Autonomous Programme Structure of Second Year B. Tech. Computer Engineering Academic Year 2017-2018

S. Y. B. Tech. Computer Engineering Semester – I

Course Code	Course Title	Teaching Scheme Hours /Week			Examination Scheme				Marks	Credit
		Lecture	Tutorial	Practical	In Semester	End ester	Oral	Practical		
CE 2101	Principles of Programming Languages	3	0	0	50	50	0	0	100	3
CE 2102	Data Structures and Algorithms I	3	1	0	50	50	0	0	100	4
CE 2103	Discrete Mathematics	3	1	0	50	50	0	0	100	4
CE 2104	Digital Systems and Computer Organization	3	1	0	50	50	0	0	100	4
CE 2105	Principles of Programming Languages Laboratory	0	0	4	25	0	25	0	50	2
CE 2106	Data Structures and Algorithms I Laboratory	0	0	4	25	0	0	25	50	2
HS 2101	Principles of Economics and Finance	3	0	0	50	50	0	0	100	3
AC 2101	Self Expression	0	0	2	0	0	0	0	0	No Credit
	Total	15	3	10	300	250	25	25	600	22
	Grand Total	28			600				600	22

AC 2101 -- Audit Course : Self Expression

1. Dance
2. Drawing / Painting / Sketching
3. English Communication Skill
4. Film Appreciation
5. Origami
6. Theater

[illegible]

DEAN ACADEMICS
MKSSS's Cummins College of Engineering for Women Karvenagar, Pune-411052

[illegible]

Principal
MKSSS's Cummins College of Engg. For Women, Karvenagar, Pune-52

MKSSS's Cummins College of Engineering For Women ★ Karve Nagar Pune-411052.

[illegible]

APPROVED BY
Governing Body Members
MKSSS's Cummins College of Engineering for Women
Karvenagar, Pune-411052

CE 2101 PRINCIPLES OF PROGRAMMING LANGUAGES

Teaching Scheme	**Examination Scheme**
Lecture : 3 Hrs/week	In semester : 50 marks
	End semester : 50 marks
	Credits: 3

Prerequisite:
ES 1202 Fundamentals of Programming Languages - II

Course Objectives:
To facilitate the learners :
1) To understand and apply object-oriented principles for application development.
2) To develop programming applications using Java.
3) To understand design concepts for programming languages
4) To analyse various programming paradigms.

Course Outcome:
By taking this course, the learner will be able to :
1) Make use of object-oriented principles for effective programming.
2) Construct simple programs using object-oriented programming language Java.
3) Explore languages design concepts for programming languages
4) Classify different programming paradigms for application development.

Unit 1: INTRODUCTION (06)

Role of programming languages, need to study programming languages, characteristics of a good programming languages, introduction to various programming paradigms. Need of object-oriented paradigm, basic concepts of object oriented programming (OOP), benefits of OOP. General characteristics for OOP, concepts - object, classes, messages, methods. Class Identification, object-oriented as abstract data type. Data abstraction, encapsulation, polymorphism, inheritance, dynamic binding, abstract classes, interfaces, generic class, run time type identification

Unit 2: OBJECT-ORIENTED PROGRAMMING WITH JAVA (08)

Java history, Java features, Java and Internet, Java virtual machine, class, object, methods, constructors, this keyword. Garbage collection, finalize method, argument passing, function overloading, constructor overloading. Access Control, static, final, Arrays, inheritance, base class and derived class, protected members, constructor in derived class. Concept of polymorphism, abstract classes, overriding member functions, super keyword

Unit 3: INTERFACES, EXCEPTION HANDLING AND COLLECTIONS (08)

Interfaces, package, exception fundamentals, try, catch, throw, throws, finally, built-in exceptions, custom exceptions. Java collection framework overview, collection interfaces, collection classes : ArrayList, accessing collection via iterator. Basic input output in Java, Basics of AWT and Swing.

Unit 4: LANGUAGE DESIGN CONCEPTS (06)

Programming language design, programming language processing. Data types: primitive data types, composite data types, recursive data types, implementation and storage representation of data types. Type binding, binding and binding times, type checking, type conversion, expressions, statements

Unit 5: PROCEDURAL PROGRAMMING (06)

Introduction to procedures, parameter passing methods, lifetime of variables, scope rules: static and dynamic scope, nested scope, procedure call and return, recursive sub-program. Referencing environment, activation records, storage management , desirable and undesirable characteristics of procedural programming

Unit 6:FUNCTIONAL PROGRAMMING (08)

Introduction to functional programming, lambda calculus, ambiguity, free and bound identifiers, reductions, typed lambda calculus, application of functional programming. Functional Programming with python, elements of functional Programming, function declaration, expression evaluation, type checking

Text Books:

1. Roosta S., "**Foundations of Programming Languages**", Thomson, Brooke/Cole, (India Edition) (2009)
2. Herbert Schilt, “**JAVA Complete Reference**”, Tata McGraw Hill, (9th Edition), (2014)
3. David Mertz, “ **Functional Programming in Python** ”, O’Reilly , (1st Edition), (2015)
4. Sethi R., "**Programming Languages concepts & constructs"**, Pearson Education, (2nd Edition) (2007)

Reference Books:

1. Sebesta R., "**Concepts Of Programming Languages**", Pearson Education, (10th Edition)(2014)
2. Eckel B., "**Thinking in Java**", Pearson Education, (3rd Edition)
3. T. W. Pratt , "**Programming Languages**", Prentice-Hall Of India, (4th Edition),(2009)
4. Summerfield M, “ **Programming In Python 3: A Complete Introduction To The Python Language”,** Pearson Education. (2nd Edition) (2011).
5. Lutz M, “ **Programming Python**”, SPD/O'reilly, (4th Edition),(2015).
6. Allen Tucker, Robert Noonan, “**Programming Languages: Principles and Paradigms**”, Tata McGraw Hill, (2nd edition),(2007)
7. Carlo Ghezzi, Mehdi Jazayeri, “**Programming Language Concepts**”,3rd Ed, Wiley Publication ISBN : 978-81-265-1861-6.

CE 2102 DATA STRUCTURES AND ALGORITHMS I

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

Prerequisite:

1. ES 1202 Fundamentals of Programming Language - II

Course Objectives:
To facilitate the learners:

1. To recall and understand the concepts of problem solving, algorithms and data structures.
2. To understand data representation, implementation and applications of linear data structures.
3. To learn, apply and analyze various data searching and sorting techniques.
4. To analyze algorithms using time and space complexity.

Course Outcomes:
By taking this course, the learner will be able to:

1. Apply appropriate linear data structure to construct efficient algorithms to approach the given problem .
2. Apply the concept of Linked list to solve given problem.
3. Distinguish between various linear data structures based on their representations and applications.
4. Solve examples using data searching and sorting techniques.
5. Analyse algorithms using time and space complexity.

Unit 1: Introduction to Algorithm, Data Structures and Analysis of Algorithms (07)
Concept of Problem Solving, Introduction to Algorithms, Characteristics of Algorithms, Pseudo code and Flowchart , Abstract Data Types (ADT), Set as an ADT. Introduction to Data Structures, Classification of Data Structures. Frequency Count, Analyzing Algorithm using Frequency count, Time complexity and Space complexity of an Algorithm, Asymptotic notations, Best, Worst and Average case analysis of an Algorithm.

Unit 2: Linear Data Structures Using Sequential Organization (06)
Concept of Sequential Organization, Concept of Linear Data Structures, Array as an ADT, Storage Representation of an Array – Row major and Column major, Introduction to Multidimensional Arrays. Concept of Ordered List , Application: Polynomial as an ADT using Array. Introduction to Strings and operations on Strings. Sparse Matrices

Unit 3: Linked List (08)
Concept of Linked List, Comparison of Sequential and Linked Organizations, Linked List using Dynamic Memory Management, Linked List as an ADT, Introduction to types of Linked List, Linked List operations. Time complexity analysis of Linked List operations. Application: Polynomial as ADT using Linked List.

Unit 4: Stacks (07)
Stack as an ADT, Representation and Implementation of Stack using Sequential and Linked Organization. Applications of Stack- Simulating Recursion using Stack, Arithmetic

Expression Conversion and Evaluation, Reversing a String. Time complexity analysis of Stack operations.

Unit 5: Queues **(06)**

Queue as an ADT, Representation and Implementation of Linear Queue, Circular Queue, Priority Queue, Double Ended Queue. Applications: Job scheduling, Queue simulation, Categorizing data. Time complexity analysis of Queue operations. Comparison of Linear Data Structures.

Unit 6: Sorting and Searching Techniques **(08)**

Need of Sorting and Searching, Sorting Order and Stability in Sorting. Concept of Internal and External Sorting. Bubble Sort, Insertion Sort, Selection Sort, Quick Sort and Merge Sort, Radix Sort, Shell Sort. Time complexity analysis of Sorting Algorithms. Linear Search, Binary Search, Time complexity analysis of Searching Algorithms.

Text Books:

1. E. Horwitz , S. Sahani, D. Mehta, **"Fundamentals of Data Structures in C++"**, *University Press*, (2nd edition) (2008).
2. R. Gilberg, B. Forouzan, **"Data Structures: A Pseudocode approach with C++"**, *Brooks* (1st Edition) (2001).

References:

1. Yedidyah Langsam, Moshe J Augenstein, Aron M Tenenbaum, **"Data Structures using C and C++"** , *Pearson Education*, (2nd edition) (2009).
2. A. Aho, J. Hopcroft, J. Ulman, **"Data Structures and Algorithms"**, *Pearson Education*, (2nd edition) (2008) .
3. Brassard and Bratley, **"Fundamentals of Algorithmics"**, *Prentice Hall India/Pearson Education*, (2nd edition) (2009).
4. Goodrich, Tamassia, Goldwasser, **"Data Structures and Algorithms in C++"**, *Wiley publication*, (2nd edition) (2011).
5. R. Gillberg, B. Forouzn, **"Data Structures: A Pseudocode approach with C"**, *Cenage Learning*, (2nd edition) (2003).
6. M. Weiss, **"Data Structures and Algorithm Analysis in C++"**, *Pearson Education*, (4th edition) (2002).

List of the Tutorial Assignments:

Every student should perform 12 to 14 tutorials which will cover topics of all units mentioned in the syllabus of Data Structures and Algorithms I. Tutorial assignments will enhance the understanding of the concepts of problem solving, algorithms and data structures. Students will perform practice exercise on data representation and corresponding implementation of the data structures. Students will get opportunity to develop their logic building abilities.

Following list of tutorials can be considered as guideline for designing tutorials:

1. Demonstration of C++ program implementation and execution using eclipse tool.
2. Design an algorithm for simple problems like GCD calculation, power calculation etc.
3. Calculate frequency count, time complexity of sample algorithmic constructs.
4. For given algorithms of array operation, write equivalent C++ code.
5. Practice exercise on sorting algorithms for set of predefined inputs.
6. Calculate time complexity of sorting algorithms using concept of frequency count.
7. Practice exercise on searching algorithms for set of predefined inputs.
8. Run through code of searching algorithms.
9. Create a linked list and write algorithms for traversal, delete a node, add a node operations on a list.
10. Create a doubly or circular linked list and write algorithms for traversal, delete a node, add a node operations on a list.

11. Solve brain teaser based on recursive code snippets.
12. Demonstration on debugging techniques.
13. Select appropriate data structures and design algorithmic solution to given application.
14. Solve puzzles based on queue data structure.

CE 2103 DISCRETE MATHEMATICS

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

Course Objectives:
To facilitate the learners

1. To understand Discrete Mathematics concepts and its significance in Computer Engineering.
2. To understand set theory, logic and apply reasoning to solve problems.
3. To solve problems based on algebraic systems, functions and relations.
4. To learn the basic properties of graphs, trees and permutation, combination to find solutions of related applications.

Course Outcomes:
By taking this course, the learner will be able to

1. Apply concepts of propositional calculus for solving problems, formal proofs, reasoning and represent problems using concepts of predicate calculus.
2. Solve problems on sets, relations and functions.
3. Solve problems on groups, permutations and combinations.
4. Apply basic terminologies of graphs and trees to solve problems on paper.

Unit 1: Sets and Mathematical Induction (07)
Significance of Discrete Mathematics in Computer Engineering, Sets, Subset, Universal Set, Empty Set, Algebra of Sets and Duality, Operations on Sets, Finite and Infinite Sets, Un Countably Infinite Sets, Multi-Sets, Power Set, Venn Diagram, Principle of Inclusion and Exclusion, Principle of Mathematical Induction.

Unit 2: Logic and Propositional Calculus (06)
Propositions, Logical connectives, Conditionals and Bi-Conditionals, Tautology, Contradiction, Contingency, Truth Tables, Logical Equivalences, Algebra of Propositions, Logical Implications, Conjunctive and disjunctive Normal Forms, Rules of Inference, Predicates and Quantifiers, Nested Quantifiers.

Unit 3: Groups, Rings and Permutations and Combinations (08)
Algebraic Systems, Groups, Semi Groups, Monoids, Subgroups, Introduction to Isomorphism, Homomorphism and Automorphism of groups, Cosets and Normal Subgroups, Rings, Integral Domain and Field, Introduction to Permutations and Combinations.

Unit 4: Relations and Functions (08)
Introduction to Relations, Product Sets, Pictorial Representation of Relations, Composition of Relations, Closure of Relations, Properties of Binary Relations, Equivalence Relations and Partitions, Partial Ordering Relations, Hasse Diagram, Lattices, Chains and Anti-Chains, Warshall's Algorithm, Functions, Composition of Functions, Invertible Functions, Introduction to Discrete Numeric Functions, Generating Functions and Recurrence Relation.

Unit 5: Graph Theory (07)
Basic Terminology, Multi-Graphs and Weighted Graphs, Sub-Graphs, Isomorphic Graphs, Complete, Regular and Bipartite Graphs, Operations on Graph, Factors of a Graph, Paths and Circuits, Connectivity, Hamiltonian and Euler Paths and Circuits, Shortest Path in Weighted Graphs (Dijkstra's Algorithm), Planer Graph and Theorem, Graph Coloring Problem, Travelling Salesman Problem.

Unit 6: Trees **(06)**
Basic Terminologies in Trees and Properties of Trees, Binary Search Trees, Tree Traversal, Spanning Trees, Fundamental Circuits and Cut Sets, Minimal Spanning Trees, Kruskal's and Prim's Algorithms for Minimal Spanning Trees.

Text Books:

1. Kenneth H. Rosen, **"Discrete Mathematics and its Applications",** 7th Edition, 2012, *Tata McGraw-Hill*, ISBN 978-0-07-338309-5.
2. C. L. Liu and D. P. Mohapatra, **"Elements of Discrete Mathematics"**, 4th Edition, *Tata McGraw-Hill*, 2017, ISBN 978-1-25-900639-5.

References:

1. Norman L. Biggs, **"Discrete Mathematics"**, Second Edition, *Oxford University Press*, 2004, ISBN 0–19 –850717 – 8.
2. J. P.Tremblay and R.Manohar, "**Discrete Mathematical Structures with Applications to Computer Science"**, 1997, *Tata McGraw-Hill*, ISBN 0-07- 463113-6.
3. E. Goodaire and M. Parmenter, **"Discrete Mathematics with Graph Theory"**, third edition, *Pearson Education*, 2008, ISBN 81 – 7808 – 827 – 4.
4. B. Kolman, R. Busby and S. Ross, **"Discrete Mathematical Structures"**, 6th Edition, *Pearson Education*, 2009, ISBN 81-7808-556-9.
5. N. Deo, **"Graph Theory with application to Engineering and Computer Science"**, Eastern Economy Edition, *Prentice Hall of India*, 1990, 0 – 87692 – 145 – 4.
6. Seymour Lipsehutz and Marc Lars Lipson **"Discrete Mathematics"**, 3rd Special Indian Edition, ISBN-13: 978-0-07-060174-1.

List of the Tutorial Assignments:

Every student should perform 12-14 tutorials which will cover topics of all units mentioned in the Syllabus of Discrete Mathematics.

Following list of tutorials can be considered as a guideline for designing tutorials in such a way that all topics should be distributed and covered amongst all batches.

1. Problems on set, multi-set operations,Venn diagram. and algebra of sets.
2. Problems on Principle of Inclusion-Exclusion and Mathematical Induction.
3. Translating English statement into propositional logic and predicate logic.
4. Problems on groups.
5. Problems on permutation and combination.
6. Representation of relations and functions, closure of relations and equivalence relation.
7. Problems on partitions, posets, Hasse diagram and Lattices.
8. Problems on Warshall's Algorithm.
9. Problems on composition of functions, invertible functions, recurrence relation.
10. Problems on multi-graphs and weighted graphs, sub-graphs, isomorphic graphs.
11. Solve problems for shortest path in weighted graphs (Dijkstra's algorithm) : (Paper pencil method)
12. Problems on minimal spanning trees, Kruskal's and Prim's algorithms for minimal spanning trees.

CE 2104 DIGITAL SYSTEMS AND COMPUTER ORGANIZATION

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

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

Prerequisite:

1. Basic Electrical and Electronics Engineering II (ES1201)

Forward Course Linkage(s):

1. Microprocessor Architectures (CE 2204)
2. Program Specific Elective – Embedded systems (PECE 3101)
3. Open Elective - High Performance Computing (OE 4101)

Course Objectives:

To facilitate the learners

1. To understand the basic digital circuits and logic design.
2. To apply techniques for designing combinational and sequential circuits.
3. To understand the functional components of a computer and its organization.
4. To understand design issues of instructions and instruction pipelining.
5. To understand and classify memory and input/output organizations.

Course Outcomes:

By taking this course, the learner will be able to

1. Make use of the knowledge of basic digital circuit elements and model the logic circuits.
2. Build simple combinational and sequential digital circuits.
3. Utilize the basic building blocks and their coordination in computer organization.
4. Classify the instruction pipeline organizational issues.
5. Interpret the memory and I/O organization concepts.

Unit – 1: COMBINATIONAL CIRCUITS (08)

Minimization of Product of Sum(POS) and Sum of Product(SOP) functions and realization using logic gates, Introduction to Numbers and Codes, BCD, Gray, Excess-3 and their applications, Code conversion, Integer and floating point number representation, Signed and unsigned numbers, arithmetic operations, Introduction to basic Arithmetic Logical Unit(ALU) and Floating Point Unit(FPU).

Unit – 2: COMBINATIONAL LOGIC DESIGN (06)

Realization of basic combinational functions like comparison, decoding, multiplexing, demultiplexing, Design of Half Adder and Full Adder, Design of Half Subtractor and Full subtractor, BCD Adder, Look ahead and carry generator, Introduction to Carry Propagation Adder, Carry Save Adder.

Unit – 3: SEQUENTIAL CIRCUITS DESIGN (07)

Flip flops (FFs) and their excitation tables, FF conversions, Shift registers, Applications of FFs, Asynchronous and Synchronous counters, Sequence generators and detectors using

Moore and Mealy, Introduction to Algorithmic State Machines (ASM) charts, notations, design of a simple controller using ASM.

Unit – 4: INTRODUCTION TO COMPUTER ORGANIZATION (07)

Introduction to Computer Organization, Function and structure of a computer, Functional components and their Interconnection, Register organization, Number and size of registers, General purpose registers, Design and Organizational issues of registers, Control Unit organization, Hardwired vs. microprogrammed organization.

Unit- 5: CHARACTERISTICS, FUNCTIONS AND PIPELINING OF INSTRUCTIONS (07)

Instruction cycle, type of instructions, types of operands, Instruction set design, machine instructions characteristics, design issues of instructions, Instruction pipelining, performance and hazards of pipelining, RISC, CISC.

Unit – 6: MEMORY AND INPUT/OUTPUT ORGANIZATION (07)

Memory devices and organization, ROM, RAM, EPROM, Flash memory. Cache memory organization, principles, cache design elements, performance characteristics, External memory devices and organization, hard disk, RAID, Introduction to buses, types of buses, bus organization, DMA organization, need, working principle.

Text Books:

1. R. P. Jain, '**Modern Digital Electronics**', *Tata McGraw-Hill,* (3rd Edition), (2003)
2. C. Hamacher, Z. Vranesic and S. Zaky, '**Computer Organization'**, *McGrawHill*, (2002)
3. W. Stallings, '**Computer Organization and Architecture - Designing for Performance'**, *Prentice Hall of India*,(8th edition), (2002)

References:

1. Anil Maini, '**Digital Electronics: Principles and Integrated Circuits'**, *Wiley India Ltd*, (2008)
2. Malvino, D. Leach, **'Digital Principles and Applications'**, *Tata Mc-Graw Hill*, (5th edition)
3. Stephan Brown, Zvonko Vranesic, **'Fundamental of Digital Logic with VHDL Design'**, Mc-Graw Hill, (2016)
4. John P Hays, **'Computer Architecture and Organization'**, *McGraw-Hill Publication*, (3rd Edition), (2001)
5. Tanenbaum, **'Structured Computer Organization'**, *Pearson,* (5th Edition)

Web References:

1. NPTEL series – nptel.ac.in/courses/117105080/ (Digital System Design by Prof. D. Roy Choudhary, Dept. of Computer Science and Engineering, IIT Kh.)
2. Online Chapters – WilliamStallings.com/COA/COA8e.html

List of Tutorial Assignments:

The subject Digital Systems and Computer Organization is a blend of two divergent subjects like Digital Electronics and Computer Organization. The list of tutorial topics tries to give demonstrations of circuit realizations on digital boards along with numerical problem solving. It also gives the demonstration of computer components and peripherals. The tutorial also aims to develop the research aptitude, soft skills and self-learning by different group activities, paper presentations, etc.

1. Solve SOP and POS examples using Boolean algebra and K-maps.
2. Demonstrate half adder and full adder using digital board.
3. Problem solving for number system and code conversions.
4. Problems on BCD operation to understand integer arithmetic.
5. Problem solving of Boolean expression using multiplexer/ demultiplexer.
6. Group activity on current trends / methods/ software tools used in digital electronics.
7. Design of flip flop conversion.
8. Problem solving on sequence detector using Moore and Mealy.
9. Demonstration of computer components and their interconnections.
10. Identify and explain the addressing modes of x 86 families.
11. & 12. Group-wise Presentations on recent trends in Microprocessor architectures by students.

CE 2105 PRINCIPLES OF PROGRAMMING LANGUAGES LABORATORY

Teaching Scheme
Lecture : 4 Hrs/week

Examination Scheme
In semester : 25 marks
Oral : 25 marks
Credits : 2

Course Objectives:
To facilitate the learners :
1) To explore the principles of object oriented programming
2) To apply object oriented programming concept for developing applications using Java
3) To apply Java collection framework for simple application development
4) To handle built-in and user defined exceptions
5) To explore functional language programming in python using simple examples

Course Outcome:
By taking this course, the learner will be able to :
1) Develop programming application using object oriented programming language Java
2) Make use of Java collection framework for effective programming.
3) Handle exceptions using inbuilt classes and user defined exceptions
4) Implement functional programming language concepts in python.

A large part of CE 2105 lab would be in understanding the syntax or semantics of languages which fall under various paradigms like Object Oriented (Java), and Functional and Scripting (Python). Main focus would be on Java programming whereas Python assignments are of introductory level as an example of programming paradigm. Assignment statements are in brief. Faculty members are encourage to expand problem statements with variations. Assignments can be framed and expanded in such a way that it explores concepts, language constructs, logic of solution and simple application.

List of assignments:

Group A: (Mandatory)
1. Design a user defined abstract data type ‘Complex' in Java. Write a program to perform arithmetic operations of two complex numbers
2. Implement the following concepts by constructing suitable classes in Java - a. Constructors b. Constructor Overloading c. Function Overloading d. Function Overriding e. Inheritance
3. Implement the following concepts by constructing suitable classes in Java - a. Abstract classes and abstract methods b. Interfaces
4. Create an application for a book shop and maintain the inventory of books that are being sold at the shop.
5. Write a Python program to count the number of articles in a given text.

Group B: (Any Four)
1. Create User defined exception to check the specific conditions for recruitment system and throw the exception if the criterion does not met in Java.
2. Create a student result database in Java. Calculate the grades of students. Decide a criteria for best student and short-list students who satisfies the criteria.
3. Find appropriate class hierarchy in banking application and implement it.
4. Find suitable class hierarchy in the human resource department of an organization and implement it.
5. Write a JAVA program to perform String operations.
6. Write a JAVA program to create an abstract data types like Stack/Set/Queue/List as an interface and implement its methods.
7. Write a Python program for sorting students marks.

Group C: (Any one)

1. Write a Python program that prompts a user to enter a list of words and store in another list only those words whose first letter occurs again within the word (e.g. Baboon). The program should display resulting list
2. Write a program in Python using functional paradigm for generating two sub-lists of even and odd numbers from given list. Perform addition of individual sub-list and display the result

CE 2106 DATA STRUCTURES AND ALGORITHMS I LABORATORY

Teaching Scheme
Practical : 4 Hrs/Week

Examination Scheme
In Semester : 25 Marks
Practical : 25 Marks
Credits : 2

Prerequisite:

1. ES 1202 Fundamentals of Programming Language - II
2. ES 1206 Fundamentals of Programming Language Laboratory - II

Course Objectives:
To facilitate the learners:

1. To develop algorithmic foundations to solve problems.
2. To select and use appropriate linear data structure for a given problem statement.
3. To analyze algorithms using time complexity.
4. To implement sorting and searching algorithms.

Course Outcome:
By taking this course, the learner will be able to:

1. Select linear data structures for given problem.
2. Develop the solution for the given problem using programming language.
3. Analyze solutions using time complexity.
4. Design a small application using linear data structure.

List of Assignments
The laboratory assignments are designed in a set of group A, B and C such that students will be able to design and implement solution for a given problem. Group A assignments are designed in such a way that students will choose appropriate data structures to implement solution of a given problem. All the units of the syllabus of Data Structures and Algorithms II are covered in group B assignments. Some assignments of group B are designed to make students able to implement Abstract Data Type of a data structure and use it for a given application. In group C assignments students will design an algorithmic solution for selected problem using concepts covered in the subject Data Structures and Algorithms II.
The laboratory assignments of group A and B are to be submitted by student individually using C++/JAVA object oriented programming language. Group C assignments may be performed in a group of 2 to 4 students from the same batch. For each assignment program code with sample output is to be submitted as a soft copy. Handwritten write up (Title, Objectives, Problem Statement, Algorithms, and Outcomes) of each assignment is to be submitted by students.

Group A: (Mandatory)

1. Shopkeeper keep a record for different items purchased by customers on a day. Select appropriate data structure and write a program to perform various operations on given information.
2. Design a system to maintain CSI student branch membership information. Choose appropriate data structure.
3. College Library maintains records of books. Write a program to implement sorting, searching operations on it. Use appropriate data structure.
4. Implement Queue as ADT using linked list or array. Use Queue ADT to simulate 'waiting list' operations of railway reservation system.

Group B: (At least six)

1. Implement permutation and combination based on word problem.

2. In a group of M persons, some people can speak English and some people can speak French. Write program to find union, intersection, difference of given sets.
3. Write a program to represent polynomial equation and perform operations to add and evaluate polynomials.
4. Write a program to perform add, multiply, transpose operations on matrices.
5. Write program to perform various operations on strings.
6. A mobile phone list stores name and contact number in ascending order. Write program to search a contact details of specified name.
7. Write a program to store first year CGPA of students. Use various sorting algorithms to sort data.
8. Implement Doubly Linked List as ADT .Use same ADT to simulate Browser URL application.
9. Implement Singly Linked List as ADT. Use same ADT to simulate deck of cards application.
10. A 'concordance List' is an alphabetical list of words that appear in the book . Implement concordance list using ordered Linked List with insertion function that restrict duplicate value to be inserted in the list.
11. Implement Singly Linked List as ADT. Use it to simulate banking operations.
12. Student's information along with their percentage is stored in linked list for every division. Generate a combine list of students which is sorted in descending order based on their percentage.

13. Implement Stack as ADT using linked list or array. Use same ADT to check given expression is well formed parenthesized.
14. Implement Stack as ADT using linked list or array. Use same ADT to evaluate given postfix expression.
15. Implement Priority Queue as ADT using linked list or array. Use ADT to simulate pizza parlor order management.
16. Operating system stores N jobs and processing time require to complete each job in data structure. Design a program to simulate the job execution sequence.

Group C:

Design a game OR Design a small application to manage library data / medical shop data/ College admission data / P.M.P.M.L. bus scheduling data etc. using appropriate data structures.