driven and DMA, privilegedand non-privileged instructions, software interrupts and exceptions. Programs andprocesses – role of interrupts in process state transitions, I/O device interfaces – SCII, USB.

Pipelining: Basic concepts of pipelining, throughput and speedup, pipeline hazards. Parallel Processors: Introduction to parallel processors, Concurrent access to memory and cache coherency

UNIT – V: Memory organization: Memory interleaving, concept of hierarchicalmemory organization, cache memory, cache size vs. block size, mapping functions, replacement algorithms, write policies.

Text Books:

S. No.	Title	Author(s)	Publisher
1	*Computer System Architecture*	M. M. Mano: 3rd ed.,	Prentice Hall of India, New Delhi, 1993.

Reference Books:

S. No.	Title	Author(s)	Publisher
1	*Computer Architecture and Organization*	John P. Hayes.	
2	*Computer Organization and Architecture Designing for Performance*	William Stallings.	
3	*Computer System Design and Architecture*	Vincent P. Heuring and Harry F. Jordan	

Subject Code CS112303	Object Oriented Programming	L =	T = 0	P = 0	Credits =
Examination Scheme	ESE	CT	TA	Total	ESE Duration
	100	20	30	150	3 Hours
	Minimum number of class tests to be conducted=02			Minimum Assignments=02	

Course Objectives	Course Outcomes
1. Understanding about object oriented programming. 2. Gain knowledge about the capability to store information together in an object. 3. Understand the capability of a class to rely upon another class. 4. Learn how to store one object inside another object and use of one method can be used in variety of different ways. 5. Create and process data in files using file I/O functions. 6. Understand about constructors which are special type of functions. 7. Learn how to write code in a way that it is independent of any particular type.	At the end of the course, a student will be able to: 1. Students will understand the concepts of flow of control, abstraction, pointer and recursion. 2. Analyse a simple programming problem specification. 3. Design a high-level solution to the problem using functional abstraction and general imperative programming language constructs. 4. Write, compile, execute and debug a C++ program which maps the high-level design onto concrete C++ programming constructs.

UNIT – I Introduction: Procedural programming, An Overview of C: Types Operator and Expressions, Scope and Lifetime, Constants, Pointers, Arrays, and References, Control Flow, Functions and Program Structure, Namespaces, error handling, Input and Output (*C*-way), Library Functions (*string*, *math*, *stdlib*), Command line arguments, Pre-processor directive

UNIT – II Some difference between C and C++: Single line comments, Local variable declaration within function scope, function declaration, function overloading, stronger type checking, Reference variable, parameter passing – value vs reference, passing pointer by value or reference, #define constant vsconst, Operator new and delete, the typecasting operator,Inline Functions in contrast to macro, default arguments

UNIT – III: The Fundamentals of Object Oriented Programming: Necessity for OOP, Data Hiding, Data Abstraction, Encapsulation, Procedural Abstraction, Class and Object.

More extensions to C in C++ to provide OOP Facilities: Scope of Class and Scope Resolution Operator, Member Function of a Class, private, protected and public Access Specifier, this Keyword, Constructors and Destructors, friend class, error handling (exception)

UNIT –IV Essentials of Object Oriented Programming: Operator overloading, Inheritance – Single and Multiple, Class Hierarchy, Pointers to Objects, Assignment of an Object to another Object, Polymorphism through dynamic binding, Virtual Functions, Overloading, overriding and hiding, Error Handling

UNIT – V: Generic Programming: Template concept, class template, function template, template specialization **Input and Output:** Streams, Files, Library functions, formatted output

Text Books:

S. No.	Title	Author(s)	Publisher
1	*The C++ Programming Language*,	BjarneStroustrup	Addison Wesley
2	*C++ and Object-Oriented Programming Paradigm*,	Debasish Jana	PHI Learning Pvt. Ltd

Reference Books:

S. No.	Title	Author(s)	Publisher
1	*The C++ Programming Language*,	BjarneStroustrup	Addison Wesley
2	*The Design and Evolution of C++*,	BjarneStroustrup,	Addison Wesley.

Subject Code AM112304	Computational Statistics	L =	T = 0	P = 0	Credits =
Examination Scheme	ESE	CT	TA	Total	ESE Duration
	100	20	30	150	3 Hours
	Minimum number of class tests to be conducted=02			Minimum Assignments=02	

Course Objectives	Course Outcomes
• Apply the concepts of probability and distributions to some case studies. Apply C213.2 Apply the concepts of discrete probability distributions. Apply • Apply the concepts of continuous probability distributions. Apply • Assess the sampling theory and making inferences. Eval Correlate the material of one unit to the material in other units. Understand • Resolve the potential misconceptions and hazards in each topic of study. Remember	CO1 Identify the difference between structured program and procedureoriented program Remembering CO2 Develop programs for file handling Understanding CO3 Implementing the concepts of Exceptions Handling in programming Applying CO4 Develop applications for a range of problems using object-oriented programming techniques. Understanding CO5 Apply the concepts of inheritance Applying C215.6 Encapsulation of data in virtual functions.

UNIT – I Introduction: Multivariate Normal Distribution: Multivariate Normal Distribution Functions, Conditional Distribution and its relation to regression model, Estimation of parameters.

UNIT – II Discriminant Analysis: Statistical background, linear discriminant function analysis, Estimating linear discriminant functions and their properties

UNIT – III: Principal Component Analysis: Principal components, Algorithm for conducting principal component analysis, deciding on how many principal components to retain, H-plot.

UNIT –IV Factor Analysis: Factor analysis model, Extracting common factors, determining number of factors, Transformation of factor analysis solutions, Factor scores

UNIT – V: Clustering: Introduction, Types of clustering, Correlations and distances, clustering by partitioning methods, hierarchical clustering, overlapping clustering, K-Means Clustering-Profiling and Interpreting Clusters

Text Books:

S. No.	Title	Author(s)	Publisher
1	*An Introduction to Multivariate Statistical Analysis*	T.W. Anderson.	
2	*Applied Multivariate Data Analysis,*	*Vol I & II,* J.D. Jobson.	

Reference Books:

S. No.	Title	Author(s)	Publisher
1	*Regression Diagnostics , Identifying Influential Data and Sources of Collinearety,*	D.A. Belsey, E. Kuh R.E. Welsch	
2	*Applied Linear Regression Models*	J. Neter, W. Wasserman and M.H. Kutner	
3	*The Foundations of Factor Analysis,*	A.S. Mulaik.	

Subject Code CS112305	Database Management Systems	L =	T = 0	P = 0	Credits =
Examination Scheme	ESE	CT	TA	Total	ESE Duration
	100	20	30	150	3 Hours
	Minimum number of class tests to be conducted=02			Minimum Assignments=02	

Course Objectives	Course Outcomes
• To understand the role of a database management system and its users in an organization. • To understand database concepts, including the structure and operation of the relational data model • Can successfully apply logical database design principles, including E-R diagrams and database normalization. • Construct simple and moderately advanced database queries using Structured Query Language (SQL). • To understand the concept of transaction, its properties and how to persist the data in complex concurrent users environment.	1. Be familiar with basic concepts of RDBMS, Relational data model & be able to write relational algebra expressions for queries; 2. Be familiar with basic database storage structures and access techniques: file and page organizations, indexing methods including B-tree and hashing; 3. Understand DML, DDL and will be able to construct queries using SQL by knowing the importance of data & its requirements in any application. 4. Utilize a database modeling technique for a single entity class, a one-to-one (1:1) relationship between entity classes, a one-to-many (1:M) relationship between entity classes, a many-to-many (M:M) relationship between entity classes, and recursive relationships; 5. Be familiar with the basic issues of transaction, its processing and concurrency control.

UNIT – I Introduction: Introduction: Introduction to Database. Hierarchical, Network and Relational Models. **Database system architecture**: Data Abstraction, Data Independence, Data Definition Language (DDL), Data Manipulation Language (DML). **Data models**: Entity-relationship model, network model, relational and object orienteddata models, integrity constraints, data manipulation operations.

UNIT – II Relational query languages: Relational algebra, Tuple and domain relational calculus,SQL3, DDL and DML constructs, Open source and Commercial DBMS - MYSQL, ORACLE, DB2, SQL server. **Relational database design**: Domain and data dependency, Armstrong's axioms, Functional Dependencies, Normal forms, Dependency preservation, Lossless design.

UNIT – III: Query processing and optimization: Evaluation of relational algebra expressions, Query equivalence, Join strategies, Query optimization algorithms. **Storage strategies**: Indices, B-trees, Hashing.

UNIT –IV Transaction processing: Concurrency control, ACID property, Serializability of scheduling, Locking and timestamp based schedulers, Multi-version and optimistic Concurrency Control schemes, Database recovery.

UNIT – V: Database Security: Authentication, Authorization and access control, DAC, MAC and RBAC models, Intrusion detection, SQL injection. **Advanced topics**: Object oriented and object relational databases, Logical databases, Web databases, Distributed databases, Data warehousing and data mining.

Text Books:

S. No.	Title	Author(s)	Publisher
1	Database System Concepts	Abraham Silberschatz, Henry F. Korth andS. Sudarshan	
2	*Applied Multivariate Data Analysis,*	*Vol I & II,* J.D. Jobson.	

Reference Books:

S. No.	Title	Author(s)	Publisher
1	*Principles of Database and Knowledge*	*Base* Systems, Vol 1 by J. D. Ullman.	
2	*Fundamentals of Database Systems*..	R. Elmasri and S. Navathe	
3	*Foundations of Databases*.	Serge Abiteboul, Richard Hull, Victor Vianu.	

Subject Code CS112391	Computer Organization and Architecture Lab	L =	T = 0	P = 0	Credits =
Examination Scheme	ESE	CT	TA	Total	ESE Duration
	100	20	30	150	3 Hours
	Minimum number of class tests to be conducted=02			Minimum Assignments=02	

1. Circuits on breadboard or simulators

   (a) Implementation of Combinational Digital/Boolean Circuits: Adder, Subtractor, Multiplication Module, Division Module, Multiplexer, Demultiplexer, Encoder, Decoder.

   (b) Implementation of Sequential Circuits: Counters, Linear Feedback Shift Registers (LFSR)

2. C/C++ programming to understand the formats of char, int, float, double, long etc.

3. Machine language programming on x86 or higher version kits or simulators:

   (i) Add/subtract/multiplication/division/GCD/LCM

   (ii) Accessing some specific memory locations/ports

   (iii) Counting odd and even integers from a series of memory locations

   (iv) Printing values of selected registers

   (v) Handing interrupts

Subject Code CS112392	Object Oriented Programming Lab	L =	T = 0	P = 0	Credits =
Examination Scheme	ESE	CT	TA	Total	ESE Duration
	100	20	30	150	3 Hours
	Minimum number of class tests to be conducted=02			Minimum Assignments=02	

Course Objectives	Course Outcomes
The objective of course is to develop programming skills of students, using object oriented programming concepts, learn the concept of class and object using C++ and develop classes for simple applications.	Upon successful completion of this Lab the student will be able to: 1. You will be able to know about Object oriented programming. 2. Use Abstract Data Types in the programs. 3. Application of Non recursive functions. 4. OOP principles like Encapsulation Inheritance Polymorphism were frequently used. 5. Different sorting techniques (Quick sort, Merge sort, Heap sort) were used. 6. Polymorphism and exception handling in the programs.

Laboratory

1. Parameter passing: passing parameter by value vs by reference, passing array as constant pointer
2. Function overloading: writing string operations like strcat and strncat, strcpy and strncpy as overloaded functions.
3. Dynamically allocating space for a pointer depending on input and doing this repeatedly, depending on different inputs and finally de-allocating the pointer.
4. Define class complex with all possible operations: constructor, destructor, copy constructor, assignment operator with the data members stored as pointer to integers.
5. Define class vector of integers with all possible operations like constructor, destructor, copy constructor and assignment operators
6. Define class matrix of integers with all possible operations like constructor, destructor, copy constructor and assignment operators
7. Define class matrix of integers using vector, with all possible operations like constructor, destructor, copy constructor and assignment operators
8. Define class stack, queue, linked-list, array, set using some data-type (int) with data members kept as private and functions kept in both protected and public sections.
9. Define class complex with all possible operators: constructor, destructor, copy constructor, assignment operator and operators >, <, >=, <=, ==, ++ (pre and post), +, +=, ( ), with the data members stored as pointer to integers.
10. Define class vector of integers with all possible operations like constructor, destructor, copy constructor and assignment operators>, <, >=, <=, ==, ++ (pre and post), +, +=, ( )
11. Define class matrix of integers with all possible operations like constructor, destructor, copy constructor and assignment operators>, <, >=, <=, ==, ++ (pre and post), +, +=, ( ).
12. Define class matrix of integers using vector, with all possible operations like constructor, destructor, copy constructor and assignment operators>, <, >=, <=, ==, ++ (pre and post), +,

+=, ( ).

13. Define stack and queue inherited from array class, with standard functions and operators
14. Define a class called 'array' with data type passed as template type with constructor, destructor, copy constructor and assignment operators and index operator.
15. Define template functions for compare and use it in the algorithms like bubble sort, insertion sort, merge sort.
16. Formatted input-output examples
17. Input manipulators
18. Overriding operators <<, >>
19. Define class model for complex number, student class, book class and show it using UML diagram as well as concrete class.
20. Show behavioral modeling through sequence diagram and activity diagram for workflow in a typical log-in, log-out situation.

Subject Code CS112393	Computational Statistics Lab	L =	T = 0	P = 0	Credits =
Examination Scheme	ESE	CT	TA	Total	ESE Duration
	100	20	30	150	3 Hours
	Minimum number of class tests to be conducted=02			Minimum Assignments=02	

Laboratory

Python Concepts, Data Structures, Classes: Interpreter, Program Execution, Statements, Expressions, Flow Controls, Functions, Numeric Types, Sequences and Class Definition, Constructors, Text & Binary Files - Reading and Writing

Data Wrangling: Combining and Merging Datasets, Reshaping and Pivoting, Data Transformation, String Manipulation, Regular Expressions

Data Aggregation, Group Operations, Time series: GoupBy Mechanics, Data Aggregation, Groupwise Operations and Transformations, Pivot Tables and Cross Tabulations, Time Series Basics, Data Ranges, Frequencies and Shifting

Visualization in Python: Matplotlib package, Plotting Graphs, Controlling Graph, Adding Text, More Graph Types, Getting and setting values, Patches

Subject Code CS112394	Database Management Systems Lab	L =	T = 0	P = 0	Credits =
Examination Scheme	ESE	CT	TA	Total	ESE Duration
	100	20	30	150	3 Hours

LIST OF EXPERIMENTS **[12 hrs.]**

1.Database Schema for a customer-sale scenario
Customer (Cust id: integer, Cust name: string)
Item (item_id: integer, item name: string, price: integer)
Sale (bill no: integer, bill data: date, cystoid: integer, item_id: integer, qty_sold: integer)
For the above schema, perform the following—
Create the tables with the appropriate integrity constraints
Insert around 10 records in each of the tables

(a) List all the bills for the current date with the customer names and item numbers.
(b) List the total Bill details with the quantity sold, price of the item and the final amount.
(c) List the details of the customer who have bought a product which has a price>200.
(d) Give a count of how many products have been bought by each customer.
(e) Give a list of products bought by a customer having cust_id as 5.
(f) List the item details which are sold as of today.
(g) Create a view which lists out the bill no, bill date, cust_id, item_id, price, qty_sold, amount.
(h) Create a view which lists the daily sales date wise for the last one week.
(i) Create a view which lists out the bill no, bill date, cust_id, item_id, price, qty_sold, amount.
(j) Create a view which lists the daily sales date wise for the last one week.

2.Database Schema for a Student Library scenario
Student (Stud no: integer, Stud name: string)
Membership (Mem_no: integer, Stud no: integer)
Book(book no: integer, book_name:string, author: string)
Iss_rec (iss_no: integer, iss_date: date, Mem_no: integer, book_no: integer)
For the above schema, perform the following—
(a) Create the tables with the appropriate integrity constraints.
(b) Insert around 10 records in each of the tables.
(c) List all the student names with their membership numbers.
(d) List all the issues for the current date with student and Book names.
(e) List the details of students who borrowed book whose author is CJ DATE.
(f) Give a count of how many books have been bought by each student.
(g) Give a list of books taken by student with stud_no as 5.
(h) List the book details which are issued as of today.
(i) Create a view which lists out the iss_no, iss _date, stud_name, book name.
(j) Create a view which lists the daily issues-date wise for the last one week.

3. Database Schema for a Employee-pay scenario
employee (emp_id: integer, emp_name: string)
department (dept_id: integer, emp_name: string)
pay details (emp_id: integer, dept_id: integer, basic: integer, deductions: integer, additions: integer, DOJ: date)
payroll (emp_id: integer, pay_date: date)
For the above schema, perform the following:

(a) Create the tables with the appropriate integrity constraints
(b) Insert around 10 records in each of the tables
(c) List the employee details department wise
(d) List all the employee names who joined after particular date
(e) List the details of employees whose basic salary is between 10,000 and 20,000
(f) Give a count of how many employees are working in each department
(g) Give a name of the employees whose net salary>10,000
(h) List the details for an employee_id=5
(i) Create a view which lists out the emp_name, department, basic, deductions, net salary
(j) Create a view which lists the emp_name and his netsalary

4. Database Schema for a student-Lab scenario
Student (stud_no: integer, stud_name: string, class: string)
Class (class: string, descript: string)
Lab (mach_no: integer, Lab_no: integer, description: String)
Allotment (Stud_no: Integer, mach_no: integer, dayof week: string)
For the above schema, perform the following—

(a) Create the tables with the appropriate integrity constraints
(b) Insert around 10 records in each of the tables
(c) List all the machine allotments with the student names, lab and machine numbers
(d) List the total number of lab allotments day wise
(e) Give a count of how many machines have been allocated to the _CSIT 'class
(f) Give a machine allotment detail of the stud_no 5 with his personal and class details
(g) Count for how many machines have been allocated in Lab_no 1 for the day of the week as —Monday‖
(h) How many students class wise have allocated machines in the labs?
Create a view which lists out the stud_no, stud_name, mach_no, lab_no, dayofweek
(i) Create a view which lists the machine allotment details for —Thursday

5(a) Write and execute subprogram to find largest number from the given three numbers.
(b) Write and execute subprogram using loop, while and for iterative control statement.

6(a) Write and execute subprogram to check whether the given number is Armstrong or not
(b) Write and execute subprogram to generate all prime numbers below 100.

7(a) Write and execute subprogram to demonstrate the GOTO statement.
(b) Write a subprogram to demonstrate %type and %rowtype attributes

8(a) Write and execute subprogram to demonstrate predefined exceptions
(b) Write and execute subprogram to demonstrate user defined exceptions

9(a) Create a cursor, which displays all employee numbers and names from the EMP table.
(b) Create a cursor, which update the salaries of all employees as per the given data.

10(a) Create a cursor, which displays names of employees having salary > 50000.
(b) Create a procedure to find reverse of a given number

11(a) Create a procedure to update the salaries of all employees as per the given data
(b) Create a procedure to demonstrate IN, OUT and INOUT parameters

12(a) Create a function to check whether given string is palindrome or not.
(b) Create a function to find sum of salaries of all employees working in depart number 10.

13(a) Create a trigger before/after update on employee table for each row/statement.

14 Create a trigger before/after delete on employee table for each row/statement.

15 Create a trigger before/after insert on employee table for each row/statement.

16(a) Create a Form to display employee details using SQL
(b) Create a Report to generate all employee annual salaries….

Additional Programs:

1. Create a Master/details relationship form which perform Add New, Search, Delete, Save and Update on the records
2. Generate a report to calculate employee ‘s salaries department wise from employee table.
3. Create a Report to generate the details of employee table including sum and average salaries department wise.

Reference Books:

S. No.	Title	Authors	Publisher
1	Fundamentals of Database Systems	Elma Sri Nava the	Pearson Education
2	An Introduction to Database systems	C.J. Date, A. Kannan, S. Swami Nadhan,	Pearson, Eight Edition

Subject Code HM100395	Indian constitution	L =	T = 0	P = 0	Credits =
Examination Scheme	ESE	CT	TA	Total	ESE Duration
	100	20	30	150	3 Hours
	Minimum number of class tests to be conducted=02			Minimum Assignments=02	

Course Objectives	Course Outcomes