<table>
<thead>
<tr>
<th>Course Code</th>
<th>Course Title</th>
<th>Teaching Scheme Hours /Week</th>
<th>Examination Scheme</th>
<th>Marks</th>
<th>Credit</th>
</tr>
</thead>
<tbody>
<tr>
<td>IT 2201</td>
<td>Data Structures II</td>
<td>3 0 0</td>
<td>50 50 0 0 0</td>
<td>100</td>
<td>3</td>
</tr>
<tr>
<td>IT 2202</td>
<td>Computer Network</td>
<td>3 1 0</td>
<td>50 50 0 0 0</td>
<td>100</td>
<td>4</td>
</tr>
<tr>
<td>IT 2203</td>
<td>Computer Organization and Architecture</td>
<td>3 1 0</td>
<td>50 50 0 0 0</td>
<td>100</td>
<td>4</td>
</tr>
<tr>
<td>IT 2204</td>
<td>Object Oriented Paradigms *</td>
<td>3 1 0</td>
<td>50 50 0 0 0</td>
<td>100</td>
<td>4</td>
</tr>
<tr>
<td>BSIT 2201</td>
<td>Engineering Mathematics III</td>
<td>3 1 0</td>
<td>50 50 0 0 0</td>
<td>100</td>
<td>4</td>
</tr>
<tr>
<td>IT 2205</td>
<td>Data Structures II Laboratory</td>
<td>0 0 4</td>
<td>0 0 0 50 50</td>
<td>50</td>
<td>2</td>
</tr>
<tr>
<td>IT 2206</td>
<td>Network Laboratory</td>
<td>0 0 2</td>
<td>0 0 0 25</td>
<td>25</td>
<td>1</td>
</tr>
<tr>
<td>IT 2207</td>
<td>Computer Organization and Architecture</td>
<td>0 0 2</td>
<td>0 0 0 25</td>
<td>25</td>
<td>1</td>
</tr>
<tr>
<td>IT 2208</td>
<td>Object Oriented Programming Laboratory</td>
<td>0 0 2</td>
<td>0 0 0 25</td>
<td>25</td>
<td>1</td>
</tr>
<tr>
<td>Total</td>
<td></td>
<td>15 4 10</td>
<td>250 250 0 125</td>
<td>625</td>
<td>24</td>
</tr>
<tr>
<td>Grand Total</td>
<td></td>
<td>29</td>
<td>625 625 625</td>
<td>24</td>
<td></td>
</tr>
</tbody>
</table>
Teaching Scheme:
Lectures: 3 Hrs/Week

Course Objectives:
1. To learn concepts and use of stack and queue data structures.
2. To learn basic tree data structure and traversals with BST
3. To learn graphs, traversals and algorithms on graph data structure.
4. To learn symbol tables and hashing with their applications.
5. To study some advanced tree concepts.
6. To learn different file organizations and their use in practice.

Course Outcomes:
1. Understand different advanced abstract data type (ADT) and data structures and their implementations.
2. Understand use of object oriented programming concepts in using different data structures.
3. Logic building and algorithm implementation for different data structures.
4. Analysis of appropriate use of data structures for the given problem.
5. Understand use of trees, graphs and advance trees for real situations.
6. Implementation and use of different file organization techniques.

Unit – I: Stacks and Queues
Concept of stack, stack as ADT, Implementation of stack using array and linked organization, multistacks, use of stack- Recursion, expression conversion & evaluation.
Concept of queues as ADT, Implementation using array and linked organization, multiqueues, priority queue.

Unit – II: Trees
Difference in linear and non-linear data structure, Trees and binary trees-concept and terminology. Expression tree. Conversion of general tree to binary tree. Binary tree as an ADT. Recursive and non-recursive algorithms for binary tree traversals, Binary search trees, Binary search tree as ADT

Unit – III: Graphs
Graph as an ADT, Representation of graphs using adjacency matrix and adjacency list, Depth First Search and Breadth First Search traversal. Prim’s and Kruskal’s algorithms for minimum spanning tree, shortest path using Warshall’s and Dijkstra’s algorithm.

Unit – IV: Tables
Symbol Table: Symbol Table, Huffman’s algorithm, Heap data structure, applications of heap, Heap sort Hash table: hashing function, collision resolution techniques- linear probing, rehashing, chaining without replacement and chaining with replacement.

Unit – V: Advance Trees
Concept of threaded binary tree. Preorder and In-order traversals of in-order threaded binary tree, AVL Trees, OBST

Unit – VI: File organization
External storage devices, File, File types and file organization (sequential, index sequential and Direct access), Primitive operations and implementations for each type and comparison
Text Books:

Reference Books:
IT 2202 Computer Networks

Teaching Scheme:
Lectures: 3 Hrs/Week
Tutorial: 1 Hr/Week

In-Semester: 50 Marks
End-Semester: 50 Marks
Credits: 4

Course Objectives:
1. To understand routing and network layer.
2. Understanding of TCP and UDP key functions.
3. Understanding the role of transport layer in congestion control, fairness and stability of Internet.
4. To understand Wireless Technologies.

Course Outcomes:
1. Understand and implementing routing functions.
2. Recognize and analyse the usage of various protocols at network layer
3. Recognize usage of various protocols at application layer
4. Acquaint with wireless transmission media and their standards

Unit – I: Internetworking

Unit – II: Introduction to Routing and Packet Forwarding
Inside the Router, CPU, NVRAM, Router Interfaces, Routers and the Network Layer, Command Line Interface Configuration and Addressing, Basic Router Configuration, Building the Routing Table, IP Routing, Path Determination and Switching Functions Protocols.

Unit – III: IP Routing

Unit – IV: Transport Layer
Transport layer duties and functionalities, application expectations and IP delivery semantics. UDP: UDP functionality, UDP Header.

Unit – V: Application Layer
Client/Server Model, Telnet, Domain Name System, File Transfer protocol: FTP, TFTP, Hyper Text Transfer Protocol, POP3, IMAP, SMTP, E-mail, MIME, Simple Network Management Protocol

Unit – VI: Wireless Technologies & SDN
Introduction to wireless internetwork, IEEE 802.11, Cellular Technology, WLAN, Internet of Things. Bring Your Own Device. Software defined networking, concept, architecture, applications.

Text Books:

Reference Books:

IT 2203 Computer Organization and Architecture

Teaching Scheme:
Lectures: 3 Hrs/Week
Tutorial: 1 Hr/Week

Examination Scheme:
In-Semester: 50 Marks
End-Semester: 50 Marks
Credits: 4

Course Objectives:
1. To understand configuration of Computer Systems
2. To understand fundamental working of Computer Systems.
3. To study architecture and features of 8086 microprocessor
4. To learn interfacing of 8086 microprocessor with I/O peripherals

Course Outcomes:
On completion of the course, student will be able to explain—
1. Structure and function of Computer System
2. Architectural details of 8086 microprocessor
3. Memory management and Interrupts of 8086
4. Interfacing of microprocessor with I/O peripherals

Unit – I: Basic Processing Unit and Machine Instructions (07)
Fundamental Concept of basic processing Unit: Register Transfer, Arithmetic Logic Operation, Fetching and storing a word, Execution of Complete Instruction. Instruction and Instruction Sequencing: Instruction Types, Straight line Sequencing, branching, Condition codes. Addressing Modes

Unit – II: Processing Unit 8086 Microprocessor: Architecture, (08)
Instruction Descriptions and Assembler Directives
Introduction to 8086: internal Architecture, generation of physical address, minimum/maximum mode, study of 8086 supporting chips 8288(Latch), 8284(Clock Generator), 8286(trans receiver), 8288(Bus controller). Timing diagram read Write machine cycle. Introduction to assembly language programming- Instruction Descriptions, Assembler Directives.

Unit – III: Assembly Language Programming and Interrupt structure (07)
Address translation, addressing modes, Examples of programming, Procedures and Macros Interrupt Structure, Interrupt service routine, Interrupt vector table, hardware and software interrupts, INTR, NMI, Interrupt response, Execution of ISR, Priorities of interrupt

Unit – IV: Interfacing with 8086-I (07)
8259(Programmable Interrupt Controller)- Block Diagram, control and status register, Interfacing and programming. 8255(Programmable peripheral interface)- Block diagram control word, Interfacing ADC and DAC.

Unit – V: Interfacing with 8086-II (06)
8253/54(programmable interval timer/counter)- Block Diagram, control word. Modes of timer 8251(USART)- Features, Block Diagram, Control and Status register, operating modes.

Unit – VI: Parallel Organization (05)
Parallel Organization – Multiprocessors, Multicores & Clusters. Flynn’s Taxonomy for Multiple Processor Organizations, Closely and Loosely Coupled Multiprocessors Systems, Symmetric Multiprocessor (SMP) Organization, Multi-threading – Fine Grained, Coarse

Text Books:


Reference Books:

IT 2204 Object Oriented Paradigms

Teaching Scheme:  Examination Scheme:
Lectures: 3 Hrs/Week In-Semester: 50 Marks
Tutorial: 1 Hr/Week End-Semester: 50 Marks
Credits: 4

Course Objectives:
1. The students should be able to understand abstraction
2. The students should be able to understand the encapsulation
3. The students should be able to understand the inheritance and polymorphism.
4. The students will be able to convert a small problem description in an object oriented code

Course Outcomes:
1. The students will be able to abstract required properties and behavior of a class from a given description
2. The students will be able to identify inheritance from the given description.
3. The students will be able to identify encapsulation and polymorphism from a given description
4. The students will be able to identify all the features of object oriented paradigm from the given description

Unit – I: Building blocks of Object Oriented Programming (06)
Revision of procedural programming, Limitations of procedural programming, Algorithmic decomposition Vs Object Oriented decomposition.
Concepts of Class, Object, State of an Object, behavior of an object and identity for an object.
Introduction to scope: private/ protected/ public/package level
Concepts of Information hiding, Abstraction and Encapsulation as what are those and their necessity.

Unit – II: Abstraction (06)
Writing a class with private instance variables and instance methods in appropriate scope, properties with accessor (getXXX) and modifier (setXXX) methods, and constructors.
Effective use of comments such as class level, method level, and inline
Class as a user defined data type against primitive data types. Instantiating an object, using it through its abstraction. Introduction to terms ‘Reference’

Unit – III: Inheritance and substitution (06)
Method overloading, overloaded constructors, chaining of constructors. ‘this’ keyword and its concept, division into parts, composition, layers of specialization, subclass, subtypes, forms of inheritance, variations on inheritance, benefits and cost of inheritance
Best practices: naming conventions, packaging (name space).
Methods from Object class: rules for overriding equals(), hashCode() and toString().

Unit – IV: Polymorphism and code reuse (06)
Containment: Code reuse through containment of objects. Object as a smallest reusable unit.
Distribution of responsibilities across application. Localization of impact due to changes in requirement.
Inheritance: Concept referring to generalization-specialization, inheritance for members according to the scope, code reuse, method overriding, polymorphism, effects of using base
class reference for child class object, chaining of constructors (passing data to super class).

**Unit – V: Abstract class and aggregation in Object orientation**

Abstract class, abstract methods, concept of Interface, final class/method
Array of ‘primitive data type’ and Array of ‘user defined data type’, introduction to multi
dimension array.

**Unit – VI: Introduction to I/O Programming and Exception**

Introduction to language specific Collections framework, introduction to concept of List/Set/
Map and techniques to iterate over them.

**Text Books:**

1. Kathy Sierra, ‘OCA / OCP Java SE 7 Programmer I & II Study Guide, Chapter 1, 2
   and 7 Oracle press (2014)

**Reference Books:**

1. Khalid A Mughal, ‘A programmer’s guide to Java SE 8 oracle certified associate’
   Oracle press (2017)
BSIT 2201 Engineering Mathematics III

Teaching Scheme:
Lectures: 3 Hrs/Week
Tutorial: 1 Hr/Week

Examination Scheme:
In-Semester: 50 Marks
End-Semester: 50 Marks
Credits: 4

Prerequisites:
1. Permutation and Combination
2. Complex numbers - Properties, Argand Diagram, Basic properties of integration.
3. Partial Fractions, Basic properties of integration, Beta and Gamma Functions.
4. Number System.
5. First order linear ordinary differential equations.

Course Objectives:
1. To recall and remember basics of Statistics, complex analysis, Fourier and Z-transforms, Number theory and Higher order LDE.
2. To understand the concepts of basic mathematical methods for solving Statistics, complex analysis, Fourier and Z-transforms, Number theory and Higher order LDE.
3. To apply these methods to solve engineering problems.
4. To analyze engineering problems and evaluate.

Course Outcomes:
On completion of the course, learner will be able to –
1. Remember terminologies and formulae in Statistics, complex analysis, Fourier and Z-transforms, Number theory and Higher order LDE.
2. Understand and interpret the concepts of Statistics, complex analysis, Fourier and Z-transforms, Number theory and Higher order LDE.
3. Apply the methods in Statistics, complex analysis, Fourier and Z-transforms, Number theory and Higher order LDE.
4. Compare and analyze the problems in Statistics, complex analysis, Fourier and Z-transforms, Number theory and Higher order LDE.

Unit – I: Statistics
Measures of central tendency, Standard deviation, Coefficient of Variation, Moments, Skewness & Kurtosis, Probability, Theorems on probability, Conditional Probability & Bayes’ theorem.

Unit – II: Probability Distributions

Unit – III: Complex Analysis
Functions of Complex variables, Analytic Functions, Cauchy Riemann-Equations, Cauchy’s Integral Theorem, Cauchy’s Integral Formula, Laurent’s series, Residue theorem, Conformal mapping, Bilinear Transformation

Unit – IV: Transforms
Z Transforms - Definition, standard properties, Z- Transform of standard sequences and their inverses, solution of difference equation.
Fourier Transforms - Complex exponential form of Fourier series, Fourier integral theorem, sine and cosine integrals, Fourier transform, Fourier Sine and Cosine transform and their inverses.

Unit – V: Number Theory
Modular Arithmetic, Greatest common divisor, Euclid's algorithm, Chinese remainder theorem, Fermat's theorem, Discrete Logarithm.

Unit – VI: Higher Order Linear Differential equation and application

Higher order Linear differential equation with constant coefficients, Method of Variation of parameters, Cauchy’s and Legendre’s DE, Simultaneous DE, Modelling of electrical circuits.

Text Books:

Reference Books:
IT 2205 – Data Structures II Laboratory

Teaching Scheme:
Practical: 4 Hrs/Week

Examination Scheme:
Practical: 50 Marks
Credits: 02

Prerequisites:
IT 2106: Data Structures I Laboratory

Course Objectives:
1. To use linear data structures – stack & queue.
2. To learn non-linear data structures and their applications.
3. To learn different file organizations
4. To learn different hashing techniques
5. To understand use of data structures using OOP language

Course Outcomes:
1. Students will be able to apply appropriate data structures for real world problems.
2. Students will be to use dynamic memory allocation concepts and file handling in various application developments.
3. Students will be able to perform basic analysis of algorithms with respect to time and space complexity.
4. Students will be able to use data structure using OOP environment.
5. Students will be able to use algorithmic foundations for solving problems and programming.

Suggested List of Laboratory Assignments (11 assignments)

Group A Assignments (C Programming)
1. Implement stack as an abstract data type using linked list and use this ADT for conversion of infix expression to postfix, prefix and evaluation of postfix expression.
2. Construct an expression tree from postfix/prefix expression and perform recursive and non-recursive In-order, pre-order and post-order traversals.
4. Represent any real world graph using adjacency list /adjacency matrix find minimum spanning tree using Kruskal’s algorithm.
5. Represent a given graph using adjacency matrix /adjacency list and find the shortest path using Dijkstra’s algorithm (single source all destination).
6. Implement direct access file using hashing (chaining without replacement) perform following operations on it a) Create Database b) Display Database c) Add a record d) Search a record e) Modify a record.

Group B: (Using Python programming) (Any2)
1. Implement priority queue as ADT using single linked list for servicing patients in an hospital with priorities as a) Serious (top priority) b) medium illness (medium priority) c) General (Least priority).
2. Create Binary tree and perform following operations: a) Insert b) Display c) Depth of a tree d) Display leaf-nodes e) Create a copy of a tree.
3. Consider a friends’ network on face book social web site. Model it as a graph to represent each node as a user and a link to represent the friend relationship between them. Store data such as date of birth, number of comments for each user. a) Find who is having maximum friends b) Find who has post maximum and minimum
comments c) Find users having birthday in this month. Hint: (Use adjacency list representation and perform DFS and BFS traversals)

4. Store data of students with telephone no and name in the structure using hashing function for telephone number and implement chaining with and without replacement.

5. A business house has several offices in different countries; they want to lease phone lines to connect them with each other and the phone company charges different rent to connect different pairs of cities. Business house want to connect all its offices with a minimum total cost. Solve the problem by suggesting appropriate data structures

**Group C Assignments (C++ / Java) (Any 2)**
1. Expression conversion using STL
2. Expression conversion using linked list
3. Binary Tree operations
4. Huffman coding
5. Sequential file handling

**Group D Assignment (Any Programming Language) (Any 1)**
One assignment to be carried out by a group of 2 students.
Each group will design an application using the appropriate data structures – to solve real world problem (proof of concept). The group requires to get the application approved by the respective faculty member.
1. Implementation of Tower of Hanoi (Non recursive implementation)
2. Recursive solution to problems (e.g. Tower of Hanoi)
3. Text editor (Hint – GLL)
4. Implementation of Process scheduling (e.g. long-term, short-term scheduler)
5. Implementation of AVL trees
6. Implementation of Loss less compression technique (Huffman) – encode & decode
7. Threaded binary tree – thread creation, display
8. Implementation of Hierarchical structure of organization (e.g. no. of first classes etc)
9. Simulation of college network
10. Searching & counting no. of occurrence & location (line no) of a word in a given text file
11. Formation of Magic square
12. Implementation to generalize Time / Train based aptitude question (R. S. Agarwal)
13. Develop games (e.g. Tic-tac-toe, sudoku)
14. Code beautifier (e.g. int a ; - keyword shown in blue color, others in black)

**Text Books:**


**Reference Books:**


IT 2206 Network Laboratory
Teaching Scheme:
Practical: 2Hrs/Week

Examination Scheme:
Practical: 25 Marks
Credits: 1

Course Objectives:
1. To understand Routing and its Concepts.
2. To acquaint students with IP routing.
3. To understand dynamic Routing Protocols.
4. To understand Wireless Technologies.

Course Outcomes:
1. Understand routing and its configuration
2. Recognize usage of various protocols at network layer
3. Recognize usage of various protocols at application layer
4. Acquaint with wireless transmission media and their standards

Suggested List of Laboratory Assignments
1. Configuration of Local Area Network.
2. Configuration of Static Routes on Router.
4. Implementation of Virtual LAN.
7. Configuration of FTP, TELNET and DHCP.
8. Configuration of wireless network.

Text Books:

Reference Books:
IT 2207 Computer Organization and Architecture Laboratory

Teaching Scheme: Practical: 2 Hrs/Week

Examination Scheme: Practical: 25 Marks
Credits: 1

Course Objectives:
1. To understand configuration of Computer Systems
2. To understand fundamental working of Computer Systems.
3. To study architecture and features of 8086 microprocessor
4. To learn interfacing of 8086 microprocessor with I/O peripherals

Course Outcomes:
On completion of the course, student will be able to –
1. Write assembly language programs to perform numeric operations
2. Write assembly language programs to perform string operations
3. Interface various I/O peripherals with microprocessor
4. Understand the internal architecture of modern processors

Suggested List of Laboratory Assignments
1. Write Assembly Language Program (ALP) for
   a) Addition and subtraction of 8 bit numbers. OR
   b) Program to count negative numbers from signed numbers either stored in memory
      or given by user. OR
   c) Ascending/descending sort
2. Write ALP to convert 4-digit Hex number into its equivalent BCD number and 4-digit BCD number into its equivalent HEX number.
3. Write ALP to perform following operation on string:
   a) Find and display length
   b) Display reverse
   c) Check whether string is palindrome or not.
   d) Concatenation of two strings
   e) Find number of words
   Display proper strings to prompt the user while accepting the input and displaying the result.
4. Write ALP to interface 8255 (PPI) with 8086
5. Write ALP to interface 8251 (Serial Interface) with 8086
6. Write ALP to interface 8254/8253(Timer/Counter) with 8086
7. Write ALP to interface 8259 (Programmable interrupt Controller) with 8086
8. Study Assignment: Explain architecture of Quad core Processor in detail with an application

Text Books:

Reference Books:
1. Intel Manual

IT 2208 Object Oriented Programming Laboratory
Teaching Scheme:
Practical: 2 Hrs/Week

Examination Scheme:
End-Semester: 50 Marks
Credits: 1

Course Objectives:
1. The students should be able to understand abstraction
2. The students should be able to understand the encapsulation
3. The students should be able to understand the inheritance and polymorphism.
4. The students will be able to convert a small problem description in an object oriented code

Course Outcomes:
1. The students will be able to abstract required properties and behavior of a class from a given description
2. The students will be able to identify inheritance from the given description.
3. The students will be able to identify encapsulation and polymorphism from a given description
4. The students will be able to identify all the features of object oriented paradigm from the given description

List of assignments
1. Convert the given description into an object oriented language code. An employee has an employeeID, name. Display the data for five employees

2. Convert the given description into an object oriented language code. An employee has an employeeID, name. Every employee has a basic pay and a joining date. Display the data for five employees

3. Convert the given description into an object oriented language code. An employee has an employeeID, name and salutation. Every employee has a basic pay and a joining date. Display the data for five employees

4. Convert the given description into an object oriented language code. An employee has an employeeID, name, salutation and an address. The address has apartment number, apartment name, road, city, state and pincode. Every employee has a basic pay and a joining date. Display the data for five employees

5. Convert the given description into an object oriented language code. An employee has an employeeID, name, salutation and an address. The address has apartment number, apartment name, road, city, state and pincode. Each employee gets a monthly salary. A team leader is an employee and a software engineer is an employee. The team leader gets the monthly salary as basic pay, DA as 70 percent of basic pay and traveling allowance as 15 percent of basic pay. The software engineer gets a monthly salary as basic pay, DA as 35 percent of basic pay and traveling allowance as 12 percent of basic pay.

6. Convert the given description into an object oriented language code. An employee has an employeeID, name, salutation and an address. The address has an apartment number, apartment name, road, city, state and pincode. Each employee gets a monthly salary. A team leader is an employee and a software engineer is an
employee. The team leader gets the monthly salary as basic pay, DA as 70 percent of basic pay and traveling allowance as 15 percent of basic pay. The software engineer gets a monthly salary as basic pay, Da as 35 percent of basic pay and traveling allowance as 12 percent of basic pay. Now, it is policy of the company that every software engineer will get an add on compensation if she works for more than 8 hours in a day. The compensation is calculated as Rs 200.00 per hour. If a team lead works for more than 8 hours in a day, she gets an add on compensation as Rs 600.00 for a slab of 4 hours. Incorporate this in the code.

7. Convert the following description in an object oriented language using abstraction, has a relationship, is a relationship, encapsulation and data hiding

A bank issues many credit cards. Each credit card has a credit card no. It has a list of purchases associated with it. Every purchase made using the credit card has date of purchase, amount of purchase and pay back points for that purchase. The credit card has the total payback points accumulated across all the purchases made. The policy for adding the payback points for every purchase is as follows

<table>
<thead>
<tr>
<th>Sr No</th>
<th>Date of purchase</th>
<th>Quarter</th>
<th>Pay back points</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1&lt;sup&gt;st&lt;/sup&gt; Jan to 31&lt;sup&gt;st&lt;/sup&gt; Mar</td>
<td>First</td>
<td>1 payback point for every 200 Rs purchase</td>
</tr>
<tr>
<td>2</td>
<td>1&lt;sup&gt;st&lt;/sup&gt; April to 30&lt;sup&gt;th&lt;/sup&gt; June</td>
<td>Second</td>
<td>1 payback point for every 150 Rs purchase</td>
</tr>
<tr>
<td>3</td>
<td>1&lt;sup&gt;st&lt;/sup&gt; July to 30&lt;sup&gt;th&lt;/sup&gt; Sept</td>
<td>Third</td>
<td>1 payback point for every 100 Rs purchase</td>
</tr>
<tr>
<td>4</td>
<td>1&lt;sup&gt;st&lt;/sup&gt; Oct to 31&lt;sup&gt;st&lt;/sup&gt; Dec</td>
<td>Fourth</td>
<td>1 payback point for every 80 Rs purchase</td>
</tr>
</tbody>
</table>

Calculate the total payback points for the following details
Credit Card = 123456789000

<table>
<thead>
<tr>
<th>Date of purchase</th>
<th>Purchase amount</th>
</tr>
</thead>
<tbody>
<tr>
<td>29&lt;sup&gt;th&lt;/sup&gt; March</td>
<td>20000.00</td>
</tr>
<tr>
<td>10&lt;sup&gt;th&lt;/sup&gt; July</td>
<td>30000.00</td>
</tr>
<tr>
<td>15&lt;sup&gt;th&lt;/sup&gt; Oct</td>
<td>15000.00</td>
</tr>
<tr>
<td>24&lt;sup&gt;th&lt;/sup&gt; Dec</td>
<td>10000.00</td>
</tr>
</tbody>
</table>

8. Convert the following description in an object oriented language using abstraction, has a relationship, is a relationship, encapsulation and data hiding
An account has an accountNo, balance and an account holder. An account holder has a name and an address. Address has apartment number, apartment name, road, city, state and pincode. An amount can be withdrawn from an account, deposited to an account or transferred from one account to other account. A saving account is an account. A current account is an account. A saving account gets an interest from the bank with an annual interest rate of 3.5 percent. This interest gets added to the balance amount. A current account is charged with a commission by the bank. The commission is charged annually with a rate of 2.5 percent. This commission gets deducted from the balance of the current account. Create one saving accounts with two deposits and one withdrawal. Create second saving accounts with one deposit and two withdrawals. Create third saving accounts with one deposit, one withdrawal and a transfer to first account. Create fourth account as current account with one deposit, three withdrawals and commission for two years.

Text Books:

Reference Books: