Diploma Engineering

Laboratory Manual

(Data Structures and Algorithms)
(4330704)

[Computer Engineering Semester - III]

Enrolment No	
Name	
Branch	
Academic Term	
Institute	

Directorate Of Technical Education
Gandhinagar – Gujarat

Data Structures and Algorithms

4330704

Lab manual is prepared by

Shri Uresh. N. Parmar

Lecturer – Diploma in Computer Engineering

C.U. Shah Government Polytechnic, Surendranagar

Smt. Rashmika K. Vaghela

Lecturer – Diploma in Computer Engineering

Government Polytechnic, Ahmedabad

Branch Coordinator

Shri B. H. Kantevala

HOD- Diploma in Computer Engineering

Government Polytechnic, Ahmedabad

Committee Chairman

Shri R. D. Raghani

(HOD-EC)

Principal (I/C)

Government Polytechnic, Gandhinagar

DTE's Vision

- To provide globally competitive technical education.
- To provide globally competitive technical education;
- Remove geographical imbalances and inconsistencies;
- Develop student friendly resources with a special focus on girls' education and support to weaker sections;
- Develop programs relevant to industry and create a vibrant pool of technical professionals.

Institute's Vision

Institute's Mission

Department's Vision

Department's Mission

Certificate

This is to certify that Mr./Ms ……..

Enrollment No. …… of 3rd Semester of *Diploma in Computer Engineering* of …….. (GTU Code) has satisfactorily completed the term work in course ***Data Structures and Algorithms*** for the academic year: …….. Term: Odd prescribed in the GTU curriculum.

Place: ……..

Date: ……..

Signature of Course Faculty **Head of the Department**

Preface

The primary aim of any laboratory/Practical/field work is enhancement of required skills as well as creative ability amongst students to solve real time problems by developing relevant competencies in psychomotor domain. Keeping in view, GTU has designed competency focused outcome-based curriculum -2021 (COGC-2021) for Diploma engineering programmes. In this more time is allotted to practical work than theory. It shows importance of enhancement of skills amongst students and it pays attention to utilize every second of time allotted for practical amongst Students, Instructors and Lecturers to achieve relevant outcomes by performing rather than writing practice in study type. It is essential for effective implementation of competency focused outcome- based Green curriculum-2021. Every practical has been keenly designed to serve as a tool to develop & enhance relevant industry needed competency in each and every student. These psychomotor skills are very difficult to develop through traditional chalk and board content delivery method in the classroom. Accordingly, this lab manual has been designed to focus on the industry defined relevant outcomes, rather than old practice of conducting practical to prove concept and theory.

By using this lab manual, students can read procedure one day in advance to actual performance day of practical experiment which generates interest and also, they can have idea of judgement of magnitude prior to performance. This in turn enhances predetermined outcomes amongst students. Each and every Experiment /Practical in this manual begins by competency, industry relevant skills, course outcomes as well as practical outcomes which serve as a key role for doing the practical. The students will also have a clear idea of safety and necessary precautions to be taken while performing experiment.

This manual also provides guidelines to lecturers to facilitate student-cantered lab activities for each practical/experiment by arranging and managing necessary resources in order that the students follow the procedures with required safety and necessary precautions to achieve outcomes. It also gives an idea that how students will be assessed by providing Rubrics.

Data structures and algorithms are fundamental concepts in computer science and

play a crucial role in solving complex problems efficiently. This course provides basic understanding of concepts of data structures, arrays and strings, stack and queues, linked list, trees, sorting and hashing methods. Learning data structures is crucial for efficient data organization, algorithm design, problem-solving, software performance optimization, and career advancement. It provides a solid foundation for further learning and understanding of advanced concepts in computer science.

Although we try our level best to design this lab manual, but always there are chances of improvement. We welcome any suggestions for improvement.

Programme Outcomes (POs) to be achieved through practical of this course.

1. **Basic and discipline specific knowledge:**
   Apply knowledge of basic mathematics, science and engineering fundamentals and engineering specialization to solve the engineering problems.
2. **Problem analysis:**
   Identify and analyse well-defined engineering problems using codified standard methods.
3. **Design/ development of solutions:**
   Design solutions for engineering well-defined technical problems and assist with the design of systems components or processes to meet specified needs.
4. **Engineering Tools, Experimentation and Testing:**
   Apply modern engineering tools and appropriate technique to conduct standard tests and measurements.
5. **Engineering practices for society, sustainability and environment:**
   Apply appropriate technology in context of society, sustainability, environment and ethical practices.
6. **Project Management:**
   Use engineering management principles individually, as a team member or a leader to manage projects and effectively communicate about well-defined engineering activities.
7. **Life-long learning:**
   Ability to analyze individual needs and engage in updating in the context of technological changes in field of engineering.

Practical Outcome – Course Outcome matrix

Course Outcomes (COs):

- **CO (a):** Perform various operations on arrays and strings.
- **CO (b):** Demonstrate algorithms to insert and delete elements from the stack and queue data structure.
- **CO (c):** Apply basic operations on the linked list data structure.
- **CO (d):** Illustrate algorithms to insert, delete and searching a node in tree.
- **CO (e):** Apply different sorting and searching algorithms to the small data sets.

Sr. No.	Practical Outcome/ Title of Experiment	CO (a)	CO (b)	CO (c)	CO (d)	CO (e)
1	Define various terms such as algorithm, various approaches to design an algorithm, time complexity, space complexity, best case, average case and worst-case time complexity etc.	✓	–	–	–	–
2	Implement array using row major order and column major order.	✓	–	–	–	–
3	Implement Sequential search algorithms.	✓	–	–	–	–
4	Implement Binary search algorithms.	✓	–	–	–	–
5	Implement various string algorithms.	✓	–	–	–	–
6	Implement push and pop algorithms of stack using array.	–	✓	–	–	–
7	Implement recursive functions.	–	✓	–	–	–
8	Implement insert and delete algorithms of a queue using array	–	✓	–	–	–
9	Implement simple structure programs using pointers.	–	✓	–	–	–
10	Implement insertion of node in the beginning of the list in singly linked list.	–	–	✓	–	–
11	Implement insertion of node at the end of list in singly linked list.	–	–	✓	–	–
12	Implement insertion of node in sorted linked list.	–	–	✓	–	–
13	Implement insertion of node at any position in liked list.	–	–	✓	–	–
14	Implement counting no of node algorithm in singly linked list.	–	–	✓	–	–

15	Implement searching of a node algorithm in singly linked list.	–	–	✓	–	–
16	Implement delete a node algorithm in singly linked list.	–	–	✓	–	–
17	Implement construction of binary search tree. Implement in-order, preorder and post-order traversal methods in binary search tree.	–	–	✓	–	–
18	Implement searching algorithm in binary search tree.	–	–	–	✓	–
19	Implement Bubble sort algorithm.	–	–	–	✓	–
20	Implement Selection sort algorithm.	–	–	–	–	✓
21	Implement Quick Sort algorithm.	–	–	–	–	✓
22	Implement Insertion sort algorithm.	–	–	–	–	✓
23	Implement Merge Sort algorithm.	–	–	–	–	✓
24	Solve hash table example using division method, method square method, folding method. (paper work only)	–	–	–	–	✓

Industry relevant skills

The following industry relevant skills are expected to be developed in the students by performance of experiments of this course.

1. Gain the ability to think algorithmically, which involves breaking down problems into smaller, more manageable steps and developing efficient algorithms to solve them.
2. Organize data in efficient manner.
3. Able to solve real world problems using the concepts of data structures and algorithms.

Guidelines to Teachers

1. Teacher should provide the guideline with demonstration of practical to the students. Teacher is expected to refer complete curriculum document and follow guidelines for implementation strategies.
2. Teacher shall explain prior concepts and industrial relevance to the students before starting of each practical
3. Involve all students in performance of each experiment and should give opportunity to students for hands on experience.
4. Teacher should ensure that the respective skills and competencies are developed in the students after the completion of the practical exercise.
5. Teacher is expected to share the skills and competencies to be developed in the students.
6. Finally give practical quiz as per the instructions. Utilise 2 hrs of lab hours effectively and ensure completion of write up with quiz also on same day.

Instructions for Students

1. Organize the work in the group and make record of all observations.
2. Students shall develop maintenance skill as expected by industries.
3. Student shall develop the habits of evolving more ideas, innovations, skills etc.
4. Student shall refer technical magazines and data books.
5. Student should develop habit to submit the practical on date and time.
6. Student should well prepare while submitting write-up of exercise.

Continuous Assessment Sheet

Enrollment No. :- **Term :-**

Name :-

Sr. No.	Title of Experiment	Page	Date	Marks (25)	Sign
1	Define various terms such as algorithm, various approaches to design an algorithm, time complexity, space complexity, best case, average case and worst-case time complexity etc.	1			
2	Implement array using row major order and column major order.	9			
3	Implement Sequential search algorithms.	19			
4	Implement Binary search algorithms.	26			
5	Implement various string algorithms.	34			
6	Implement push and pop algorithms of stack using array.	45			
7	Implement recursive functions.	56			
8	Implement insert and delete algorithms of a queue using array.	65			
9	Implement simple structure programs using pointers.	75			
10	Implement insertion of node in the beginning of the list in singly linked list.	85			
11	Implement insertion of node at the end of list in singly linked list.	95			
12	Implement insertion of node in sorted linked list.	102			
13	Implement insertion of node at any position in liked list.	109			
14	Implement counting no of node algorithm in singly linked list.	116			
15	Implement searching of a node algorithm in singly linked list.	123			
16	Implement delete a node algorithm in	130			

	singly linked list.				
17	Implement construction of binary search tree. Implement in-order, preorder and post-order traversal methods in binary search tree.	140			
18	Implement searching algorithm in binary search tree.	157			
19	Implement Bubble sort algorithm.	164			
20	Implement Selection sort algorithm.	173			
21	Implement Quick Sort algorithm.	181			
22	Implement Insertion sort algorithm.	189			
23	Implement Merge Sort algorithm.	197			
24	Solve hash table example using division method, method square method, folding method. (paper work only)	205			

Date: _____

Practical No. 1: Define various terms such as algorithm, various approaches to design an algorithm, time complexity, space complexity, best case, average case and worst-case time complexity etc.

A Objective:

- The objective of learning algorithms is to extract meaningful patterns and knowledge from data in order to get the solution.
- To understand the complexity of an algorithm is used to analyze and understand the computational resources required, such as time and memory, to train and apply learning algorithms efficiently, aiming for faster and more scalable solutions without compromising performance.

B Expected Program Outcomes (POs):

1. **Basic and discipline specific knowledge:**
   Apply knowledge of basic mathematics, science and engineering fundamentals and engineering specialization to solve the engineering problems.
2. **Problem analysis:**
   Identify and analyse well-defined engineering problems using codified standard methods.
3. **Design/ development of solutions:**
   Design solutions for engineering well-defined technical problems and assist with the design of systems components or processes to meet specified needs.

C Expected Skills to be developed based on competency:

This practical is expected to develop the following skills for the industry-identified competency:

- ✓ Design algorithm for the given problem.
- ✓ Calculate various complexities of given algorithm.

D Expected Course Outcomes (COs):

Perform basic operations on arrays and strings.

E Practical Outcome (PRo):

Define various terms such as algorithm, time complexity, space complexity, best case, average case and worst-case time complexity etc.

F Expected Affective Domain Outcomes (ADOs):

1. Demonstrate working as a leader/a team member.
2. Maintain tools and equipment.
3. Follow ethical practices.

G. Prerequisite theory:

Digital computer is an electronics machine which is used for processing and manipulation of data. In order to make computer work we need to know,

- ✓ Representation of data in computer system
- ✓ Accessing of data from memory
- ✓ How to solve problem step by step

Data may be in form of text, image, audio, video etc. For data processing in computer system, we need to perform operations like store, access and manipulation of data.

Whenever we want to work with large amount of data, that data must be organized. To organize data, we can link different data together, group data with same characteristic, order data, etc.

Some examples of organized data are

- ✓ Dictionary
- ✓ Telephonic Directory
- ✓ City Map
- ✓ Data in Tabular format

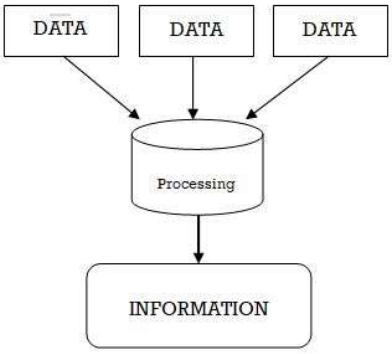
This makes accessing (searching) and manipulation of data very effective and faster. And for that, the data structure is highly needed.

We can say that "Data structure is a way of organizing data in a computer so that it can be used effectively."

Before understanding the data structure in detail, we need to learn various fundamentals first. Like: data, information, cell, field, record, list and file.

Data: data is a collection of facts (raw facts). It can be numbers, words, measurements, observations, figures or even just description of anything. Data is in unorganized form.

Information: it is processed, organised and structured data in a meaningful way. Information is more precise form of data. Information is created from data.



Cell: it is a basic structural and functional unit which represents an entity.

Field: The smallest piece of information that can be stored. OR simply *'single piece of information.'*

Record: *'it is a collection of fields'*, in a record, all the fields are logically related.

File: *'it is a collection of records'*; a file is a large list that stores the data & information and retrieves it. Files can be accessed in a sequential or random-access manner.

Various definitions of data structure.

- Data structure is a logical description or mathematical model of data organization.
- Data structure is a way of organizing data in a computer so that it can be used effectively.
- Data structure is a logical relationship between individual data elements related to the solution of given problem.

Types of data structures.

- ✓ Primitive data structure
- ✓ Non-primitive data structure
- ✓ Linear data structure
- ✓ Non-linear data structure
- ✓ Homogeneous data structure
- ✓ Non-homogeneous data structure
- ✓ Static
- ✓ Dynamic

H. Practical related Quiz

1. State TRUE (T) / FALSE (F) for the following.

I. Cell is the smallest peace of information that can be stored. ☐

II. An objective way to compare two algorithms is by comparing their execution time irrespective of the machines. ☐

III. Data is subset of information. ☐

IV. Pointer is a kind of primitive data structure. ☐

V. Collection of records is called file. ☐

2. Define algorithm.
3. List and explain key features of an algorithm.
4. Define time complexity.
5. Define space complexity.
6. Define worst case analysis, average case and best-case time complexity.
7. Write in brief about Big-Oh notation.

I. Safety and necessary precautions followed

- Shutdown computer system properly once the Lab hours are finished.
- Strictly follow the instructions given by laboratory in-charge.

J. References

I. https://www.tutorialspoint.com/Algorithms-and-Complexities
II. https://devopedia.org/algorithmic-complexity
III. https://www.javatpoint.com/daa-complexity-of-algorithm
IV. https://www.youtube.com/watch?v=vgSKOMsjLbc

K. Assessment Rubrics

Practical no. 1: Define various terms such as algorithm, various approaches to design an algorithm, time complexity, space complexity, best case, average case and worst-case time complexity etc.

Performance indicators for PROs	Weighted Marks	Max Marks	Rubrics	Obtained Marks
Correctness of program	**20%**	**5**	5 to 3 – Correct 2 – Partial correct 1 to 0 – Poor work	
Readability and documentation of the program/Quality of input and output displayed (messaging and formatting)	**20%**	**5**	5 to 4 – Excellent 3 to 2 – Good 1 to 0 – Poor	
Code efficiency	**10%**	**3**	3 – Excellent 2 – Good 1 to 0 – Poor	
Debugging ability	**20%**	**5**	5 to 4 – Excellent 3 to 2 – Good 1 to 0 – Poor	
Program execution/answer to questions	**30%**	**7**	7 to 5 – Proper execution 4 to 2 – Run with some errors 1 to 0 – Not run	
			Total Marks: (Out of 25)	

Signature with date

Date: _____

Practical No. 2: Implement array using row major order and column major order.

A Objective:

- To understand the different memory layouts and traversal patterns associated with these array representations.
- Enable efficient data access and manipulation.

B Expected Program Outcomes (POs):

1. **Basic and discipline specific knowledge:**
   Apply knowledge of basic mathematics, science and engineering fundamentals and engineering specialization to solve the engineering problems.
2. **Problem analysis:**
   Identify and analyse well-defined engineering problems using codified standard methods.
3. **Design/ development of solutions:**
   Design solutions for engineering well-defined technical problems and assist with the design of systems components or processes to meet specified needs.

C Expected Skills to be developed based on competency:

This practical is expected to develop the following skills for the industry-identified competency:

- ✓ Calculate address of array element.
- ✓ Write code for the given problem.
- ✓ Follow coding guidelines.

D Expected Course Outcomes (COs):

Perform basic operations on arrays and strings.

E Practical Outcome (PRo):

Implement array using row major order and column major order.

F Expected Affective Domain Outcomes (ADOs):

1. Demonstrate working as a leader/a team member.
2. Maintain tools and equipment.
3. Follow ethical practices.

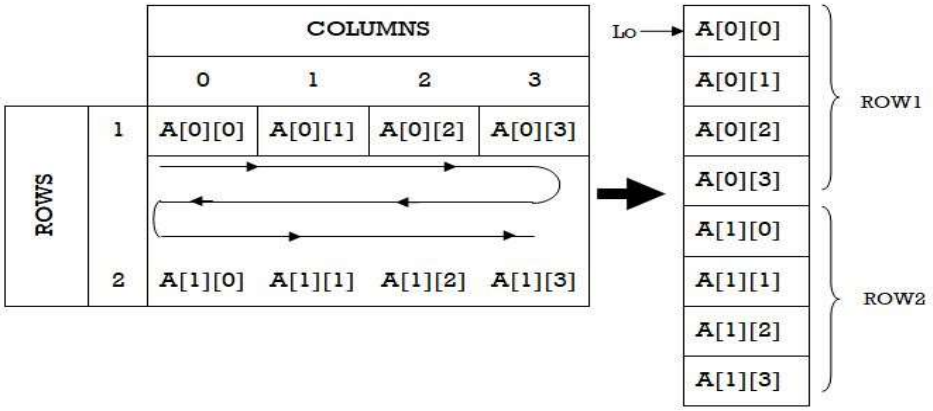
G. Prerequisite theory:

There are two main methods for representing multidimensional array in sequential memory.

1. Row major array
2. Column major array

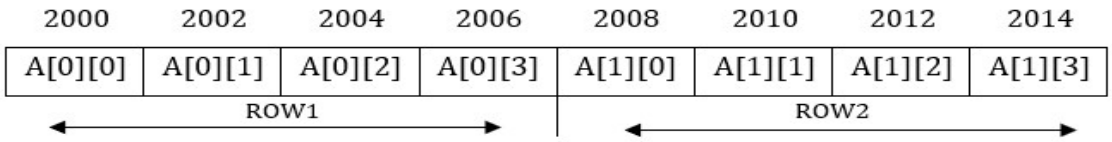
Row major arrays

In this representation → elements of array are arranged sequentially row by row, so the first row of the array occupies the first set of memory location reserved for the array, the second row occupies the next set and so on.



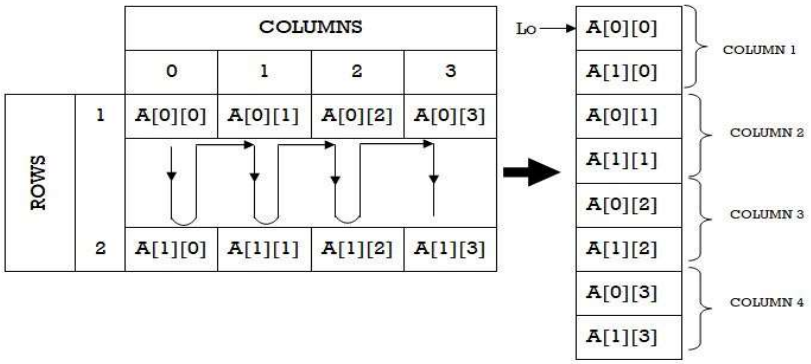
Here Lo is the base address (address of the first element)

If for example our base address is 2000, then memory allocation will be like this: (here we are considering integer data type of elements).



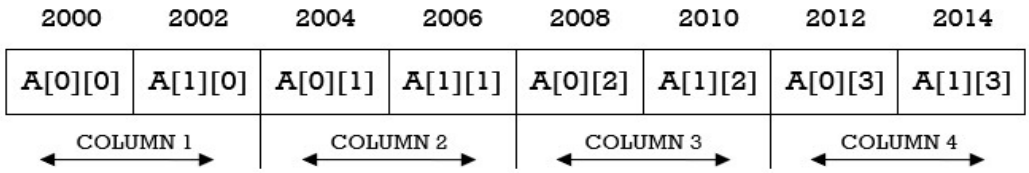
Column major arrays

In this representation → elements of array are arranged sequentially column by column, so the first column of the array occupies the first set of memory location reserved for the array, the second column occupies the next set and so on.



Here Lo is the base address (address of the first element).

If for example our base address is 2000, then memory allocation will be like this: *(here we are considering int data type of elements).*



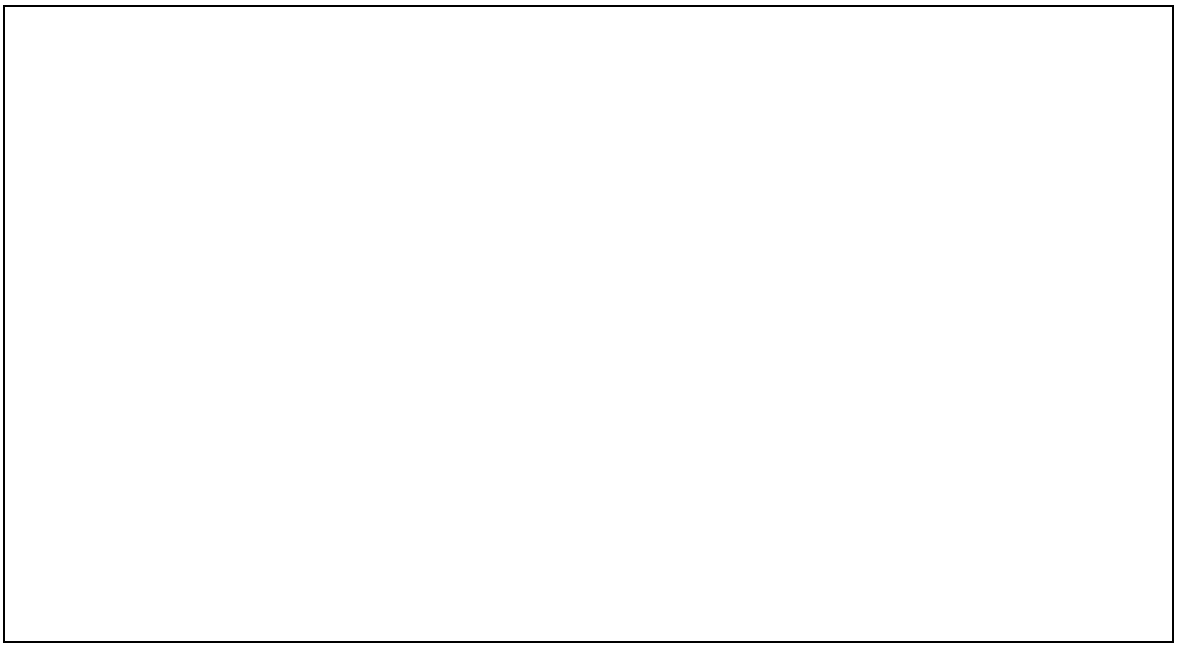
H. Resource / Equipment required

Sr. No.	Equipment / Software resources	Specifications
1	Computer system (Hardware Tool)	Computer having latest configuration with windows or Unix OS
2	C compiler (Software Tool)	Open-source software

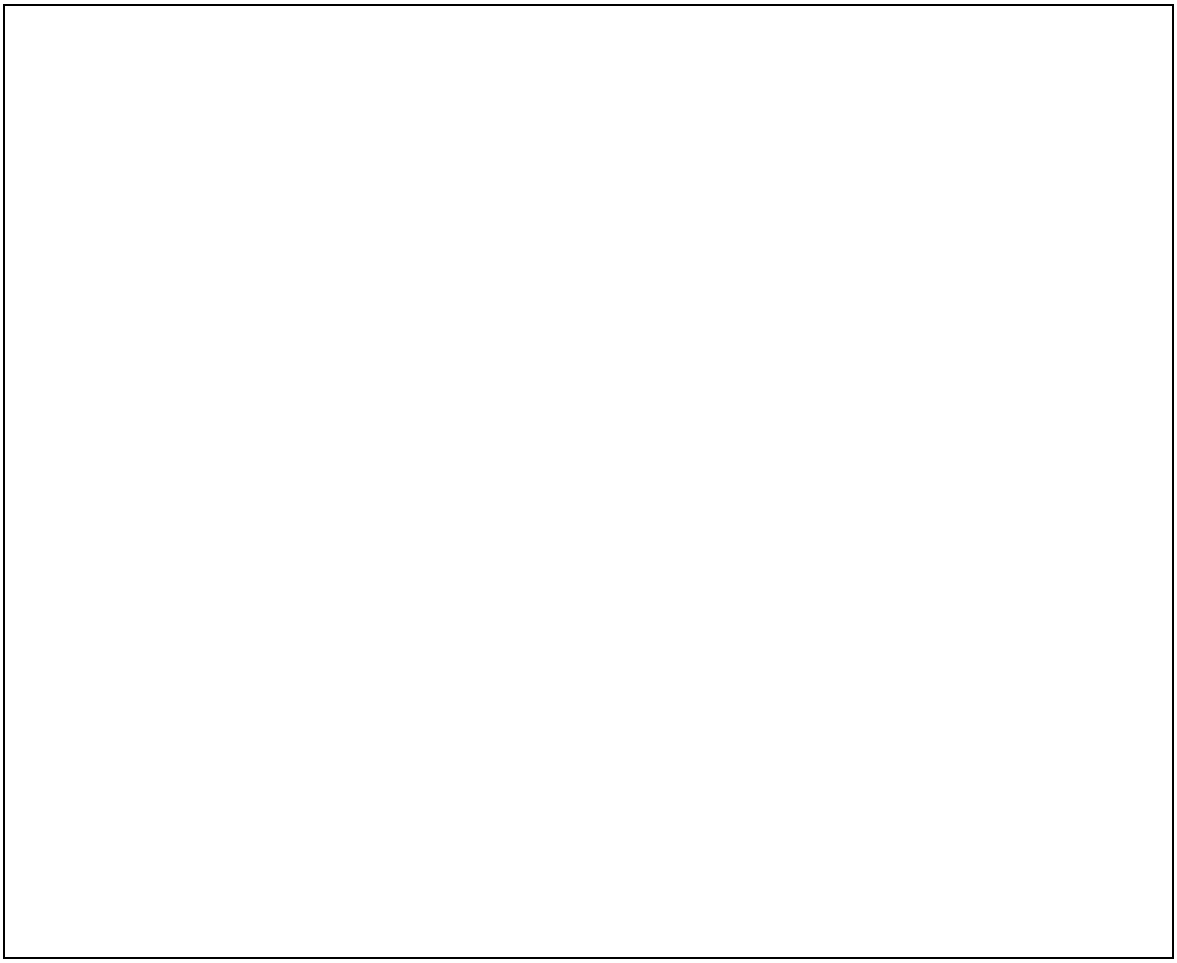
I. Safety and necessary precautions followed

- Shutdown computer system properly once the Lab hours are finished.
- Strictly follow the instructions given by laboratory in-charge.

J. Program Source Code (Implement row-major and column-major order using array).



K. Input – Output



L. Practical related Quiz

1. State TRUE (T) / FALSE (F) for the following.
   I. The memory address of the first element in a row-major array is the lowest among all elements. ☐
   II. In a two-dimensional row-major array, the second index represents the column number. ☐
   III. Row-major arrays have better cache locality compared to column-major arrays. ☐
2. Define array. Explain its characteristics.
3. Write and explain the syntax how to calculate address of particular array element in raw-major and column-major array.

M. References

I. https://www.geeksforgeeks.org/calculation-of-address-of-element-of-1-d-2-d-and-3-d-using-row-major-and-column-major-order/
II. https://iq.opengenus.org/row-major-and-column-major-order/
III. https://www.javatpoint.com/data-structure-2d-array
IV. https://www.youtube.com/watch?v=KDQXUysHLL8

N. Assessment Rubrics

Practical no. 2: Implement array using row major order and column major order.				
Performance indicators for PROs	**Weighted Marks**	**Max Marks**	**Rubrics**	**Obtained Marks**
Correctness of program	**20%**	**5**	5 to 3 – Correct 2 – Partial correct 1 to 0 – Poor work	
Readability and documentation of the program/Quality of input and output displayed (messaging and formatting)	**20%**	**5**	5 to 4 – Excellent 3 to 2 – Good 1 to 0 – Poor	
Code efficiency	**10%**	**3**	3 – Excellent 2 – Good 1 to 0 – Poor	
Debugging ability	**20%**	**5**	5 to 4 – Excellent 3 to 2 – Good 1 to 0 – Poor	
Program execution/answer to questions	**30%**	**7**	7 to 5 – Proper execution 4 to 2 – Run with some errors 1 to 0 – Not run	
			Total Marks: (Out of 25)	

Signature with date

Date: _____

Practical No. 3: Implement sequential search algorithm.

A Objective:

- To search the element from the given data.
- To locate the target element within an unordered list or array by sequentially examining each element, aiming to find a match and determine its presence or absence in the collection.

B Expected Program Outcomes (POs):

1. **Basic and discipline specific knowledge:**
   Apply knowledge of basic mathematics, science and engineering fundamentals and engineering specialization to solve the engineering problems.
2. **Problem analysis:**
   Identify and analyse well-defined engineering problems using codified standard methods.
3. **Design/ development of solutions:**
   Design solutions for engineering well-defined technical problems and assist with the design of systems components or processes to meet specified needs.

C Expected Skills to be developed based on competency:

This practical is expected to develop the following skills for the industry-identified competency:

- ✓ Design and implement search algorithm.
- ✓ Write and edit code for the given problem.
- ✓ Debug program to fix errors.

D Expected Course Outcomes (COs):

Perform basic operations on arrays and strings.

E Practical Outcome (PRo):

Implement sequential search algorithm.

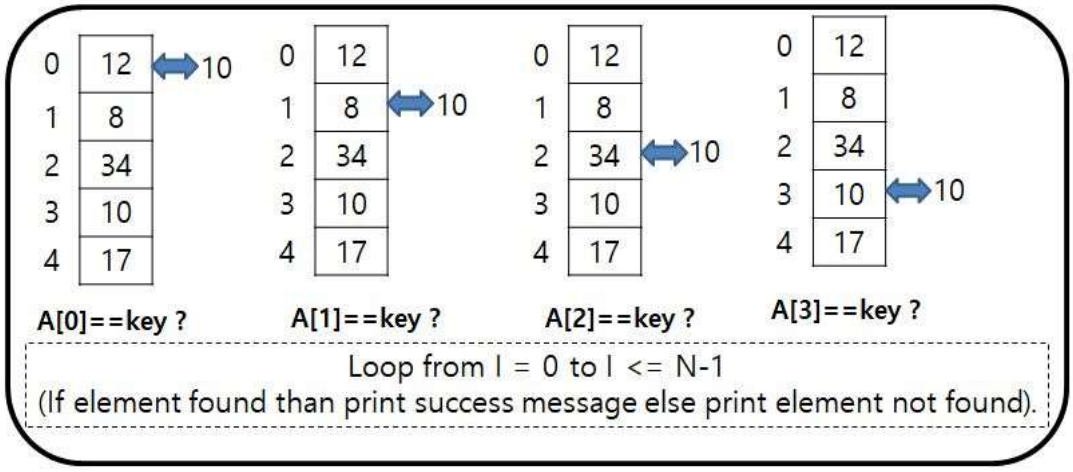
F Expected Affective Domain Outcomes (ADOs):

1. Demonstrate working as a leader/a team member.
2. Maintain tools and equipment.
3. Follow ethical practices.

G. Prerequisite theory:

In this method, searching of an element from an unordered given data or table. In it, traverse the whole table in sequential manner until the desired record is found. In this technique, first the value of the element (searching element) is compared with the value of first element in the given list.

If match is not found, then the next element in the list is compared and so on. This process is repeated until the searching element is found in the list. In worst case, for input N we need to compare KEY with all N element so here complexity of sequential search in O(N).



Algorithm: SEQ_SEARCH (A, N, KEY)

This algorithm finds an element KEY from the given array and returns 1 if search successful and 0 for unsuccessful.

- A is an array
- N represent number of elements in array
- KEY is the element to be searched
- I is temporary variable

```
Step1. [initialize search]
        i = 0
        A[N] = KEY              //put search element at next to last position
Step2. [search the array]
        repeat while (A[i] ≠ KEY)
```

```
                i = i + 1
Step3. [check for successful search]
        if (i = N)
        then    write ("Unsuccessful Search")
                return (0)
        else
                write ("Successful Search")
                return (1)
```

H. Resource / Equipment required

Sr. No.	Equipment / Software resources	Specifications
1	Computer system (Hardware Tool)	Computer having latest configuration with Windows or Unix OS
2	C compiler (Software Tool)	Open-source software

I. Safety and necessary precautions followed

- Shutdown computer system properly once the Lab hours are finished.
- Strictly follow the instructions given by laboratory in-charge.

J. Program Source Code (Implement sequential search algorithm).

K. Input - Output

L. Practical related Quiz

1. Define searching. List various searching methods.
2. Fill in the blanks for the following.
   I. Sequential search is also known as _____ search.
   II. The time complexity of sequential search in the worst case is _____.
3. State TRUE (T) / FALSE (F) for the following.
   I. Sequential search is generally faster than binary search for small-sized arrays. ☐
   II. Searching means arranging the elements in a proper order. ☐

M. References

I. https://www.tutorialspoint.com/data_structures_algorithms/linear_search_algorithm.htm
II. https://bradfieldcs.com/algos/searching/the-sequential-search/
III. https://chat.openai.com/
IV. https://www.youtube.com/watch?v=t9gErvSVid0

N. Assessment Rubrics

Practical no. 3: Implement sequential search algorithm.				
Performance indicators for PROs	**Weighted Marks**	**Max Marks**	**Rubrics**	**Obtained Marks**
Correctness of program	**20%**	**5**	5 to 3 – Correct 2 – Partial correct 1 to 0 – Poor work	
Readability and documentation of the program/Quality of input and output displayed (messaging and formatting)	**20%**	**5**	5 to 4 – Excellent 3 to 2 – Good 1 to 0 – Poor	
Code efficiency	**10%**	**3**	3 – Excellent 2 – Good 1 to 0 – Poor	
Debugging ability	**20%**	**5**	5 to 4 – Excellent 3 to 2 – Good 1 to 0 – Poor	
Program execution/answer to questions	**30%**	**7**	7 to 5 – Proper execution 4 to 2 – Run with some errors 1 to 0 – Not run	
			Total Marks: (Out of 25)	

Signature with date

Date: _____

Practical No. 4: Implement binary search algorithm.

A Objective:

- To search the element from the given data.
- locate the target element within a sorted list or array by repeatedly dividing the search space in half, reducing the number of elements to be examined at each step, aiming to quickly identify the position of the targeted element.

B Expected Program Outcomes (POs):

1. **Basic and discipline specific knowledge:**
   Apply knowledge of basic mathematics, science and engineering fundamentals and engineering specialization to solve the engineering problems.
2. **Problem analysis:**
   Identify and analyse well-defined engineering problems using codified standard methods.
3. **Design/ development of solutions:**
   Design solutions for engineering well-defined technical problems and assist with the design of systems components or processes to meet specified needs.

C Expected Skills to be developed based on competency:

This practical is expected to develop the following skills for the industry-identified competency:

- ✓ Design and implement search algorithm.
- ✓ Write and edit code for the given problem.
- ✓ Debug program to fix errors.

D Expected Course Outcomes (COs):

Perform basic operations on arrays and strings.

E Practical Outcome (PRo):

Implement binary search algorithm.

F Expected Affective Domain Outcomes (ADOs):

1. Demonstrate working as a leader/a team member.
2. Maintain tools and equipment.
3. Follow ethical practices.

G. Prerequisite theory:

This method is used to search an element in an ordered list (sorted data). It is very efficient method to search element. For this method, it's necessary to have the list in ordered (sorted) manner.

In binary Search each time we divide algorithm in two equal parts. Then compare this search element with Middle element. So here for any input N we need to compare only half element so complexity for binary search is O (log N).

Searching Logic:

- ✓ *If middle element is equal to search element than our search is completed. And we return that index.*
- ✓ *If search element is greater than middle than we have to search right part only.*
- ✓ *If Search element is smaller than middle than we have to search left part only.*

Algorithm: BINARY_SEARCH (A, N, X)

- A is an array
- N represent number of elements in array
- X is the element to be searched

```
Step1. [initialize]
        LOW = 1
        HIGH = N
Step2. [perform search]
        repeat thru step 4 while (LOW ≤ HIGH)
Step3. [obtain index of middle point value]
        MID =|(LOW + HIGH) / 2|
Step4. [compare]
        if (X < A[MID])
        then    HIGH = MID – 1
        else if  (X > A[MID])
        then    LOW = MID + 1
        else    write ("SUCCESSFUL SEARCH")
                return (MID)
step5. [unsuccessful search]
```

```
        write ("UNSUCCESSFUL SEARCH")
        return (0)
```

H. Resource / Equipment required

Sr. No.	Equipment / Software resources	Specifications
1	Computer system (Hardware Tool)	Computer having latest configuration with Windows or Unix OS
2	C compiler (Software Tool)	Open-source software

I. Safety and necessary precautions followed

- Shutdown computer system properly once the Lab hours are finished.
- Strictly follow the instructions given by laboratory in-charge.

J. Program Source Code (Implement binary search algorithm).

K. Input – Output

L. Practical related Quiz

1. Fill in the blanks for the following.
   I. Binary search requires the data to be _____.
   II. The time complexity of binary search in the worst case is_____.
   III. if the target value is less than the middle element, the algorithm continues searching in the _____ half of the current search space.
2. State TRUE (T) / FALSE (F) for the following.
   I. Binary search can be implemented recursively. ☐
   II. Binary search is not a linear search algorithm. ☐

III. Binary search requires the input array to be in ascending order. ☐

3. Provide tracing to search the element 15 from the given data using binary search method.

   5, 6, 10, 15, 20, 21, 24, 29, 32, 35

M. References

I. https://www.programiz.com/dsa/binary-search
II. https://www.tutorialspoint.com/data_structures_algorithms/binary_search_algorithm.htm
III. https://www.youtube.com/watch?v=sXABdGalFNg

N. Assessment Rubrics

Practical no. 4: Implement binary search algorithm.				
Performance indicators for PROs	**Weighted Marks**	**Max Marks**	**Rubrics**	**Obtained Marks**
Correctness of program	**20%**	**5**	5 to 3 – Correct 2 – Partial correct 1 to 0 – Poor work	
Readability and documentation of the program/Quality of input and output displayed (messaging and formatting)	**20%**	**5**	5 to 4 – Excellent 3 to 2 – Good 1 to 0 – Poor	
Code efficiency	**10%**	**3**	3 – Excellent 2 – Good 1 to 0 – Poor	
Debugging ability	**20%**	**5**	5 to 4 – Excellent 3 to 2 – Good 1 to 0 – Poor	
Program execution/answer to questions	**30%**	**7**	7 to 5 – Proper execution 4 to 2 – Run with some errors 1 to 0 – Not run	
			Total Marks: (Out of 25)	

Signature with date

Date: _____

Practical No. 5: Implement various string algorithms.

A Objective:

The objective of implementing string algorithms is to develop efficient and effective methods for manipulating and analyzing strings, enabling tasks such as searching for patterns, matching and comparing strings, extracting substrings, replacing or modifying text, and performing various text processing operations, with the goal of solving problems related to textual data efficiently.

B Expected Program Outcomes (POs):

1. **Basic and discipline specific knowledge:**
   Apply knowledge of basic mathematics, science and engineering fundamentals and engineering specialization to solve the engineering problems.
2. **Problem analysis:**
   Identify and analyse well-defined engineering problems using codified standard methods.
3. **Design/ development of solutions:**
   Design solutions for engineering well-defined technical problems and assist with the design of systems components or processes to meet specified needs.

C Expected Skills to be developed based on competency:

This practical is expected to develop the following skills for the industry-identified competency:

- ✓ Gain proficiency in string manipulation.
- ✓ Write code for the given problem.
- ✓ Debug program to fix errors.

D Expected Course Outcomes (COs):

Perform basic operations on character arrays.

E Practical Outcome (PRo):

Implement various string algorithms.

F Expected Affective Domain Outcomes (ADOs):

1. Demonstrate working as a leader/a team member.
2. Maintain tools and equipment.
3. Follow ethical practices.

G. Prerequisite theory:

A string is a sequence of collection of characters. String is one dimensional array of character and it is terminated by '\0' (NULL).

It is a group of zero or more characters between double quotation marks (" "). Like: "HELLO" is a string containing 5 characters.

A string should contain →Alphabets (lowercase a-z, uppercase A-Z), numeric values 0-9 or special characters +, -, NULL.

Algorithm: STR_LEN (str)

This algorithm counts the length of the given string.

- str is the given string

```
Step1. [initialize]
        read string "str1" from user
        count = 0
Step2. [process until end of the string]
        repeat while (str[count] ≠ NULL)
        count = count + 1
Step3. [finish]
```

Algorithm: TOLOWER (S1, S2)

This algorithm converts the entered string into lower case.

- S1 is the string that we have to convert.

```
Step1. [initialize]
        read string "s1" from user
        i = 0
Step2. [convert upper to lower case]
        repeat while(s1[i] ≠ NULL)
                if(s1[i] ≥ "A" AND s1[i] ≤ "Z")
                then s2[i] =
                        s1[i] +
                        32 i = i +
                        1
```

```
                    else
                                s2[i]
 = s1[i] Step3. [finish]
          S2[i] = NULL
return (s2)
Step1. [initialize]
          read string "str1" from user
          count = 0
Step2. [process until end of the string]
          repeat while (str[count] ≠ NULL)
          count = count + 1
Step3. [finish]
```

Algorithm: STR_CAT (S1, S2, S3)

This algorithm concatenates two strings S1 and S2 into the third string S3.

```
Step1. [initialize]
          read two strings "S1" and "S2" from user
          count1 = 0
          count2 = 0
          count3 = 0
Step2. [process until end of the first string]
          repeat while (S1[count1] ≠ NULL)
                    S3 [count3] = S1 [count1]
                    count1 = count1 + 1
                     count3 = count3 + 1
Step3. [process until end of second string]
          repeat while (S2[count2] ≠ NULL)
                     S3[count3] = S2[count2]
                     count3 = count3 + 1
                      count2 = count2 + 1
```

```
Step4. [finish]
        S3[count3] = NULL
return (S3)
```

H. Resource / Equipment required

Sr. No.	Equipment / Software resources	Specifications
1	Computer system (Hardware Tool)	Computer having latest configuration with Windows or Unix OS
2	C compiler (Software Tool)	Open-source software

I. Safety and necessary precautions followed

- Shutdown computer system properly once the Lab hours are finished.
- Strictly follow the instructions given by laboratory in-charge.

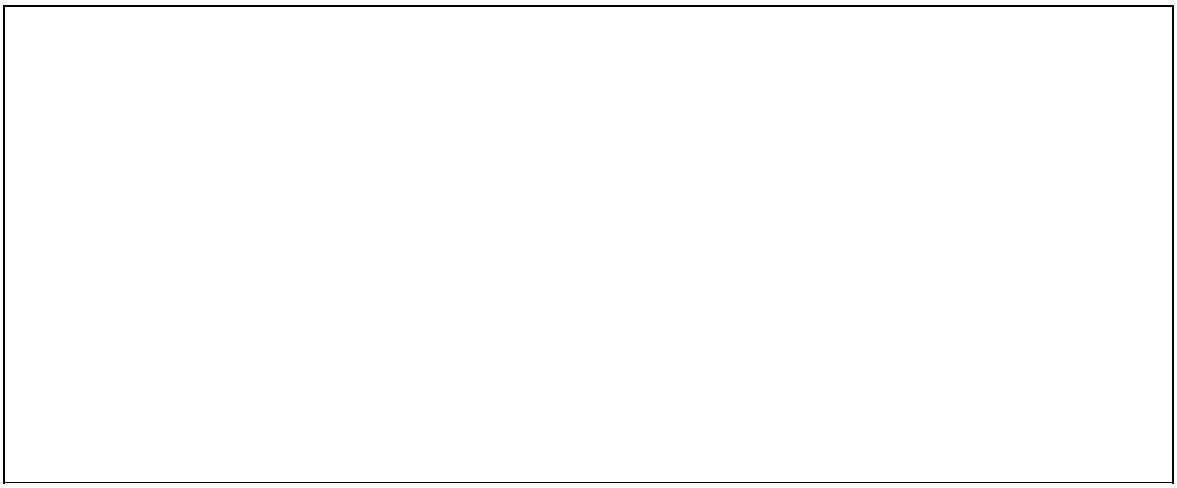
J. Program Source Code (Implement a program to find the length of given string).

I. Write a source code to find the length of a given string.

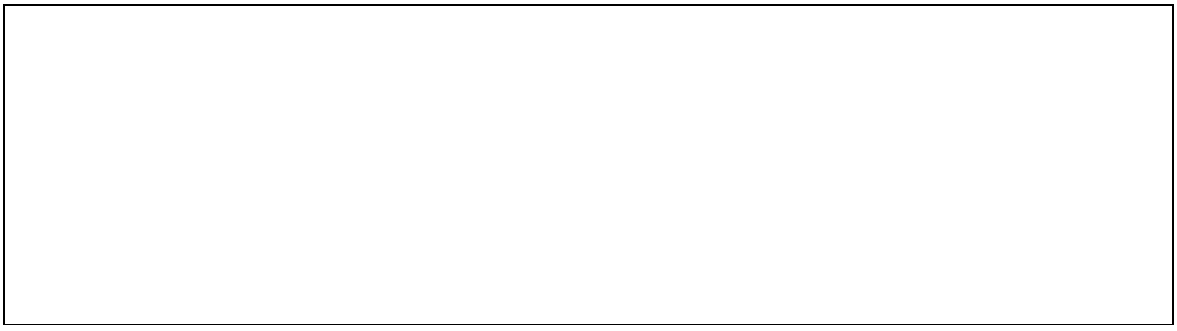
K (1). Input – Output

II. Write a source code to convert the given string into lower case.

K (2). Input – Output



III. Write a source code to concatenate two strings into third one.



K (3). Input – Output

L. Practical related Quiz

1. Define string. List various string operations.
2. Differentiate between getch( ) and getchar( ).
3. Fill in the blanks for the following.
   I. _____ function is used to get a single character from the terminal (output).
   II. The string inbuilt function _____ is used to convert the upper case in to lower one.
   III. The string inbuilt function _____ is used to copy the content of one string into another.
   IV. The string inbuilt function _____ is used to reverse the given string.
4. State TRUE (T) / FALSE (F) for the following.
   I. In C programming, a string is a built-in data type. ☐
   II. A string must always be terminated with a null character. ☐
   III. Strings in C programming are mutable, meaning they can be modified after they are created. ☐
   IV. String comparison should always be done using the "==" operator. ☐
   V. In C programming, the %s format specifier is used to print a string. ☐

M. References

I. https://www.javatpoint.com/c-string-functions
II. https://www.programiz.com/c-programming/string-handling-functions
III. https://chat.openai.com/
IV. https://www.scaler.com/topics/c/string-functions-in-c/
V. https://www.youtube.com/watch?v=4NGthO9F1Bo

N. Assessment Rubrics

Practical no. 5: Implement various string algorithms.				
Performance indicators for PROs	**Weighted Marks**	**Max Marks**	**Rubrics**	**Obtained Marks**
Correctness of program	**20%**	**5**	5 to 3 – Correct 2 – Partial correct 1 to 0 – Poor work	
Readability and documentation of the program/Quality of input and output displayed (messaging and formatting)	**20%**	**5**	5 to 4 – Excellent 3 to 2 – Good 1 to 0 – Poor	
Code efficiency	**10%**	**3**	3 – Excellent 2 – Good 1 to 0 – Poor	
Debugging ability	**20%**	**5**	5 to 4 – Excellent 3 to 2 – Good 1 to 0 – Poor	
Program execution/answer to questions	**30%**	**7**	7 to 5 – Proper execution 4 to 2 – Run with some errors 1 to 0 – Not run	
			Total Marks: (Out of 25)	

Signature with date

Date: _____

Practical No. 6: Implement PUSH and POP algorithms of stack using array

A Objective:

- To understand the fundamental principles and functionalities of stack data structures.
- To acquire knowledge and skills to perform essential stack operations such as PUSH and POP.

B Expected Program Outcomes (POs):

1. **Basic and discipline specific knowledge:**
   Apply knowledge of basic mathematics, science and engineering fundamentals and engineering specialization to solve the engineering problems.
2. **Problem analysis:**
   Identify and analyse well-defined engineering problems using codified standard methods.
3. **Design/ development of solutions:**
   Design solutions for engineering well-defined technical problems and assist with the design of systems components or processes to meet specified needs.

C Expected Skills to be developed based on competency:

This practical is expected to develop the following skills for the industry-identified competency:

- ✓ Write code for the given problem.
- ✓ Debug programs to fix errors.
- ✓ Write algorithm to implement stack operations.

D Expected Course Outcomes (COs):

Perform Demonstrate algorithms to insert and delete elements from the stack and queue data structure.

E Practical Outcome (PRo):

Implement PUSH and POP algorithms of stack using array.

F Expected Affective Domain Outcomes (ADOs):

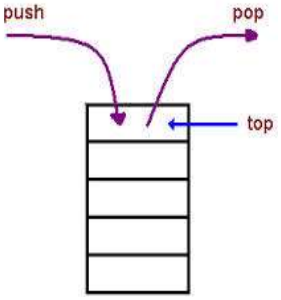
1. Demonstrate working as a leader/a team member.
2. Maintain tools and equipment.
3. Follow ethical practices.

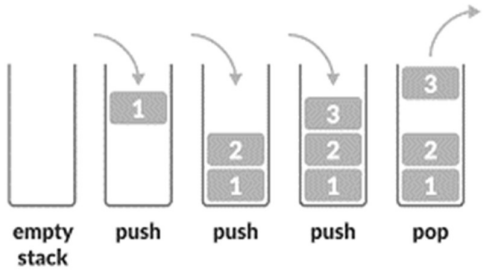
G. Prerequisite theory:

A stack is a non-primitive linear data structure in which insertion and deletion operations are performed at only one end, known as top of the stack (TOP).

As deletion and insertion operation in stack is done from one end, the last added element will be removed first. So stack is also called Last In First Out (LIFO).

Each time a new element is inserted in the stack the TOP pointer is increment by one and the pointer is decrement by one when element is deleted.





Operations of Stack (PUSH and POP operation)

Basically, two main operations performed on stack.

1. **PUSH:** To insert an element on the TOP of the stack is called PUSH operation. Pushing an element in the stack inserts new element at the TOP after every Push operation the TOP is incremented by one. *In case stack is full, it is called stack-full condition or stack overflow condition.*
2. **POP:** To remove an element from the TOP of the stack is called POP operation. After every POP operation TOP is decremented by one. *If there is no element on the stack and POP is performed then this will result in Stack Underflow.*

Algorithm: STACK_PUSH (S, TOP, X)

This algorithm inserts an element on the top of the stack

- S representing array for Stack
- TOP is a variable which points to the top of the stack
- X is the element to be inserted

```
Step1. [check for stack overflow]
        if ( TOP >= N )
        then write ("STACK OVERFLOW")
```

```
        return( )
Step2. [increment TOP]
        TOP = TOP + 1
Step3. [insert element]
        S[TOP] = X
Step4. [finished]
        return( )
```

Description

- In step1, algorithm checks for an overflow, whether the stack is full or not.
- If not, then TOP pointer is incremented in step2, and the element is inserted (pushed) in step3.

Algorithm: STACK_POP (S, TOP)

This algorithm removes an element from the top of the stack

- S representing array of Stack
- TOP is a variable which points to the top of the stack
- Y is a variable which holds the value of deleted element

```
Step1. [check for stack underflow]
        if ( TOP = 0 )
        then write ("STACK UNDERFLOW")
        return(0)
Step2. [delete an element]
        Y ← S[TOP]
Step3. [decrement TOP pointer]
        TOP ← TOP - 1
Step4. [return deleted element]
        return (Y)
```

Description

- In step1, algorithm checks for stack underflow. Because if stack is underflow, there is no item to be removed.
- If not, then popped item is stored in variable Y and TOP is decremented by value one in step3.

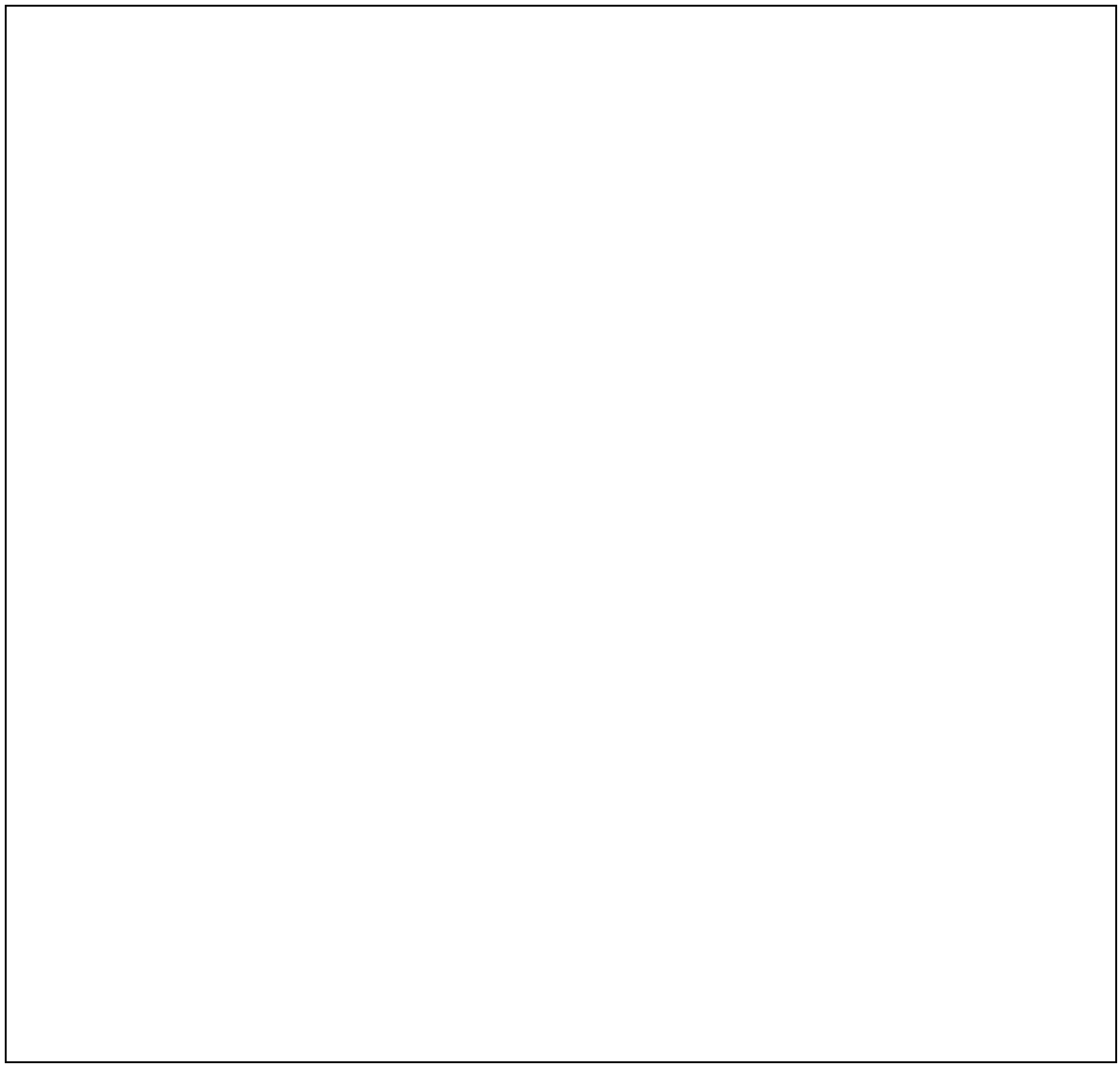
H. Resource / Equipment required

Sr. No.	Equipment / Software resources	Specifications
1	Computer system (Hardware Tool)	Computer having latest configuration with Windows or Unix OS
2	C compiler (Software Tool)	Open-source software

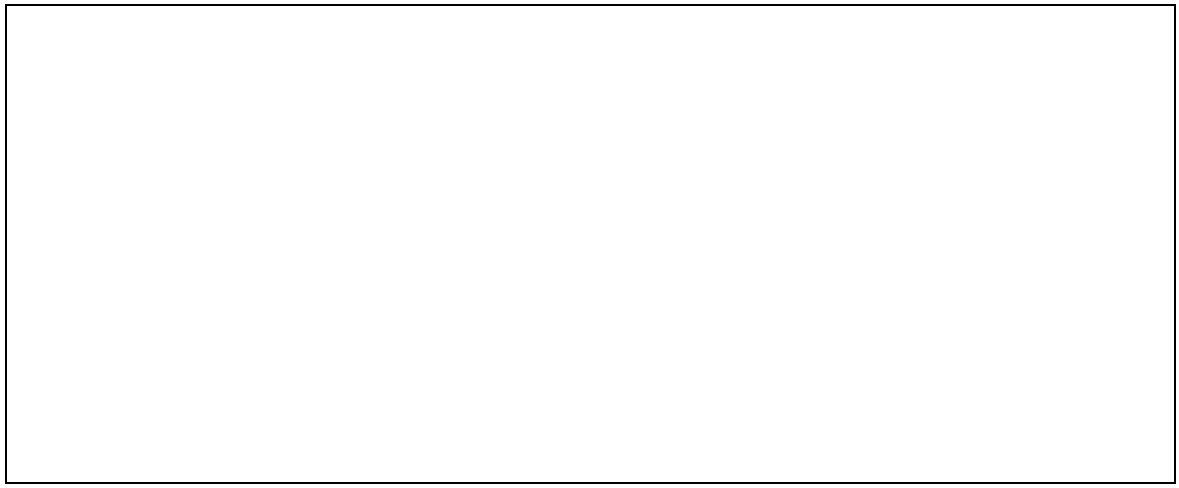
I. Safety and necessary precautions followed

- Shutdown computer system properly once the Lab hours are finished.
- Strictly follow the instructions given by laboratory in-charge.

J. Program Source Code (Implement PUSH and POP algorithms of stack using array).



K. Input – Output



L. Practical related Quiz

1. Write a short note on applications of stack.
2. Fill in the blanks for the following.
   I. In a stack, the operation of adding an element is called _____.
   II. The element that is at the top of the stack is called _____.
   III. A stack follows the _____ principle, which means that the last element added is the first one to be removed.
   IV. The _____ operation in a stack removes the element at the top and returns it.
   V. A stack can be implemented using an _____or a _____.
3. State TRUE (T) / FALSE (F) for the following.
   I. A stack follows the First-In-First-Out (FIFO) principle. _____

II. The operation of adding an element to a stack is called "POP." ☐

III. A stack can be implemented using an array or a linked list. ☐

IV. Stacks support random access to elements. ☐

V. Stacks are primarily used for searching and sorting data. ☐

M. References

I. https://www.javatpoint.com/data-structure-stack
II. https://www.tutorialspoint.com/data_structures_algorithms/stack_algorithm.htm
III. https://chat.openai.com/
IV. https://www.studytonight.com/data-structures/stack-data-structure
V. https://youtu.be/K5JGpZqXtuI

N. Assessment Rubrics

Practical no. 6: Implement PUSH and POP algorithms of stack using array.				
Performance indicators for PROs	**Weighted Marks**	**Max Marks**	**Rubrics**	**Obtained Marks**
Correctness of program	**20%**	**5**	5 to 3 – Correct 2 – Partial correct 1 to 0 – Poor work	
Readability and documentation of the program/Quality of input and output displayed (messaging and formatting)	**20%**	**5**	5 to 4 – Excellent 3 to 2 – Good 1 to 0 – Poor	
Code efficiency	**10%**	**3**	3 – Excellent 2 – Good 1 to 0 – Poor	
Debugging ability	**20%**	**5**	5 to 4 – Excellent 3 to 2 – Good 1 to 0 – Poor	
Program execution/answer to questions	**30%**	**7**	7 to 5 – Proper execution 4 to 2 – Run with some errors 1 to 0 – Not run	
			Total Marks: (Out of 25)	

Signature with date

Date: _____

Practical No. 7: Implement recursive functions.

A Objective:

- To understand and apply techniques to solve complex problems.

B Expected Program Outcomes (POs):

1. **Basic and discipline specific knowledge:**
   Apply knowledge of basic mathematics, science and engineering fundamentals and engineering specialization to solve the engineering problems.
2. **Problem analysis:**
   Identify and analyse well-defined engineering problems using codified standard methods.
3. **Design/ development of solutions:**
   Design solutions for engineering well-defined technical problems and assist with the design of systems components or processes to meet specified needs.

C Expected Skills to be developed based on competency:

This practical is expected to develop the following skills for the industry-identified competency:

- ✓ Write algorithm for stack data structure.
- ✓ Write code for the given problem.
- ✓ Debug program to fix errors.
- ✓ Gain critical thinking towards problem solving.

D Expected Course Outcomes (COs):

Perform Demonstrate algorithms to insert and delete elements from the stack and queue data structure.

E Practical Outcome (PRo):

Implement recursive functions.

F Expected Affective Domain Outcomes (ADOs):

1. Demonstrate working as a leader/a team member.
2. Maintain tools and equipment.
3. Follow ethical practices.

G. Prerequisite theory:

Recursion is a problem-solving approach in which a problem is solved using repeatedly applying the same solution to smaller instances.

Recursive function is a programming technique that allows the programmer to express operations in terms of themselves. Simply we can say that the recursive function is → *function call to itself*.

A very simple example that shows recursion is:

```
void main()
{
        printf("Hello");
        main();
        getch();
}
```

main() function is calling to itself, and program will be in infinite loop.

Different recursive functions are explained below.

- **Factorial number**

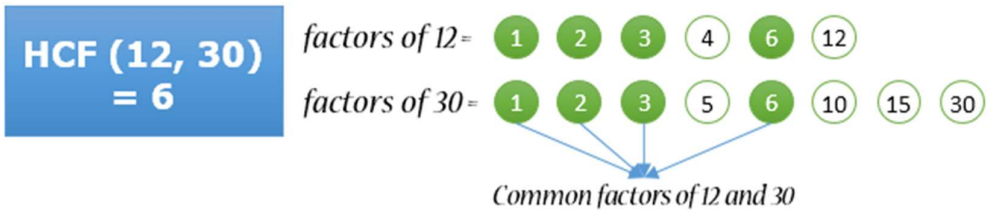
The factorial function can be recursively defined like:

Factorial (n) = 1 (if n =0)

= n * factorial (n-1) (if n != 0)

- **GCD (greatest common divisor)**

- GCD is also known as Greatest Common Factor (GCF) or Highest Common Factor (HCF) is the largest number that divides two or more numbers without a remainder. For example: GCD of 8 and 12 is 4.



- The function of GCD can be recursively defined as:

```
begin
    take two inputs a and b
    repeat while(a ≠ b)
    {
        if(a>b)
            return (gcd(a-b, b))
```

Here function is calling to itself (recursion).

```
            else
                    return (gcd(a, b-a))
        }
        return a                //      (in case of a = b, gcd(a, b) = a)
    end
```

H. Resource / Equipment required

Sr. No.	Equipment / Software resources	Specifications
1	Computer system (Hardware Tool)	Computer having latest configuration with Windows or Unix OS
2	C compiler (Software Tool)	Open-source software

I. Safety and necessary precautions followed

- Shutdown computer system properly once the Lab hours are finished.
- Strictly follow the instructions given by laboratory in-charge.

J. Program Source Code

1) Write a program to find factorial of a given number using recursion.

K. (1) Input – Output

2) Write a program to find GCD of a given number using recursion.

K. (2) Input – Output

L. Practical related Quiz

1. Give the definition of recursion and list any two advantages of recursion.
2. Fill in the blanks for the following.
   I. Recursion is a programming technique where a function calls _____.
   II. _____ is a technique that combines the solutions of smaller subproblems to solve the original problem in recursion.
   III. In a situation of recursion, a function is calling to _____.
3. State TRUE (T) / FALSE (F) for the following.
   I. Stack is the only data structure that can be implemented using recursion.
   II. Recursion always leads to a more efficient solution compared to iteration.
   III. Every problem that can be solved using recursion can also be solved using iteration.

IV. Recursion is always the most efficient approach for solving a problem. ☐

V. Recursion is only applicable to linear data structures like arrays. ☐

M. References

I. https://www.codingninjas.com/codestudio/library/types-of-recursion

II. https://www.simplilearn.com/tutorials/data-structure-tutorial/recursive-algorithm

III. https://www.cseworldonline.com/data-structure/recursion-in-data-structures.php

IV. https://youtu.be/ggk7HbcnLG8

N. Assessment Rubrics

Practical no. 7: Implement recursive functions.				
Performance indicators for PROs	**Weighted Marks**	**Max Marks**	**Rubrics**	**Obtained Marks**
Correctness of program	**20%**	**5**	5 to 3 – Correct 2 – Partial correct 1 to 0 – Poor work	
Readability and documentation of the program/Quality of input and output displayed (messaging and formatting)	**20%**	**5**	5 to 4 – Excellent 3 to 2 – Good 1 to 0 – Poor	
Code efficiency	**10%**	**3**	3 – Excellent 2 – Good 1 to 0 – Poor	
Debugging ability	**20%**	**5**	5 to 4 – Excellent 3 to 2 – Good 1 to 0 – Poor	
Program execution/answer to questions	**30%**	**7**	7 to 5 – Proper execution 4 to 2 – Run with some errors 1 to 0 – Not run	
			Total Marks: (Out of 25)	

Signature with date

Date: _____

Practical No. 8: Implement insert and delete algorithms of a queue using array.

A Objective:

- The objective of learning about queues is to understand and utilize a fundamental data structure that follows the First-In-First-Out (FIFO) principle.
- Enable efficient data processing and problem-solving approach.

B Expected Program Outcomes (POs):

1. **Basic and discipline specific knowledge:**
   Apply knowledge of basic mathematics, science and engineering fundamentals and engineering specialization to solve the engineering problems.
2. **Problem analysis:**
   Identify and analyse well-defined engineering problems using codified standard methods.
3. **Design/ development of solutions:**
   Design solutions for engineering well-defined technical problems and assist with the design of systems components or processes to meet specified needs.

C Expected Skills to be developed based on competency:

This practical is expected to develop the following skills for the industry-identified competency:

- ✓ Write algorithm for queue data structure.
- ✓ Debug program to fix errors.
- ✓ Follow coding guidelines.

D Expected Course Outcomes (COs):

Perform Demonstrate algorithms to insert and delete elements from the stack and queue data structure.

E Practical Outcome (PRo):

Implement insert and delete algorithms of a queue using array.

F Expected Affective Domain Outcomes (ADOs):

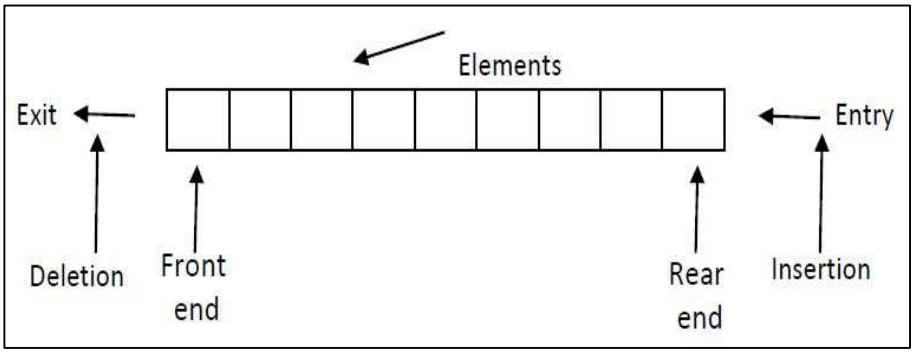
1. Demonstrate working as a leader/a team member.
2. Maintain tools and equipment.
3. Follow ethical practices.

G. Prerequisite theory:

A queue is a non-primitive linear data structure in which insertion operation is performed at one end called **REAR** and deletion operation is performed at another end of the list called **FRONT**.

Examples of Queue: -

1. A raw of students at registration counter.
2. Line of cars waiting to proceeds in some direction at traffic signal, which came first is allowed to come out first.
3. A movie ticket window example.



Basically, two main operations performed on queue.

- **INSERT:** To insert an element at the REAR end of queue is called INSERTION operation.
- **DELETE:** To remove an element at the FRONT end of queue is called DELETION operation.

Algorithm: QINSERT

This algorithm inserts an element into the queue

- Q represents array for QUEUE, F is a variable for front end and R is a variable for rear end, N is maximum number of elements.
- Initially F and R set to 0 (queue is empty)
- Y is the element to be inserted

```
Step1. [check for queue overflow]
        if ( R >= N )
        then write ("QUEUE OVERFLOW")
        return ( )
Step2. [increment REAR pointer]
```

```
        R  = R + 1
Step3. [insert element]
        Q[R] = Y
Step4. [set the FRONT]
        if ( F = 0 )
        then F = 1
        return( )
```

Explanation

- In step 1, check for the overflow.
- Step 2, incrementing rear pointer and then in Step 3, insert an element.
- Step 4, set the front pointer properly.

Algorithm: QDELETE

This algorithm deletes an element from the queue.

- Q represents array for QUEUE, F is a variable for front end and R is a variable for rear end. Y variable stores the deleted value.

```
Step1. [check for queue underflow]
        if ( F = 0 )
        then write ("QUEUE UNDERFLOW")
        return ( )
Step2. [delete element]
        Y = Q[F]
Step3. [check for queue empty after deletion]
        if ( F = R )
        then F = R = 0
        else   F = F+1
Step4. [return deleted element]
        return (Y)
```

Explanation

- In step 1, check for the queue underflow.
- Step 2, deleting an element.
- Step 3, check for the queue empty or not.
- Step 4, returning the deleted element.

H. Resource / Equipment required

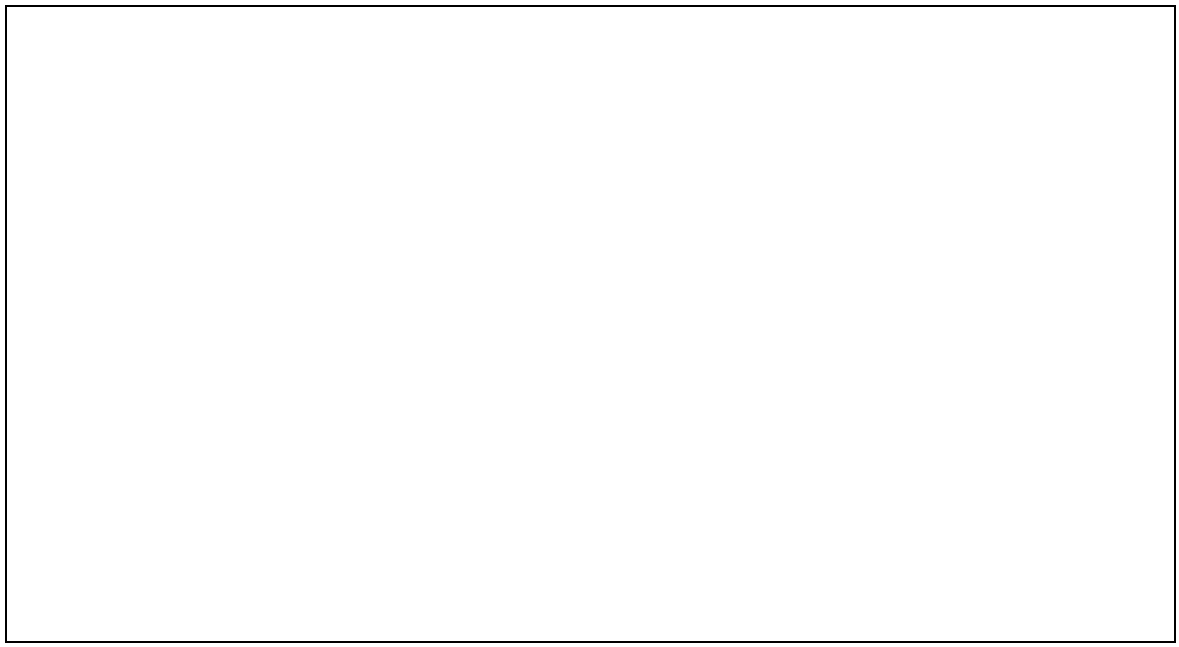
Sr. No.	Equipment / Software resources	Specifications
1	Computer system (Hardware Tool)	Computer having latest configuration with Windows or Unix OS
2	C compiler (Software Tool)	Open-source software

I. Safety and necessary precautions followed

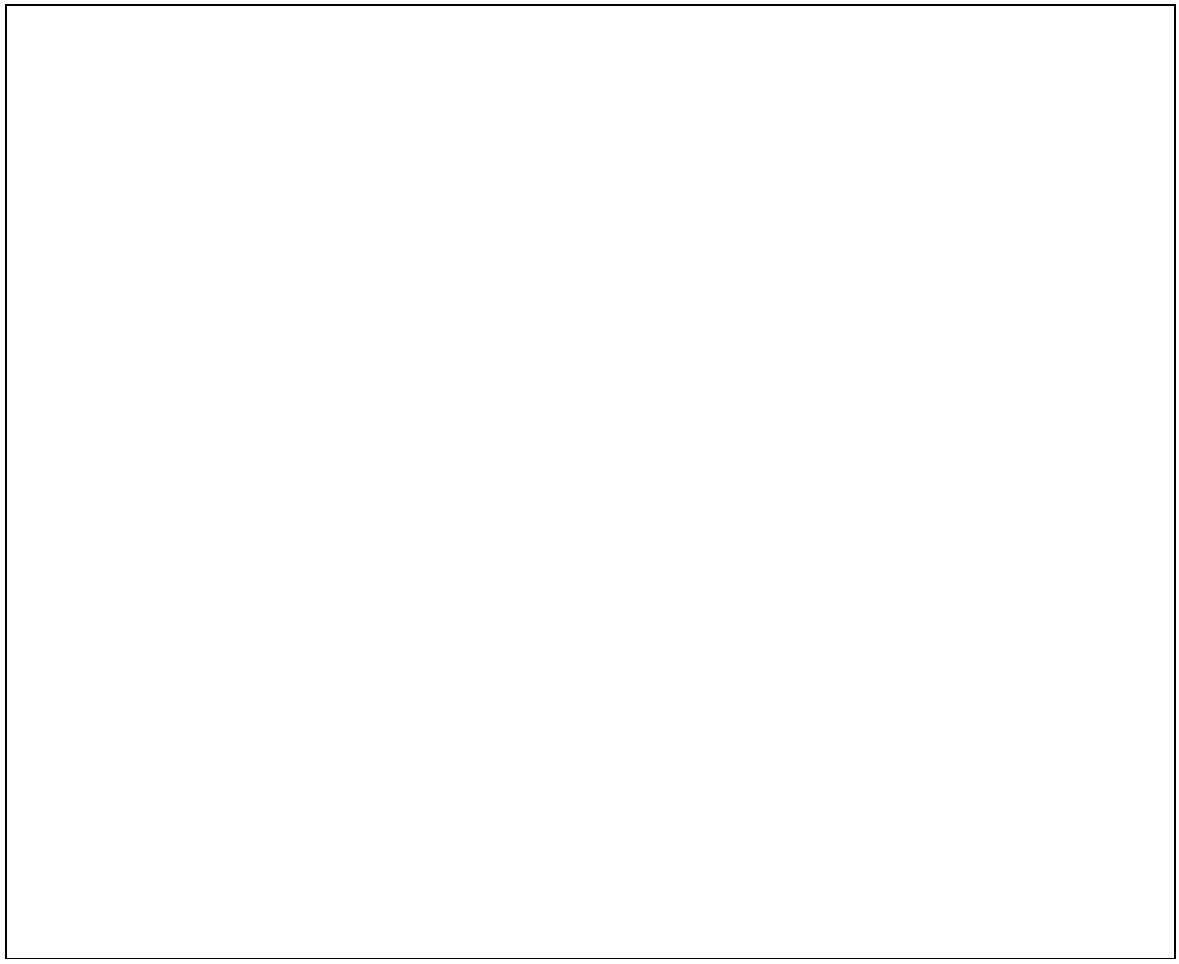
- Shutdown computer system properly once the Lab hours are finished.

- Strictly follow the instructions given by laboratory in-charge.

J. Program Source Code (Implement a program to perform insert and delete algorithms / operations of a queue data structure)



K. Input – Output



L. Practical related Quiz

1. Define: queue, circular queue.
2. Explain applications of queue in brief.
3. Differentiate between simple queue and circular queue.
4. Fill in the blanks for the following.
   I. A queue follows the _____ principle, which means that the first element added is the first one to be removed.
   II. In a queue, the insertion operation performed at _____ end.
   III. Initially when the queue is empty, the F and R ends of a queue set to _____.
5. State TRUE (T) / FALSE (F) for the following.
   I. No wastage of memory in case of simple queue. ☐
   II. In a simple queue, deletion operation is performed at REAR end. ☐
   III. Queue is a kind of FILO (First In Last Out) data structure ☐

M. References

I. https://www.tutorialspoint.com/data_structures_algorithms/dsa_queue.htm
II. https://www.javatpoint.com/data-structure-queue
III. https://www.shiksha.com/online-courses/articles/queue-data-structure-types-implementation-applications/
IV. https://youtu.be/UbAEP7P0vfk

N. Assessment Rubrics

Practical no. 8: Implement insert and delete algorithms of a queue using array.				
Performance indicators for PROs	**Weighted Marks**	**Max Marks**	**Rubrics**	**Obtained Marks**
Correctness of program	**20%**	**5**	5 to 3 – Correct 2 – Partial correct 1 to 0 – Poor work	
Readability and documentation of the program/Quality of input and output displayed (messaging and formatting)	**20%**	**5**	5 to 4 – Excellent 3 to 2 – Good 1 to 0 – Poor	
Code efficiency	**10%**	**3**	3 – Excellent 2 – Good 1 to 0 – Poor	
Debugging ability	**20%**	**5**	5 to 4 – Excellent 3 to 2 – Good 1 to 0 – Poor	
Program execution/answer to questions	**30%**	**7**	7 to 5 – Proper execution 4 to 2 – Run with some errors 1 to 0 – Not run	
			Total Marks: (Out of 25)	

Signature with date

Date: _____

Practical No. 9: Implement simple structure program using pointer.

A Objective:

- The objective of learning data structures in C is to understand the organization and manipulation of data in an efficient and organized manner.
- Acquire memory management and code modularity approach.

B Expected Program Outcomes (POs):

1. **Basic and discipline specific knowledge:**
   Apply knowledge of basic mathematics, science and engineering fundamentals and engineering specialization to solve the engineering problems.
2. **Design/ development of solutions:**
   Design solutions for engineering well-defined technical problems and assist with the design of systems components or processes to meet specified needs.

C Expected Skills to be developed based on competency:

This practical is expected to develop the following skills for the industry-identified competency:

- ✓ Write code for the given problem.
- ✓ Debug and troubleshoot the program.
- ✓ Follow coding guidelines.

D Expected Course Outcomes (COs):

Apply basic operations on the linked list data structure.

E Practical Outcome (PRo):

Implement simple structure program using pointer.

F Expected Affective Domain Outcomes (ADOs):

1. Follow safety practices.
2. Demonstrate working as a leader/a team member.
3. Maintain tools and equipment.
4. Follow ethical practices.

G. Prerequisite theory:

Concept of pointer

Pointer is a variable that stores the address of another variable. It is a derived data type.

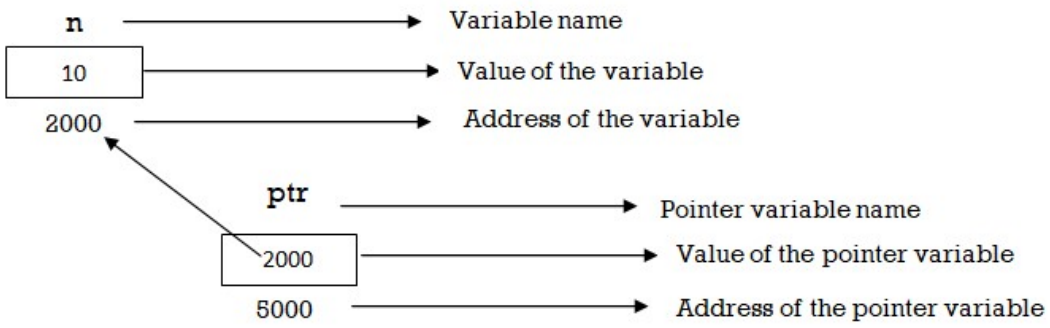
To declare the pointer (syntax):

*data_type *variable_name* Ex : int *ptr

We can also initialize the pointer at the time of declaration, like :

```
int n = 10;
int *ptr = &n;          // here, pointer 'ptr' points to the variable n.
```

Above concept can be represented in below figure.



In above figure, we can see that the value of the pointer is the address of another variable.

The '*' operator is used for *'value at address'* notation and '&' operator used as *'address of'* the variable. '*' is also called *dereferencing operator.*

Concept of structure

Structure is a collection of logically related data items of different data types.

All the data items of structure are known by a single name.

Syntax (prototype) of structure

```
struct structure_name
        {
        data_type var1;
        data_type var2;
        ----
        data_type var n;
```

(these are called the structure members)

```
};
```

- To access the members of the structures, we need to create the structure variable.
- To create the structure variable, the syntax is :

struct structure_name *structure_varname1, structure_varname2,,,, ;*

- We can create or declare structure variables by following three different ways:
  - With template itself
  - Anywhere in the program
  - Array of structures

By using the 'dot operator' (.) with structure variable, we can access the structure members.

H. Resource / Equipment required

Sr. No.	Equipment / Software resources	Specifications
1	Computer system (Hardware Tool)	Computer having latest configuration with Windows or Unix OS
2	C compiler (Software Tool)	Open-source software

I. Safety and necessary precautions followed

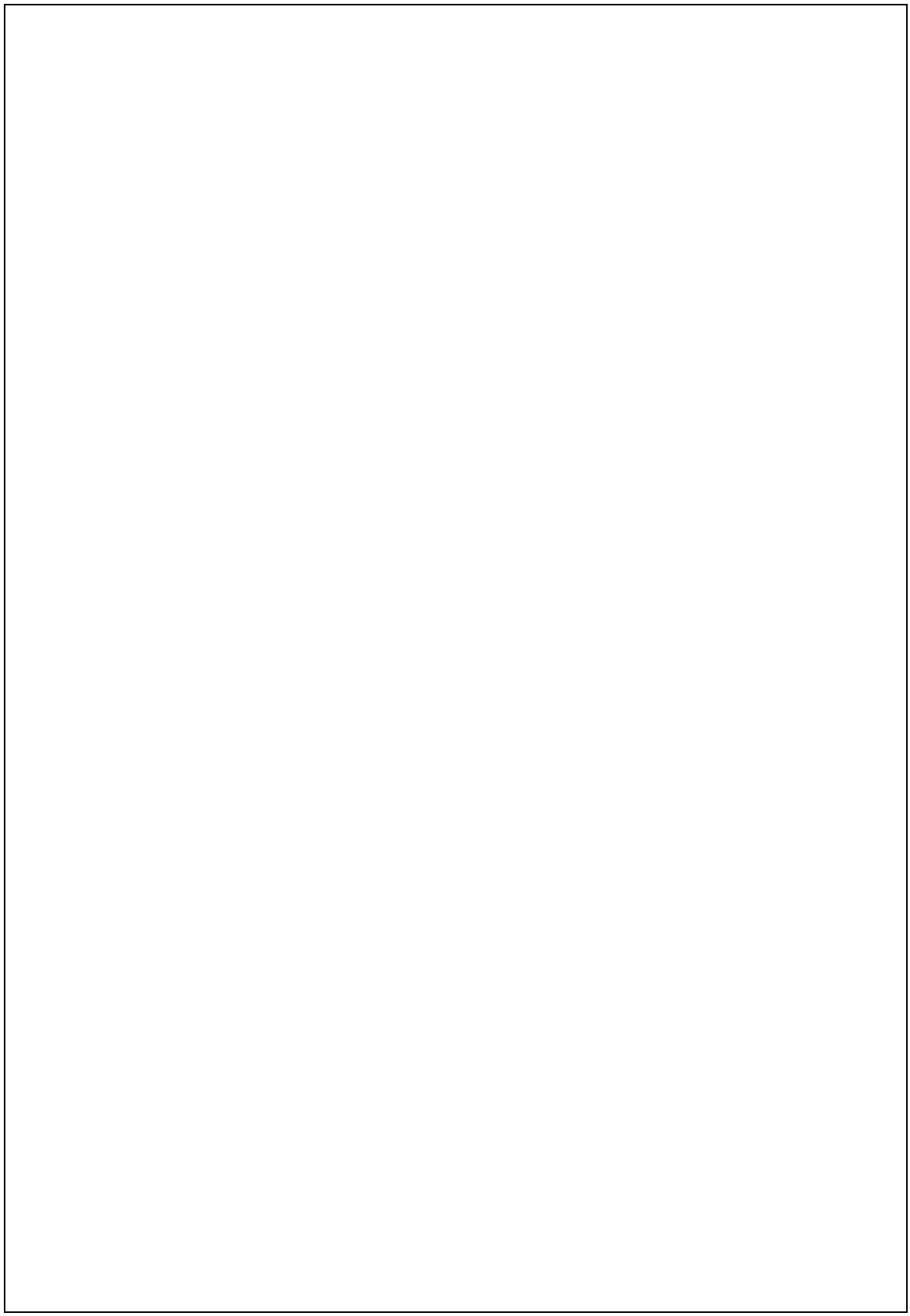
- Shutdown computer system properly once the Lab hours are finished.
- Strictly follow the instructions given by laboratory in-charge.

J. Program Source Code

1) Write a program using pointer

K. (1) Input – Output

2) Implement simple structure program using pointer.



K. (2) Input – Output

L. Practical related Quiz

1. Define pointer. Write its advantages and disadvantages.
2. Define structure.
3. Define DMA. List various library functions used for DMA.
4. Fill in the blanks for the following.
   I. In data structures, a pointer is a variable that stores the _____ of another variable or data structure.
   II. The free function is used to deallocate dynamically allocated memory, taking a _____ as its argument.
   III. The arrow operator (->) is used to access the _____ of a structure or class through a pointer.
   IV. A structure definition begins with the _____ keyword, followed by the

name of the structure.

V. The variables inside a structure are called _____.

5. State TRUE (T) / FALSE (F) for the following.

I. Dereferencing a null pointer in C/C++ will result in a runtime error. ☐

II. In a simple queue, deletion operation is performed at REAR end. ☐

III. The value of a pointer can be changed after it has been initialized. ☐

IV. Structures cannot be nested within other structures. ☐

V. Structures in C/C++ support inheritance and polymorphism. ☐

M. References

I. https://www.javatpoint.com/data-structure-pointer
II. https://www.educba.com/pointers-in-data-structure/
III. https://www.geeksforgeeks.org/features-and-use-of-pointers-in-c-c/
IV. https://youtu.be/HnnvR-a1sqw

N. Assessment Rubrics

Practical no. 9: Implement simple structure programs using pointer.				
Performance indicators for PROs	**Weighted Marks**	**Max Marks**	**Rubrics**	**Obtained Marks**
Correctness of program	**20%**	**5**	5 to 3 – Correct 2 – Partial correct 1 to 0 – Poor work	
Readability and documentation of the program/Quality of input and output displayed (messaging and formatting)	**20%**	**5**	5 to 4 – Excellent 3 to 2 – Good 1 to 0 – Poor	
Code efficiency	**10%**	**3**	3 – Excellent 2 – Good 1 to 0 – Poor	
Debugging ability	**20%**	**5**	5 to 4 – Excellent 3 to 2 – Good 1 to 0 – Poor	
Program execution/answer to questions	**30%**	**7**	7 to 5 – Proper execution 4 to 2 – Run with some errors 1 to 0 – Not run	
			Total Marks: (Out of 25)	

Signature with date

Date: _____

Practical No. 10: Implement insertion of node in the beginning of the list in singly linked list.

A Objective:

- The primary objective of learning about a singly linked list is to understand and be able to implement a fundamental data structure used in computer science.

B Expected Program Outcomes (POs):

1. **Basic and discipline specific knowledge:**
   Apply knowledge of basic mathematics, science and engineering fundamentals and engineering specialization to solve the engineering problems.
2. **Design/ development of solutions:**
   Design solutions for engineering well-defined technical problems and assist with the design of systems components or processes to meet specified needs.

C Expected Skills to be developed based on competency:

This practical is expected to develop the following skills for the industry-identified competency:

- ✓ Write code for the given problem.
- ✓ Debug program to fix errors.
- ✓ Follow coding guidelines.

D Expected Course Outcomes (COs):

Apply basic operations on the linked list data structure.

E Practical Outcome (PRo):

Implement insertion of node in the beginning of the list in singly linked list.

F Expected Affective Domain Outcomes (ADOs):

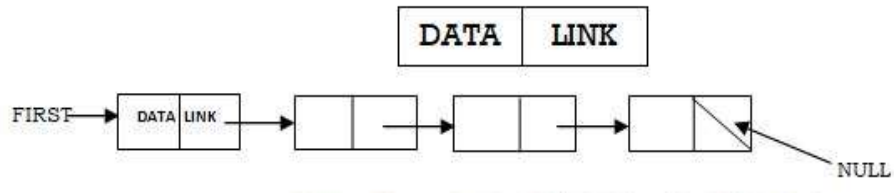
1. Follow safety practices.
2. Demonstrate working as a leader/a team member.
3. Maintain tools and equipment.
4. Follow ethical practices.

G. Prerequisite theory:

Linked list is a dynamic linear data structure which consists of group of nodes. Each node contains two fields:

1. DATA field: which contains the actual data
2. LINK field: which contains the pointer to the next node

The representation of linked list is shown as below:



In the linked list → individual element is called 'node' and it is represented using structure in C. Linked list can be represented with following declaration.

```
struct node
{
        int DATA;
        srtuct node * LINK;    // reference to itself (self-referential structure)
};
```

Algorithm: INSERT_START_SINGLY

This algorithm inserts a new node NEW with value X at the beginning of linked list

- FIRST is a pointer to the first element of linked list which contain DATA and LINK fields.
- NEW is a pointer variable which denotes new node.

```
Step1. [obtain new node from memory]
        NEW ← NODE          // obtain a new node from memory using DMA
        [ NEW = (struct NODE*) malloc (sizeof (struct NODE) ]
        if (NEW = NULL)
        then    print ("NOT ENOUGH MEMORY")
                return( )
Step2. [initialize new node]
        DATA(NEW) = X
```

```
        LINK(NEW) = NULL
Step3. [check for empty list]
        if ( FIRST = NULL )
        then    return (NEW)
Step4. [when list is not empty, insert the new node at beginning]
        LINK(NEW) = FIRST
        FIRST = NEW
        return (FIRST)
```

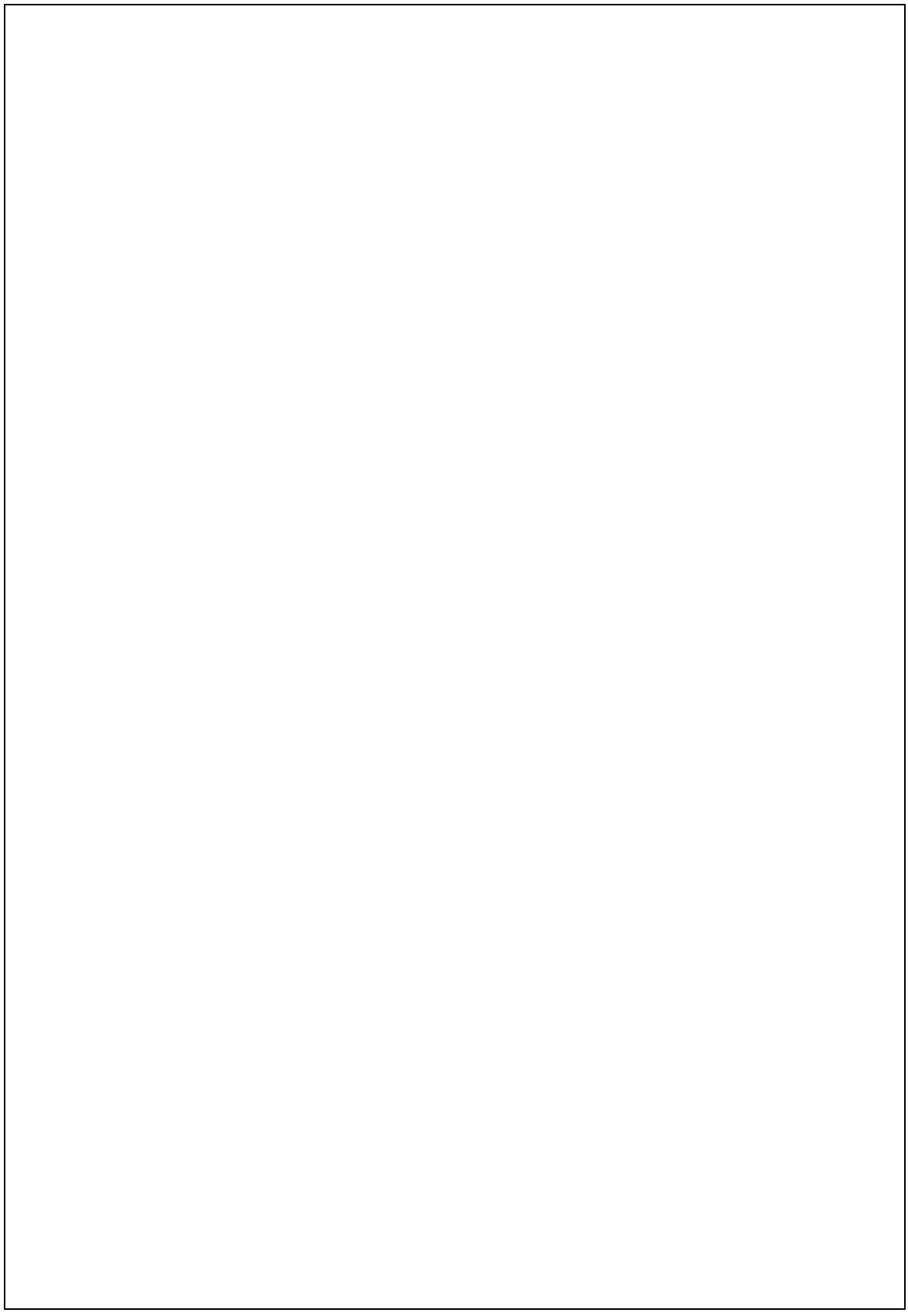
H. Resource / Equipment required

Sr. No.	Equipment / Software resources	Specifications
1	Computer system (Hardware Tool)	Computer having latest configuration with Windows or Unix OS
2	C compiler (Software Tool)	Open-source software

I. Safety and necessary precautions followed

- Shutdown computer system properly once the Lab hours are finished.
- Strictly follow the instructions given by laboratory in-charge.

J. Program Source Code (Implement insertion of node in the beginning of the list in singly linked list)



K. Input – Output

L. Practical related Quiz

1. Define linked list. List various types of linked list.
2. Write advantages and disadvantages of linked list.
3. Identify differences between array and linked list.
4. Fill in the blanks for the following.
   I. In a linked list, each node contains two parts: _____ and _____
   II. The first node in a linked list is called the _____
   III. The _____ is a special value that indicates the end of a linked list.
   IV. In a doubly linked list, each node contains a pointer/reference to the _____ and the _____.

V. A linked list that has no cycles is called a _____ linked list.

5. State TRUE (T) / FALSE (F) for the following.

I. Dereferencing a null pointer in C/C++ will result in a runtime error. ☐

II. In a simple queue, deletion operation is performed at REAR end. ☐

III. The value of a pointer can be changed after it has been initialized. ☐

IV. Structures cannot be nested within other structures. ☐

V. Structures in C/C++ support inheritance and polymorphism. ☐

M. References

I. https://www.programiz.com/dsa/linked-list
II. https://examradar.com/linked-list-concepts/
III. https://chat.openai.com/
IV. https://www.freecodecamp.org/news/data-structures-explained-with-examples-linked-list/
V. https://www.youtube.com/watch?v=jgqg6Qw68_Q

N. Assessment Rubrics

Practical no. 10: Implement insertion of node in the beginning of the list in singly linked list.				
Performance indicators for PROs	**Weighted Marks**	**Max Marks**	**Rubrics**	**Obtained Marks**
Correctness of program	**20%**	**5**	5 to 3 – Correct 2 – Partial correct 1 to 0 – Poor work	
Readability and documentation of the program/Quality of input and output displayed (messaging and formatting)	**20%**	**5**	5 to 4 – Excellent 3 to 2 – Good 1 to 0 – Poor	
Code efficiency	**10%**	**3**	3 – Excellent 2 – Good 1 to 0 – Poor	
Debugging ability	**20%**	**5**	5 to 4 – Excellent 3 to 2 – Good 1 to 0 – Poor	
Program execution/answer to questions	**30%**	**7**	7 to 5 – Proper execution 4 to 2 – Run with some errors 1 to 0 – Not run	
			Total Marks: (Out of 25)	

Signature with date

Date: _____

Practical No. 11: Implement insertion of node at the end of the list in singly linked list.

A Objective:

- The primary objective of learning about a singly linked list is to understand and be able to implement a fundamental data structure used in computer science.

B Expected Program Outcomes (POs):

1. **Basic and discipline specific knowledge:**
   Apply knowledge of basic mathematics, science and engineering fundamentals and engineering specialization to solve the engineering problems.
2. **Design/ development of solutions:**
   Design solutions for engineering well-defined technical problems and assist with the design of systems components or processes to meet specified needs.

C Expected Skills to be developed based on competency:

This practical is expected to develop the following skills for the industry-identified competency:

- ✓ Write code for the given problem.
- ✓ Debug program to fix errors.
- ✓ Follow coding guidelines.

D Expected Course Outcomes (COs):

Apply basic operations on the linked list data structure.

E Practical Outcome (PRo):

Implement insertion of node at the end of the list in singly linked list.

F Expected Affective Domain Outcomes (ADOs):

1. Follow safety practices.
2. Demonstrate working as a leader/a team member.
3. Maintain tools and equipment.
4. Follow ethical practices.

G. Prerequisite theory:

Algorithm: INSERT_END_SINGLY

This algorithm inserts a new node NEW with value X at the end of singly linked list.

- FIRST is a pointer to the first element of linked list which contain DATA and LINK fields.
- NEW and SAVE are temporary pointer variables. NEW denotes new node.

```
Step1. [obtain new node from memory]
        NEW ← NODE          // obtain a new node from memory using DMA
        [ NEW = (struct NODE*) malloc (sizeof (struct NODE) ]
        if (NEW = NULL)
        then    print ("NOT ENOUGH MEMORY")
                return( )
Step2. [initialize new node]
        DATA(NEW) = X
        LINK(NEW) = NULL
Step3. [check for empty list]
        if ( FIRST = NULL )
        then    return (NEW)
Step4. [initialize SAVE and  search for end of the list]
        SAVE = FIRST
        repeat while   ( LINK (SAVE) ≠ NULL )
                        SAVE = LINK (SAVE)
Step5. [set LINK field of last node to NEW]
        LINK (SAVE) = NEW
        return (FIRST)
```

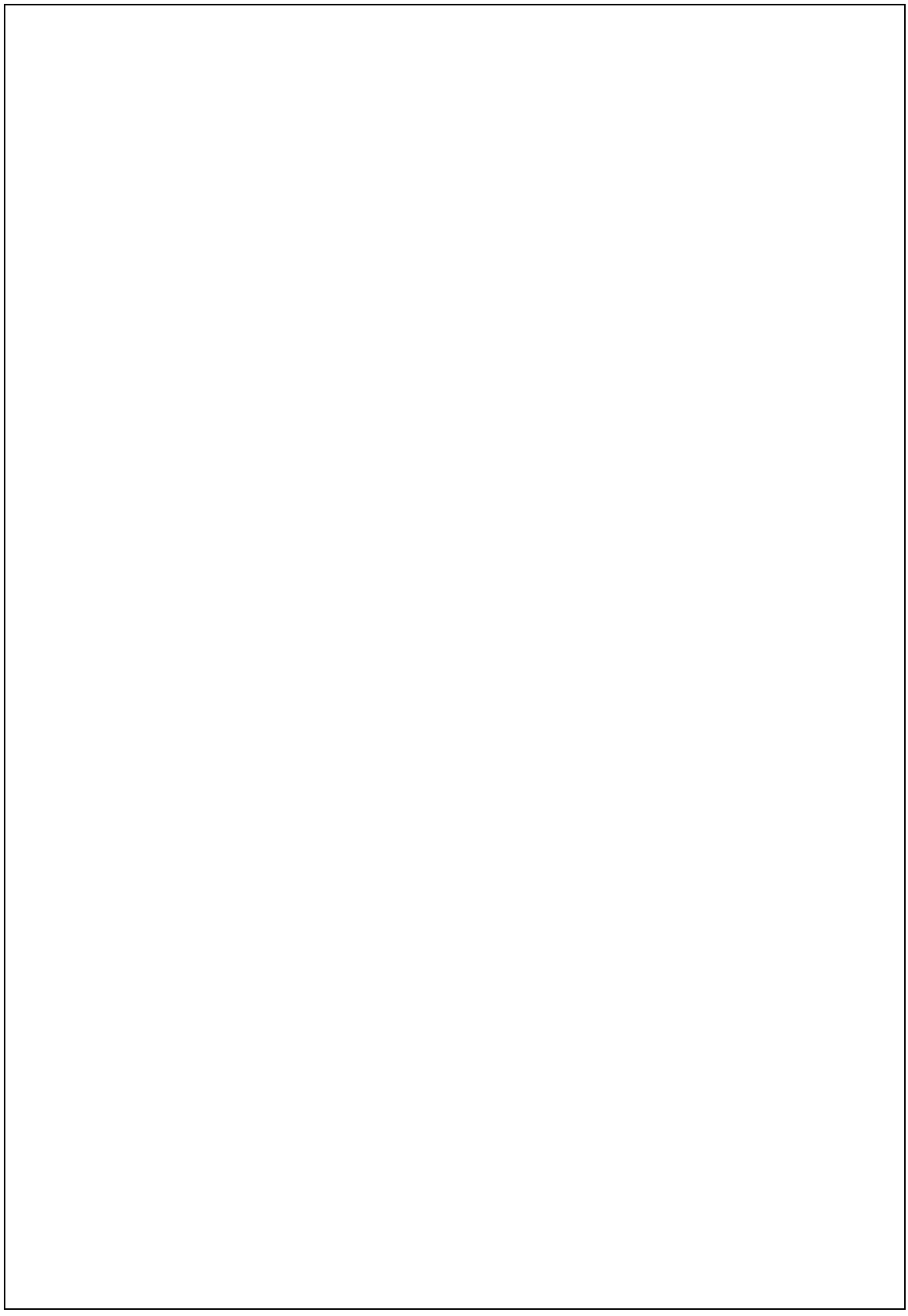
H. Resource / Equipment required

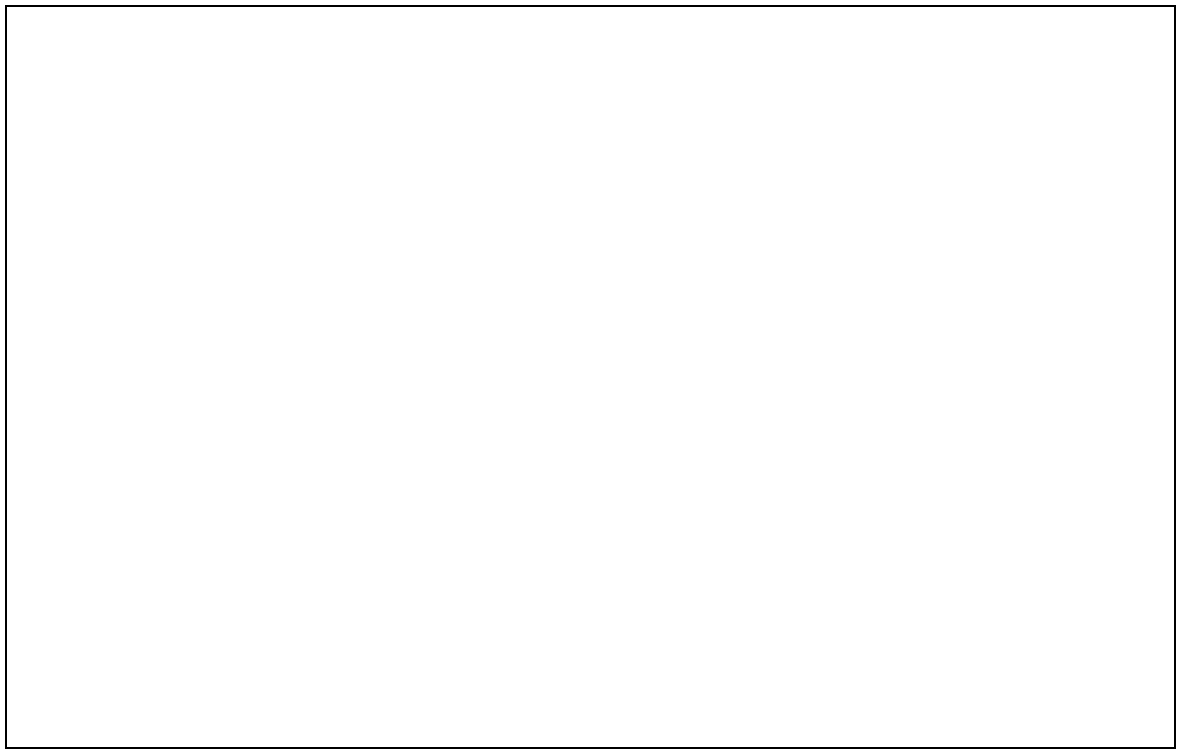
Sr. No.	Equipment / Software resources	Specifications
1	Computer system (Hardware Tool)	Computer having latest configuration with Windows or Unix OS
2	C compiler (Software Tool)	Open-source software

I. Safety and necessary precautions followed

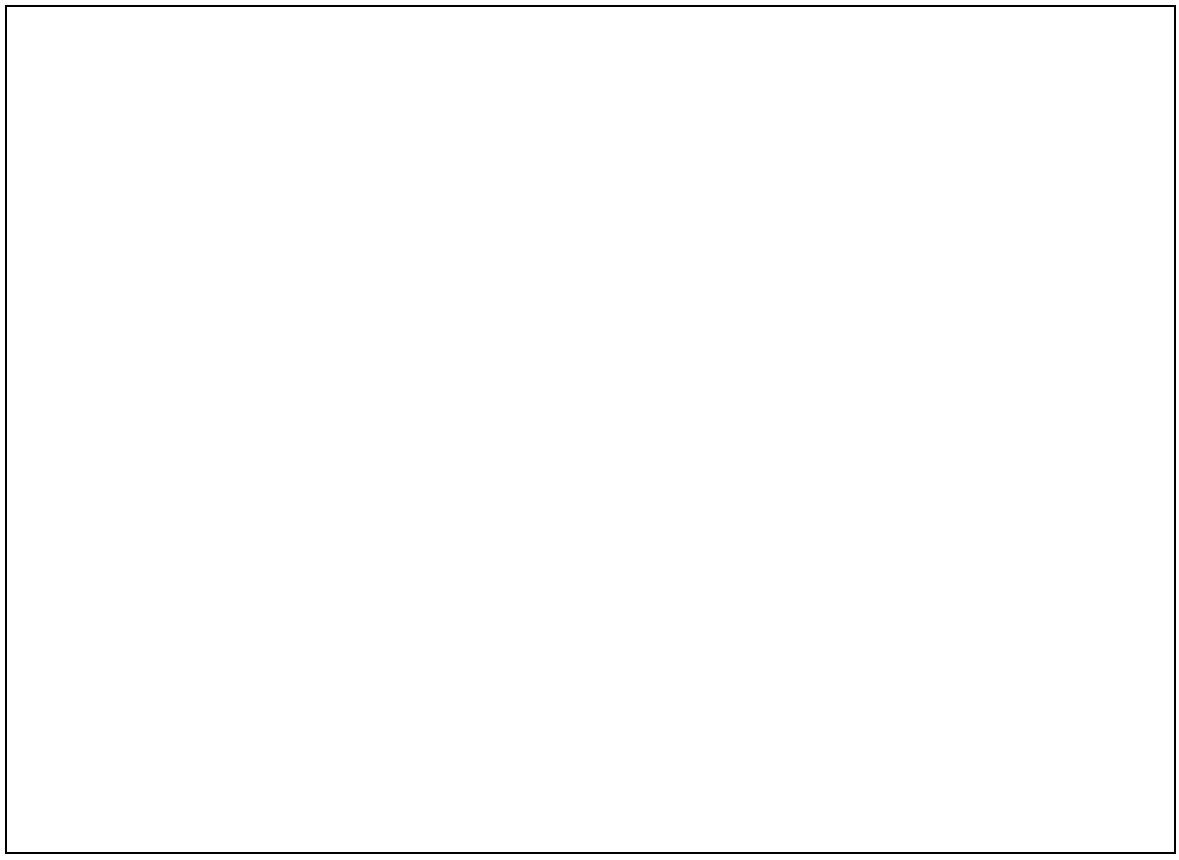
- Shutdown computer system properly once the Lab hours are finished.
- Strictly follow the instructions given by laboratory in-charge.

J. Program Source Code (Implement insertion of node at the end of the list in singly linked list)





K. Input – Output



L. References

I. https://www.programiz.com/dsa/linked-list
II. https://examradar.com/linked-list-concepts/
III. https://chat.openai.com/
IV. https://www.freecodecamp.org/news/data-structures-explained-with-examples-linked-list/
V. https://www.youtube.com/watch?v=ZTTy6jTTJUI

M. Assessment Rubrics

Practical no. 11: Implement insertion of node in the end of the list in singly linked list.				
Performance indicators for PROs	**Weighted Marks**	**Max Marks**	**Rubrics**	**Obtained Marks**
Correctness of program	**20%**	**5**	5 to 3 – Correct 2 – Partial correct 1 to 0 – Poor work	
Readability and documentation of the program/Quality of input and output displayed (messaging and formatting)	**20%**	**5**	5 to 4 – Excellent 3 to 2 – Good 1 to 0 – Poor	
Code efficiency	**10%**	**3**	3 – Excellent 2 – Good 1 to 0 – Poor	
Debugging ability	**20%**	**5**	5 to 4 – Excellent 3 to 2 – Good 1 to 0 – Poor	
Program execution/answer to questions	**30%**	**7**	7 to 5 – Proper execution 4 to 2 – Run with some errors 1 to 0 – Not run	
			Total Marks: (Out of 25)	

Signature with date

Date: _____

Practical No. 12: Implement insertion of node in sorted linked list.

A Objective:

- The primary objective of learning about a singly linked list is to understand and be able to implement a fundamental data structure used in computer science.

B Expected Program Outcomes (POs):

1. **Basic and discipline specific knowledge:**
   Apply knowledge of basic mathematics, science and engineering fundamentals and engineering specialization to solve the engineering problems.
2. **Design/ development of solutions:**
   Design solutions for engineering well-defined technical problems and assist with the design of systems components or processes to meet specified needs.

C Expected Skills to be developed based on competency:

This practical is expected to develop the following skills for the industry-identified competency:

- ✓ Write code for the given problem.
- ✓ Debug program to fix errors.
- ✓ Follow coding guidelines.

D Expected Course Outcomes (COs):

Apply basic operations on the linked list data structure.

E Practical Outcome (PRo):

Implement insertion of node in sorted linked list.

F Expected Affective Domain Outcomes (ADOs):

1. Follow safety practices.
2. Demonstrate working as a leader/a team member.
3. Maintain tools and equipment.
4. Follow ethical practices.

G. Prerequisite theory:

Algorithm: INSERT_ORD_SINGLY

This algorithm inserts a new node in ordered (sorted) linked list at proper position.

- X is a value of NEW node to be inserted, FIRST is a pointer pointing to the first node of linked list.
- NEW and SAVE are temporary pointer variables. *NEW denotes new node.*

```
Step1. [obtain new node from memory]
        NEW ← NODE          // obtain a new node from memory using DMA
        [ NEW = (struct NODE*) malloc (sizeof(struct NODE) ]
        if (NEW = NULL)
        then    print ("NOT ENOUGH MEMORY")
                return( )
Step2. [initialize new node]
        DATA(NEW) = X
        LINK(NEW) = NULL
Step3. [check for empty list]
        if (FIRST = NULL)
        then    return (NEW)
Step4. [does the new node precede all other nodes?]
        if (DATA (NEW) ≤ DATA (FIRST))
        then    LINK (NEW) = FIRST
                FIRST = NEW
                return (FIRST)
Step5. [initialize SAVE pointer and search the proper place for insertion]
        SAVE = FIRST
        repeat while (LINK(SAVE) ≠ NULL AND DATA (LINK (SAVE)) ≤ DATA   (NEW) )
                SAVE  ← LINK (SAVE)
Step6. [insert the new node at its proper place]
        LINK(NEW) ← LINK(SAVE)
        LINK(SAVE) ← NEW
```

```
        return (FIRST)
```

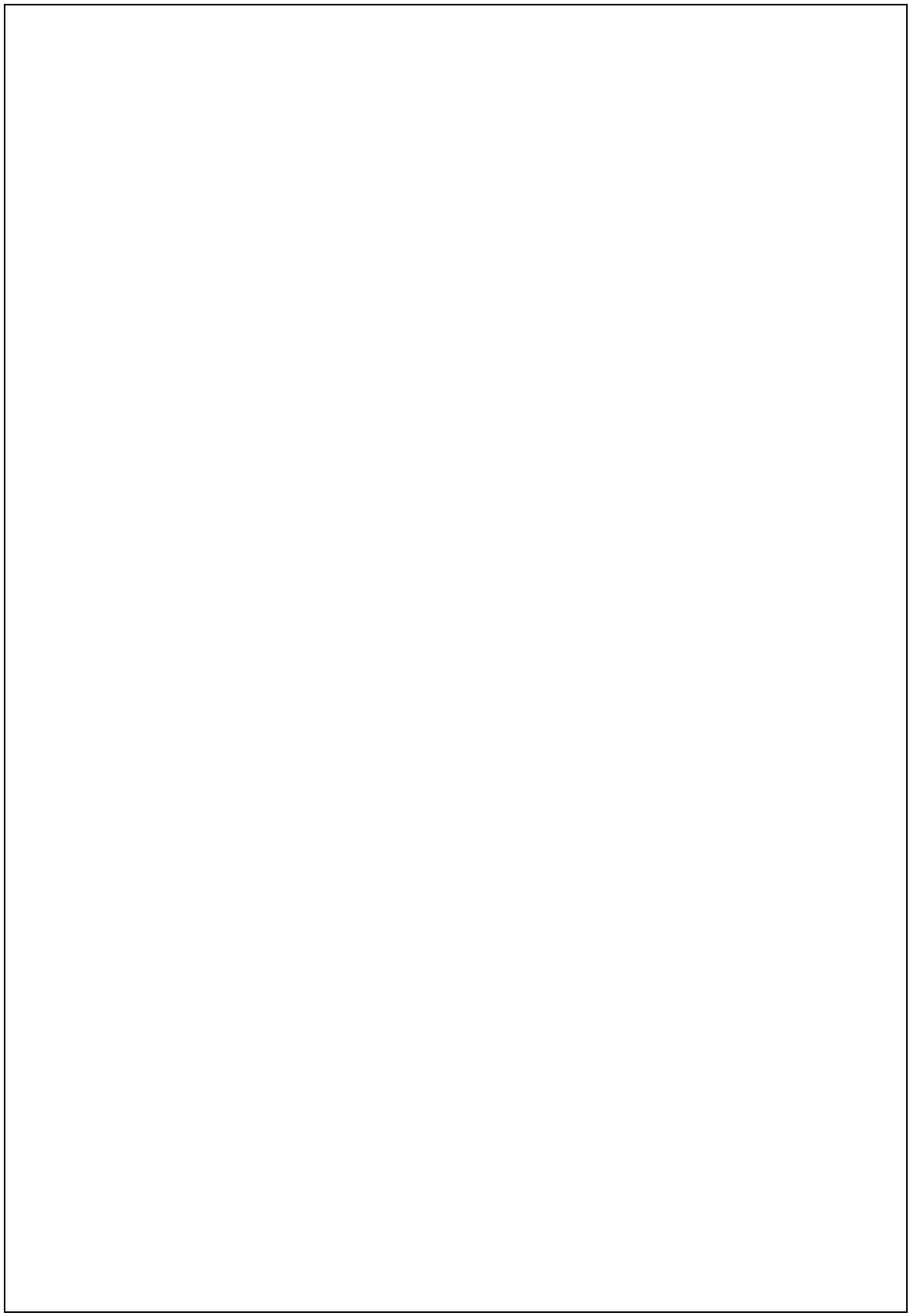
H. Resource / Equipment required

Sr. No.	Equipment / Software resources	Specifications
1	Computer system (Hardware Tool)	Computer having latest configuration with Windows or Unix OS
2	C compiler (Software Tool)	Open-source software

I. Safety and necessary precautions followed

- Shutdown computer system properly once the Lab hours are finished.
- Strictly follow the instructions given by laboratory in-charge.

J. Program Source Code (Implement insertion of node in sorted linked list)



K. Input - Output

L. References

I. https://www.programiz.com/dsa/linked-list
II. https://examradar.com/linked-list-concepts/
III. https://chat.openai.com/
IV. https://www.freecodecamp.org/news/data-structures-explained-with-examples-linked-list/
V. https://www.youtube.com/watch?v=S4PGc9S8pq8

M. Assessment Rubrics

Practical no. 12: Implement insertion of node in sorted linked list.				
Performance indicators for PROs	**Weighted Marks**	**Max Marks**	**Rubrics**	**Obtained Marks**
Correctness of program	**20%**	**5**	5 to 3 – Correct 2 – Partial correct 1 to 0 – Poor work	
Readability and documentation of the program/Quality of input and output displayed (messaging and formatting)	**20%**	**5**	5 to 4 – Excellent 3 to 2 – Good 1 to 0 – Poor	
Code efficiency	**10%**	**3**	3 – Excellent 2 – Good 1 to 0 – Poor	
Debugging ability	**20%**	**5**	5 to 4 – Excellent 3 to 2 – Good 1 to 0 – Poor	
Program execution/answer to questions	**30%**	**7**	7 to 5 – Proper execution 4 to 2 – Run with some errors 1 to 0 – Not run	
			Total Marks: (Out of 25)	

Signature with date

Date: _____

Practical No. 13: Implement insertion of node at any position in linked list.

A Objective:

- The primary objective of learning about a singly linked list is to understand and be able to implement a fundamental data structure used in computer science.

B Expected Program Outcomes (POs):

1. **Basic and discipline specific knowledge:**
   Apply knowledge of basic mathematics, science and engineering fundamentals and engineering specialization to solve the engineering problems.
2. **Design/ development of solutions:**
   Design solutions for engineering well-defined technical problems and assist with the design of systems components or processes to meet specified needs.

C Expected Skills to be developed based on competency:

This practical is expected to develop the following skills for the industry-identified competency:

- ✓ Write code for the given problem.
- ✓ Debug program to fix errors.
- ✓ Follow coding guidelines.

D Expected Course Outcomes (COs):

Apply basic operations on the linked list data structure.

E Practical Outcome (PRo):

Implement insertion of node at any position in linked list.

F Expected Affective Domain Outcomes (ADOs):

1. Follow safety practices.
2. Demonstrate working as a leader/a team member.
3. Maintain tools and equipment.
4. Follow ethical practices.

G. Prerequisite theory:

Algorithm: INSERT_POS__BEFORE_SINGLY

This procedure inserts a new node NEW into linked list before node which is specified by address ADD.

- FIRST is a pointer to the first element of linked list which contain DATA and LINK fields.
- NEW and SAVE are temporary pointer variables. NEW denotes new node
- PRED is a pointer pointing to predecessor node.

```
Step1. [obtain new node from memory]
        NEW ← NODE          // obtain a new node from memory using DMA
        [ NEW = (struct NODE*) malloc (sizeof (struct NODE) ]
        if (NEW = NULL)
        then    print ("NOT ENOUGH MEMORY")
                return( )
Step2. [initialize new node]
        DATA(NEW) = X
        LINK(NEW) = NULL
Step3. [check for empty list]
        if ( FIRST = NULL )
        then    return (NEW)
Step4. [if only one node is there in list]
        if (LINK (FIRST) = NULL OR ADD = FIRST)
        then    LINK (NEW) = FIRST
                FIRST = NEW
                return (FIRST)
Step5. [initialize SAVE and search the list until desired node ADD found]
        SAVE = FIRST
        repeat while (LINK (SAVE) ≠ NULL AND SAVE ≠ ADD)
                PRED = SAVE
                SAVE = LINK (SAVE)
Step6. [inset node when we find targeted node ADD]
```

```
    LINK (PRED) = NEW
    LINK(NEW) = SAVE   // OR  LINK(NEW) = ADD
    return (FIRST)
```

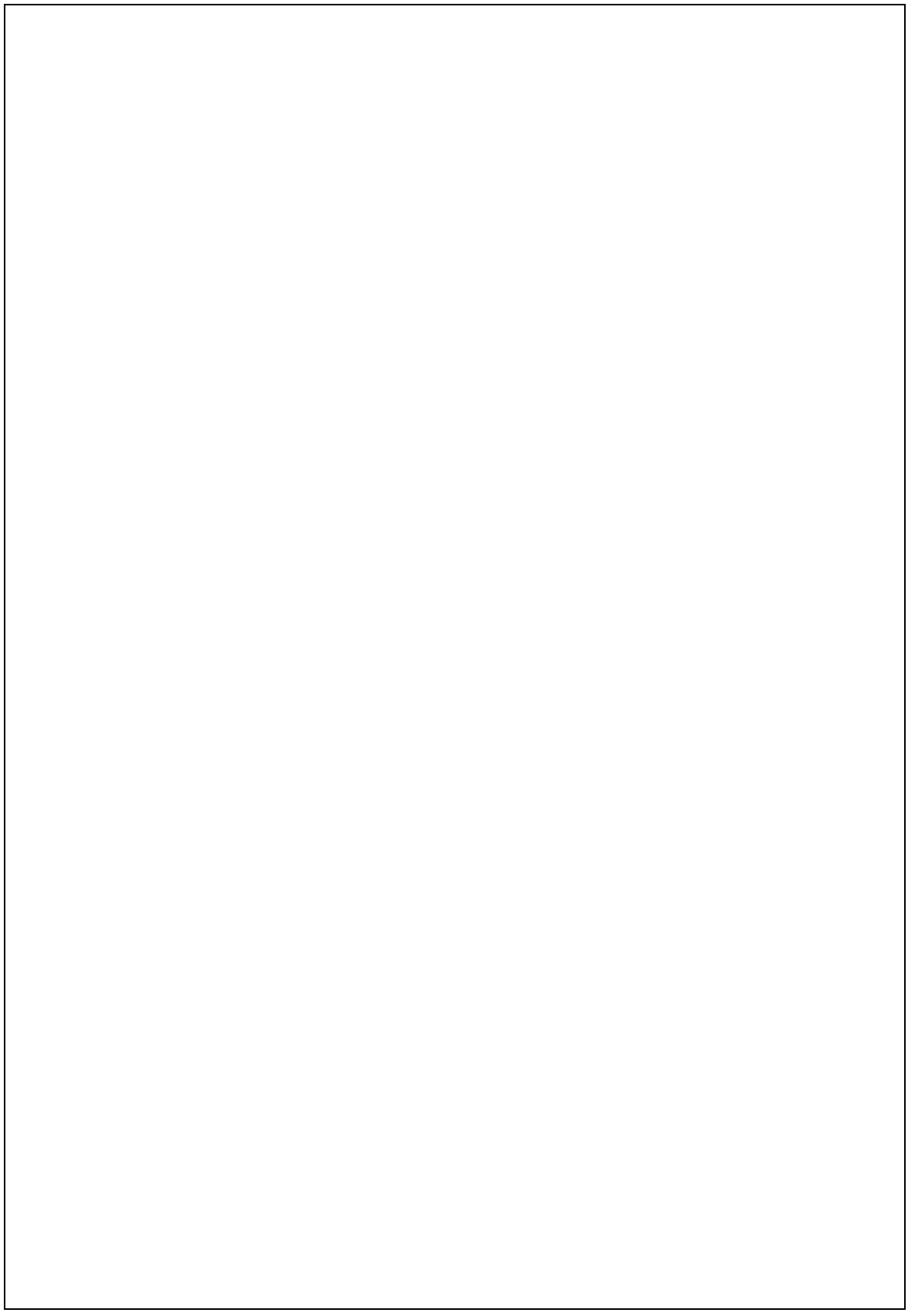
H. Resource / Equipment required

Sr. No.	Equipment / Software resources	Specifications
1	Computer system (Hardware Tool)	Computer having latest configuration with Windows or Unix OS
2	C compiler (Software Tool)	Open-source software

I. Safety and necessary precautions followed

- Shutdown computer system properly once the Lab hours are finished.
- Strictly follow the instructions given by laboratory in-charge.

J. Program Source Code (Implement insertion of node in sorted linked list)



K. Input – Output

L. References

I. https://www.programiz.com/dsa/linked-list
II. https://examradar.com/linked-list-concepts/
III. https://chat.openai.com/
IV. https://www.freecodecamp.org/news/data-structures-explained-with-examples-linked-list/
V. https://www.youtube.com/watch?v=dq3F3e9o2DM

M. Assessment Rubrics

Practical no. 13: Implement insertion of node at any position in linked list.				
Performance indicators for PROs	**Weighted Marks**	**Max Marks**	**Rubrics**	**Obtained Marks**
Correctness of program	**20%**	**5**	5 to 3 – Correct 2 – Partial correct 1 to 0 – Poor work	
Readability and documentation of the program/Quality of input and output displayed (messaging and formatting)	**20%**	**5**	5 to 4 – Excellent 3 to 2 – Good 1 to 0 – Poor	
Code efficiency	**10%**	**3**	3 – Excellent 2 – Good 1 to 0 – Poor	
Debugging ability	**20%**	**5**	5 to 4 – Excellent 3 to 2 – Good 1 to 0 – Poor	
Program execution/answer to questions	**30%**	**7**	7 to 5 – Proper execution 4 to 2 – Run with some errors 1 to 0 – Not run	
			Total Marks: (Out of 25)	

Signature with date

Date: _____

Practical No. 14: Implement counting number of nodes in singly linked list.

A Objective:

- The primary objective of learning about a singly linked list is to understand and be able to implement a fundamental data structure used in computer science.

B Expected Program Outcomes (POs):

1. **Basic and discipline specific knowledge:**
   Apply knowledge of basic mathematics, science and engineering fundamentals and engineering specialization to solve the engineering problems.
2. **Design/ development of solutions:**
   Design solutions for engineering well-defined technical problems and assist with the design of systems components or processes to meet specified needs.

C Expected Skills to be developed based on competency:

This practical is expected to develop the following skills for the industry-identified competency:

- ✓ Write code for the given problem.
- ✓ Debug program to fix errors.
- ✓ Follow coding guidelines.

D Expected Course Outcomes (COs):

Apply basic operations on the linked list data structure.

E Practical Outcome (PRo):

Implement counting number of nodes in singly linked list.

F Expected Affective Domain Outcomes (ADOs):

1. Follow safety practices.
2. Demonstrate working as a leader/a team member.
3. Maintain tools and equipment.
4. Follow ethical practices.

G. Prerequisite theory:

Algorithm: COUNT_SINGLY

This procedure counts the number of nodes in the linked list.

- FIRST is a pointer points to the first node of linked list.
- SAVE is a temporary pointer.
- count is a variable counting number of nodes.

```
Step1. [check for empty list]
        if (FIRST = NULL)
        then print ("LIST IS EMPTY")
        return( )
Step2. [initialize SAVE pointer and set count variable to 1]
        count = 1
        SAVE = FIRST
Step3. [process the linked list until end of the list and count nodes]
        repeat while ( LINK (SAVE) ≠ NULL )
                SAVE = LINK(SAVE)
                count = count + 1
step4.  [finished]
        return (count)
```

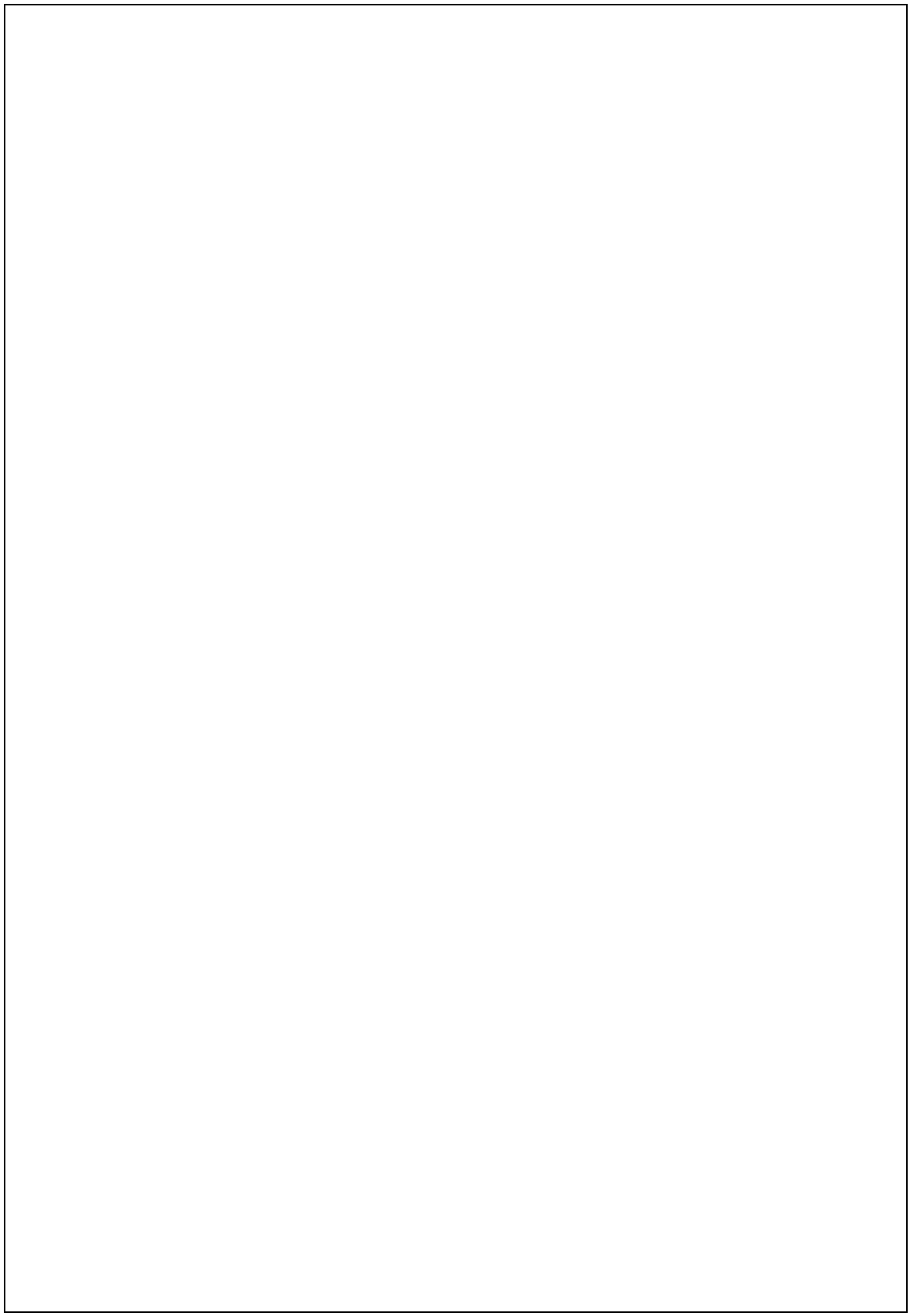
H. Resource / Equipment required

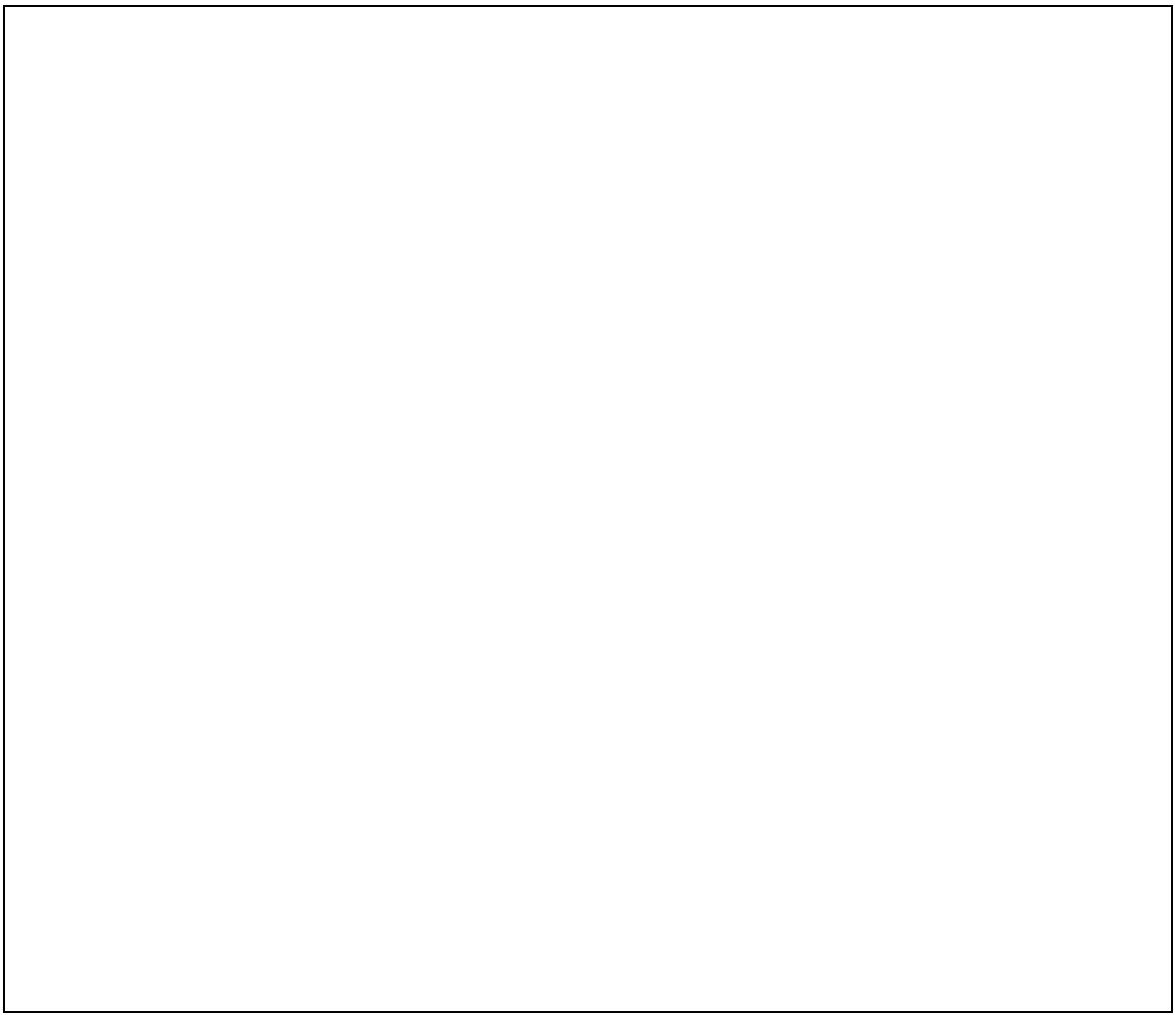
Sr. No.	Equipment / Software resources	Specifications
1	Computer system (Hardware Tool)	Computer having latest configuration with Windows or Unix OS
2	C compiler (Software Tool)	Open-source software

I. Safety and necessary precautions followed

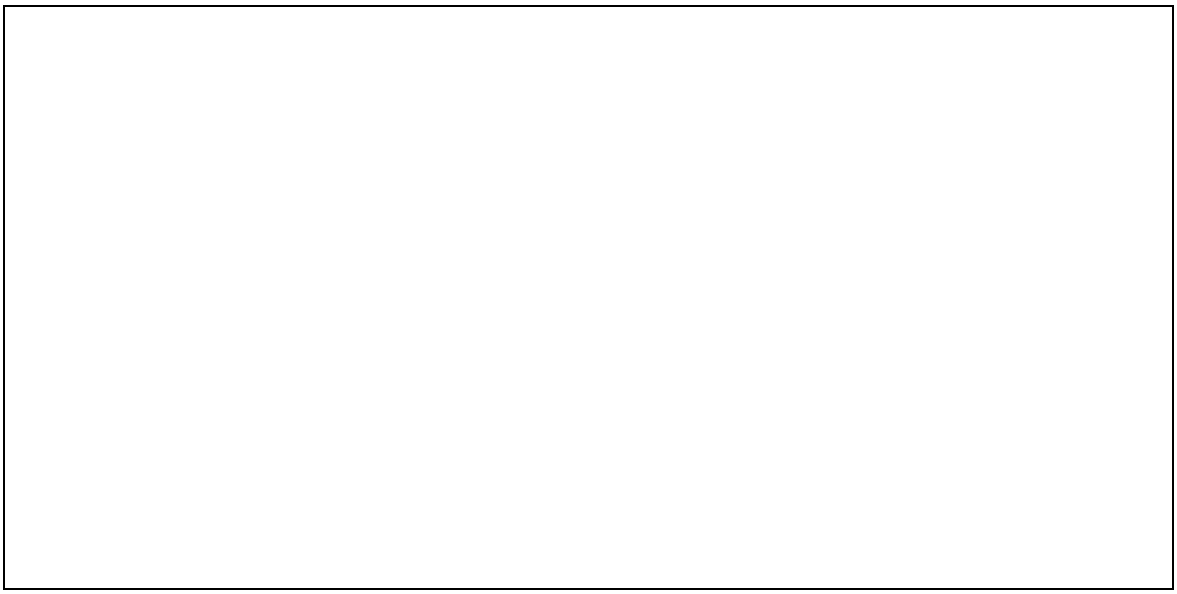
- Shutdown computer system properly once the Lab hours are finished.
- Strictly follow the instructions given by laboratory in-charge.

I. Program Source Code (Implement counting number of nodes in linked list)





K. Input – Output



L. References

I. https://www.programiz.com/dsa/linked-list
II. https://examradar.com/linked-list-concepts/
III. https://chat.openai.com/
IV. https://www.freecodecamp.org/news/data-structures-explained-with-examples-linked-list/
V. https://www.youtube.com/watch?v=ikrbSW9sVMU

M. Assessment Rubrics

Practical no. 14: Implement counting number of nodes in singly linked list.				
Performance indicators for PROs	**Weighted Marks**	**Max Marks**	**Rubrics**	**Obtained Marks**
Correctness of program	**20%**	**5**	5 to 3 – Correct 2 – Partial correct 1 to 0 – Poor work	
Readability and documentation of the program/Quality of input and output displayed (messaging and formatting)	**20%**	**5**	5 to 4 – Excellent 3 to 2 – Good 1 to 0 – Poor	
Code efficiency	**10%**	**3**	3 – Excellent 2 – Good 1 to 0 – Poor	
Debugging ability	**20%**	**5**	5 to 4 – Excellent 3 to 2 – Good 1 to 0 – Poor	
Program execution/answer to questions	**30%**	**7**	7 to 5 – Proper execution 4 to 2 – Run with some errors 1 to 0 – Not run	
			Total Marks: (Out of 25)	

Signature with date

Date: _____

Practical No. 15: Implement searching of a node in singly linked list.

A Objective:

- The primary objective of learning about a singly linked list is to understand and be able to implement a fundamental data structure used in computer science.

B Expected Program Outcomes (POs):

1. **Basic and discipline specific knowledge:**
   Apply knowledge of basic mathematics, science and engineering fundamentals and engineering specialization to solve the engineering problems.
2. **Design/ development of solutions:**
   Design solutions for engineering well-defined technical problems and assist with the design of systems components or processes to meet specified needs.

C Expected Skills to be developed based on competency:

This practical is expected to develop the following skills for the industry-identified competency:

- ✓ Write code for the given problem.
- ✓ Debug program to fix errors.
- ✓ Follow coding guidelines.

D Expected Course Outcomes (COs):

Apply basic operations on the linked list data structure.

E Practical Outcome (PRo):

Implement counting number of nodes in singly linked list.

F Expected Affective Domain Outcomes (ADOs):

1. Follow safety practices.
2. Demonstrate working as a leader/a team member.
3. Maintain tools and equipment.
4. Follow ethical practices.

G. Prerequisite theory:

Algorithm: SEARCH_SINGLY

This algorithm searches the given node (element) in the linked list.

- FIRST is a pointer to the first node of the linked list.
- X is the element which we want to search.
- SAVE is a temporary pointer variable. POS is the variable showing position of the searched node.

```
Step1. [check for empty list]
        if ( FIRST = NULL )
        then print (" LIST IS EMPTY AND UNDERFLOW")
        return( )
Step2.[initialize SAVE  pointer and set POS at first node]
        POS =1
        SAVE = FIRST
Step3. [Search target node]
        repeat while ( DATA(SAVE) ≠ X AND LINK(SAVE) ≠ NULL )
                SAVE = LINK(SAVE)
                POS = POS + 1
        if (DATA(SAVE) = X)
        then    print("Node is found at")
                print(POS)
                return( )
        else
                print("Node is not found in the list")
                return( )
```

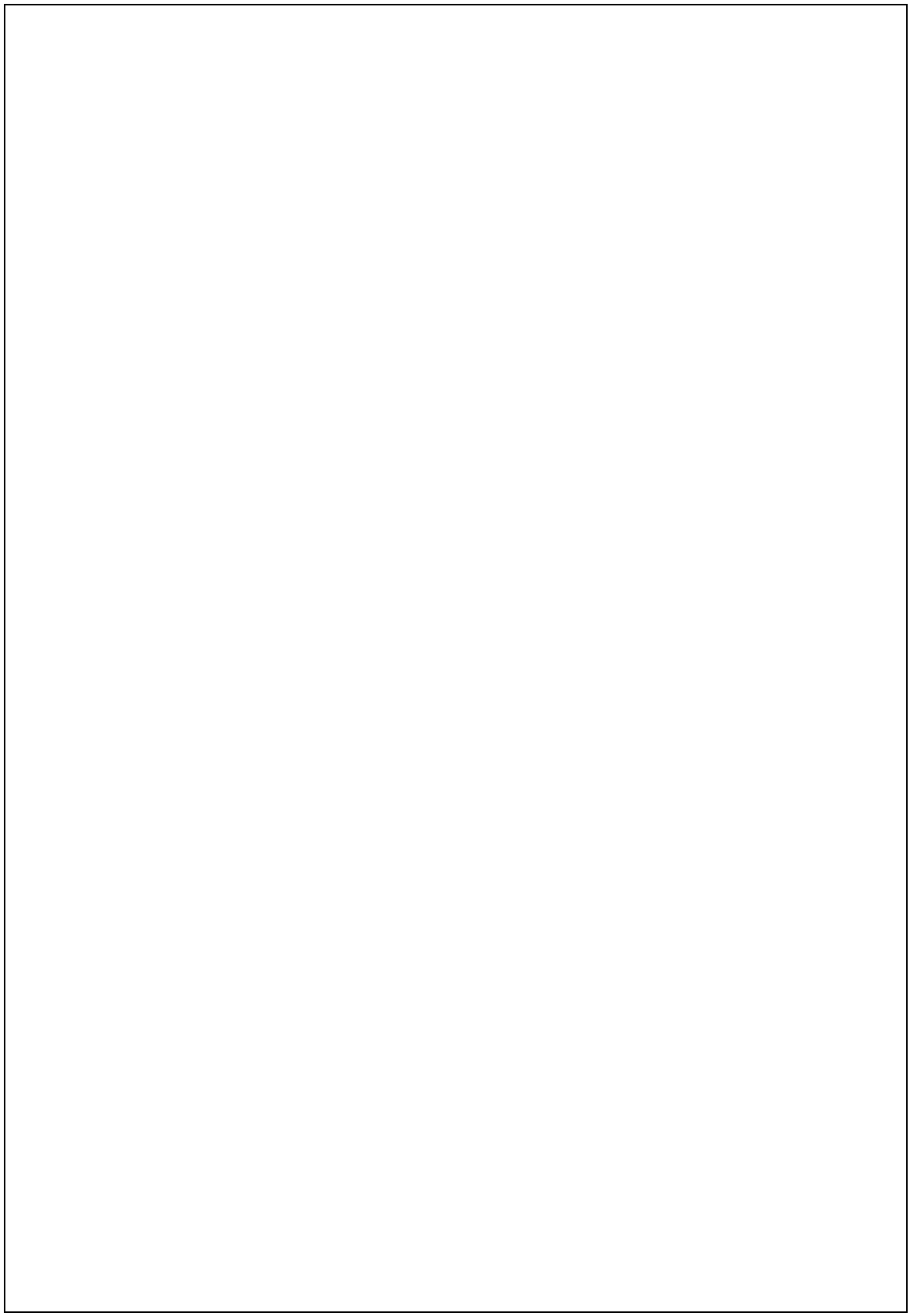
H. Resource / Equipment required

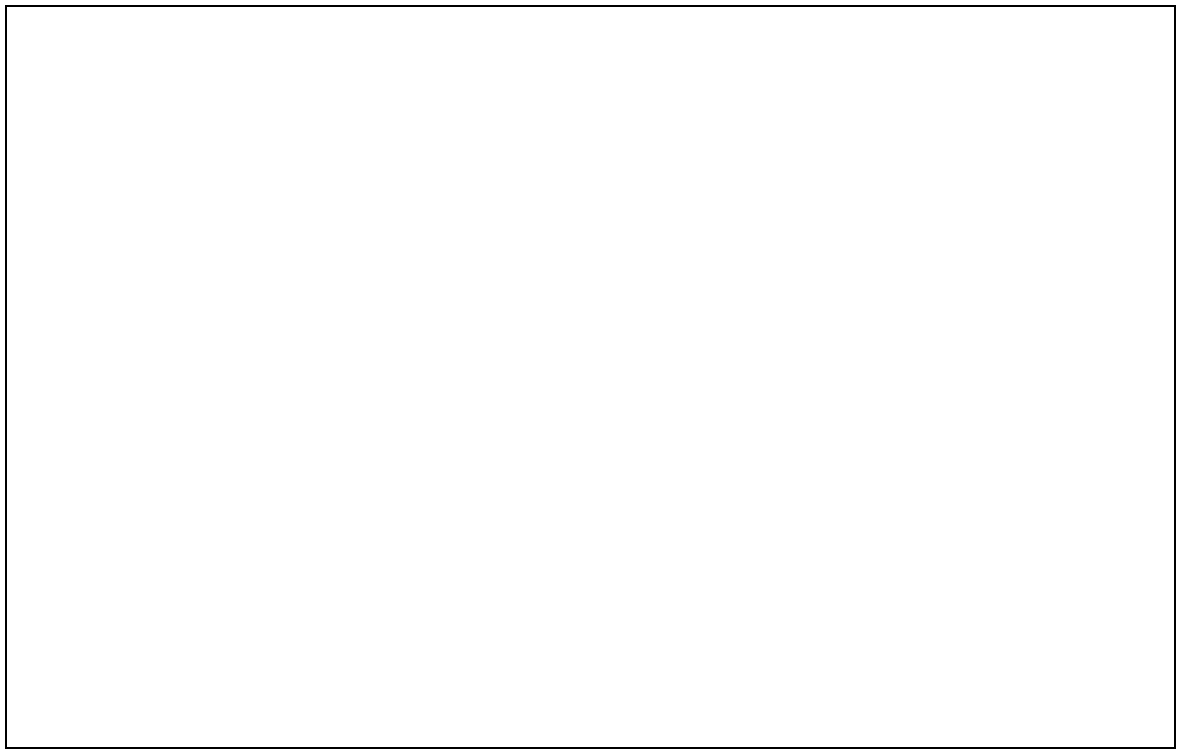
Sr. No.	Equipment / Software resources	Specifications
1	Computer system (Hardware Tool)	Computer having latest configuration with Windows or Unix OS
2	C compiler (Software Tool)	Open-source software

I. Safety and necessary precautions followed

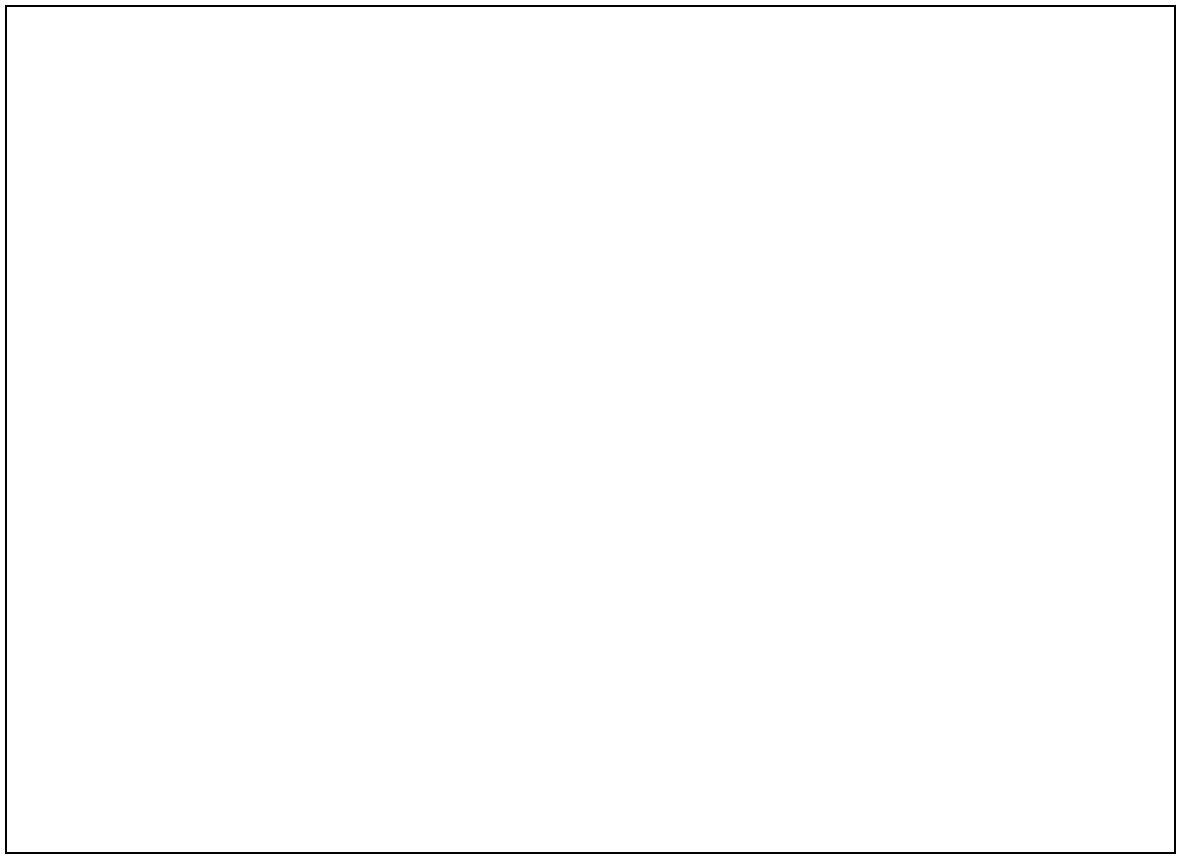
- Shutdown computer system properly once the Lab hours are finished.
- Strictly follow the instructions given by laboratory in-charge.

J. Program Source Code (Implement searching of a node in linked list)





K. Input – Output



L. References

I. https://www.programiz.com/dsa/linked-list
II. https://examradar.com/linked-list-concepts/
III. https://chat.openai.com/
IV. https://www.freecodecamp.org/news/data-structures-explained-with-examples-linked-list/
V. https://www.youtube.com/watch?v=tgFeh0mSDMM

M. Assessment Rubrics

Practical no. 15: Implement searching of a node in singly linked list.				
Performance indicators for PROs	**Weighted Marks**	**Max Marks**	**Rubrics**	**Obtained Marks**
Correctness of program	**20%**	**5**	5 to 3 – Correct 2 – Partial correct 1 to 0 – Poor work	
Readability and documentation of the program/Quality of input and output displayed (messaging and formatting)	**20%**	**5**	5 to 4 – Excellent 3 to 2 – Good 1 to 0 – Poor	
Code efficiency	**10%**	**3**	3 – Excellent 2 – Good 1 to 0 – Poor	
Debugging ability	**20%**	**5**	5 to 4 – Excellent 3 to 2 – Good 1 to 0 – Poor	
Program execution/answer to questions	**30%**	**7**	7 to 5 – Proper execution 4 to 2 – Run with some errors 1 to 0 – Not run	
			Total Marks: (Out of 25)	

Signature with date

Date: _____

Practical No. 16: Implement algorithm to delete a node in singly linked list.

A Objective:

- The primary objective of learning about a singly linked list is to understand and be able to implement a fundamental data structure used in computer science.

B Expected Program Outcomes (POs):

1. **Basic and discipline specific knowledge:**
   Apply knowledge of basic mathematics, science and engineering fundamentals and engineering specialization to solve the engineering problems.
2. **Design/ development of solutions:**
   Design solutions for engineering well-defined technical problems and assist with the design of systems components or processes to meet specified needs.

C Expected Skills to be developed based on competency:

This practical is expected to develop the following skills for the industry-identified competency:

- ✓ Write code for the given problem.
- ✓ Debug program to fix errors.
- ✓ Follow coding guidelines.

D Expected Course Outcomes (COs):

Apply basic operations on the linked list data structure.

E Practical Outcome (PRo):

Implement algorithm to delete a node in singly linked list.

F Expected Affective Domain Outcomes (ADOs):

1. Follow safety practices.
2. Demonstrate working as a leader/a team member.
3. Maintain tools and equipment.
4. Follow ethical practices.

G. Prerequisite theory:

Algorithm: DELETE_FIRST_SINGLY

This procedure deletes a node from the beginning of a singly linked list.

- FIRST is a pointer to the first node of linked list.
- SAVE is a temporary pointer variable which is to be deleted.

```
Step1. [checking for empty list]
        if ( FIRST = NULL )
        then    print (" LIST IS EMPTY AND UNDERFLOW")
                return( )
Step2. [initialize SAVE pointer]
        SAVE = FIRST
Step3. [delete first node and free the space of deleted node]
        if ( LINK (FIRST) = NULL )          // when only one node is there in list
        then    FIRST = NULL
                free (FIRST)
        else    FIRST = LINK (FIRST)        //in case of more than one node
                free (SAVE)                 //freeing the memory of deleted node
                return(FIRST)
```

Algorithm: DELETE_END_SINGLY

This procedure deletes a node from the end of a singly linked list.

- FIRST is a pointer to the first node of the linked.
- SAVE and PRED are temporary pointer variables. PRED keeping track of predecessor node. SAVE pointing to node to be deleted.

```
Step1. [checking for empty list]
        if ( FIRST = NULL )
        then    print (" LIST IS EMPTY AND UNDERFLOW")
                return( )
Step2. [initialize SAVE pointer]
        SAVE = FIRST
Step3. [search target node]
```

```
        if (LINK (FIRST) = NULL)              // if only one node in list
        then    FIRST = NULL
                free(FIRST)
        else    repeat while( LINK(SAVE) ≠ NULL )
                PRED = SAVE
                SAVE = LINK(SAVE)
Step4. [when find the last node, delete it and free the space of deleted node]
        LINK(PRED) = NULL           //remove the last node
        free(SAVE)                          // freeing the memory of deleted node
        return(FIRST)
```

H. Resource / Equipment required

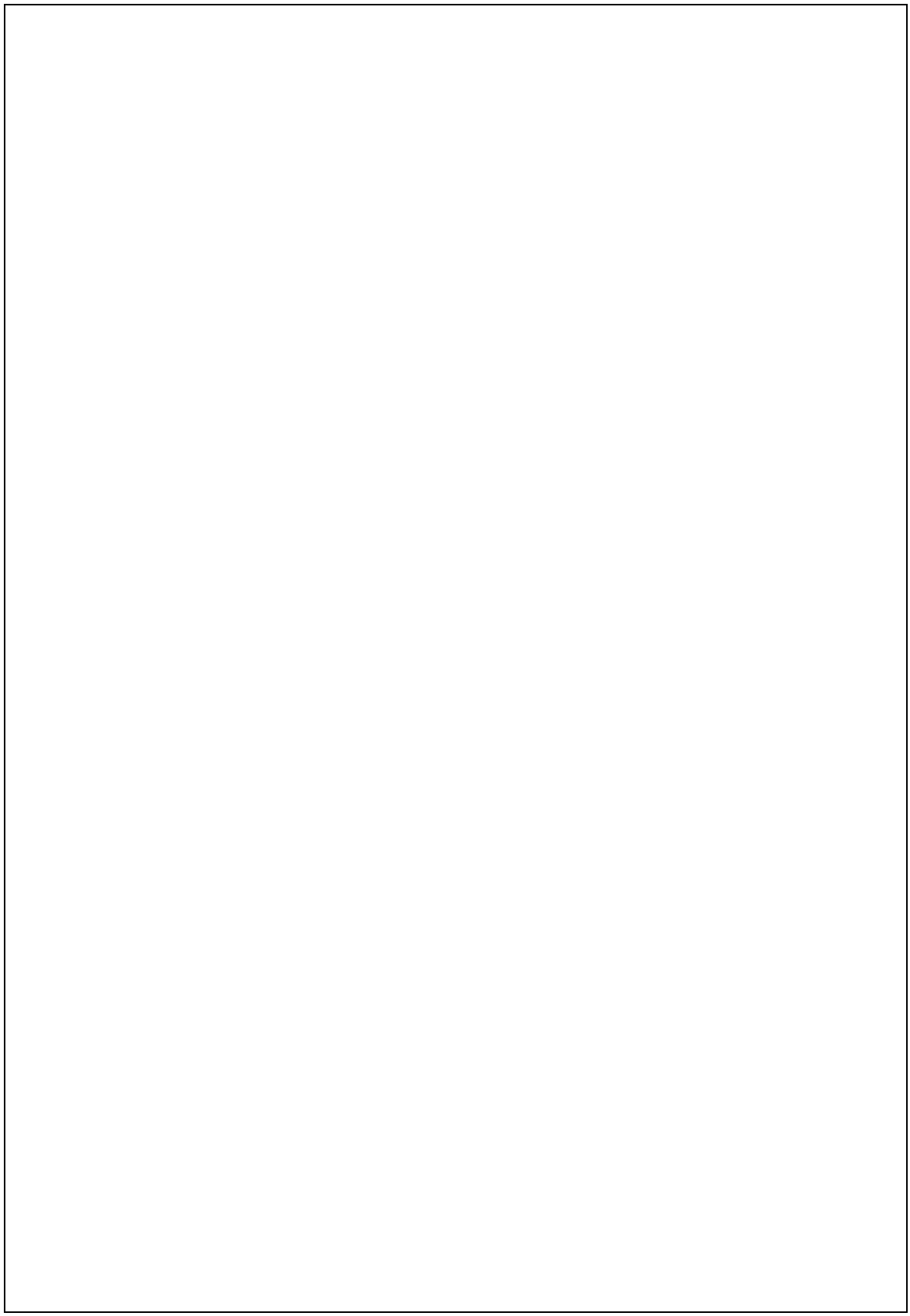
Sr. No.	Equipment / Software resources	Specifications
1	Computer system (Hardware Tool)	Computer having latest configuration with Windows or Unix OS
2	C compiler (Software Tool)	Open-source software

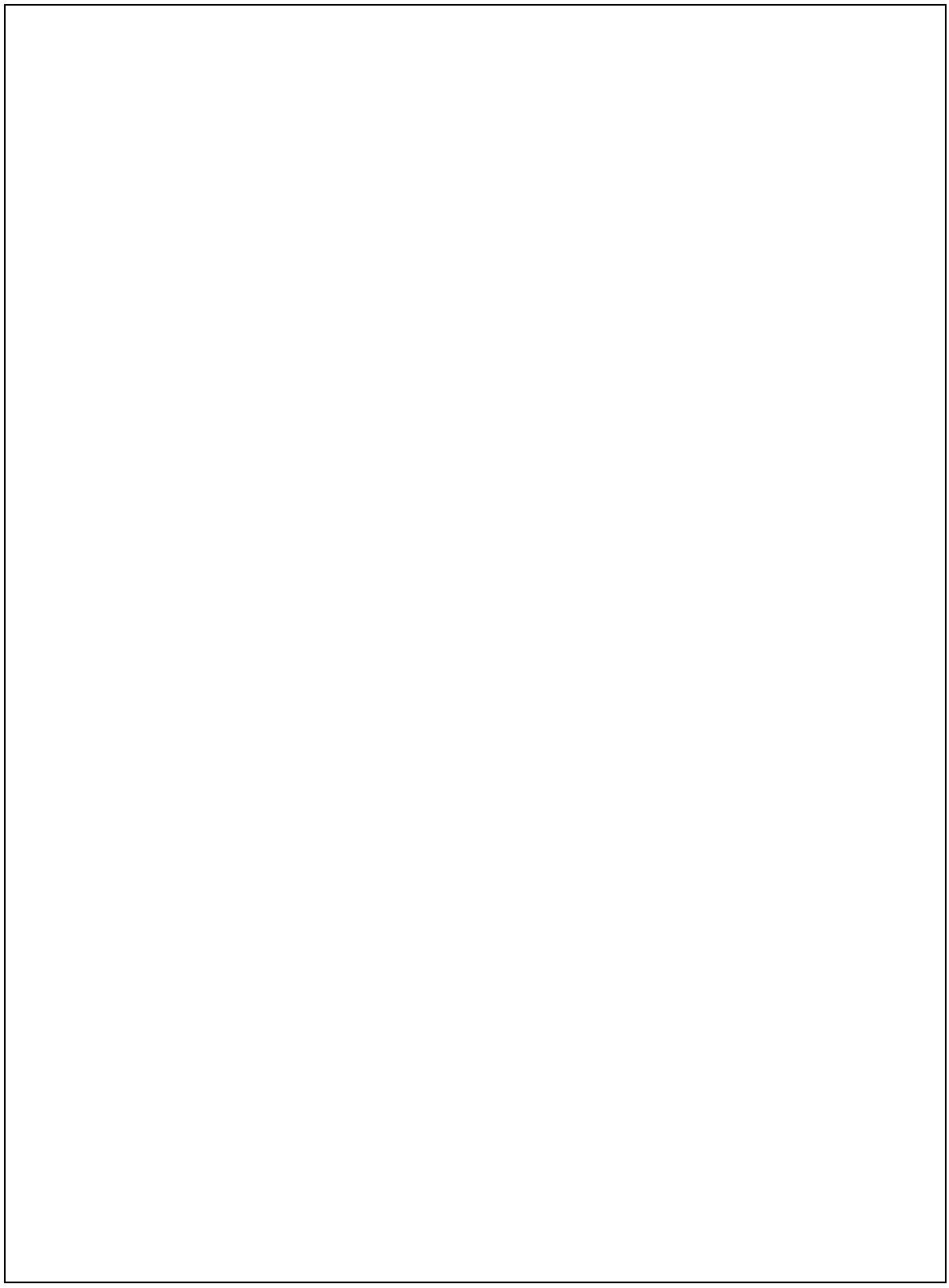
I. Safety and necessary precautions followed

- Shutdown computer system properly once the Lab hours are finished.
- Strictly follow the instructions given by laboratory in-charge.

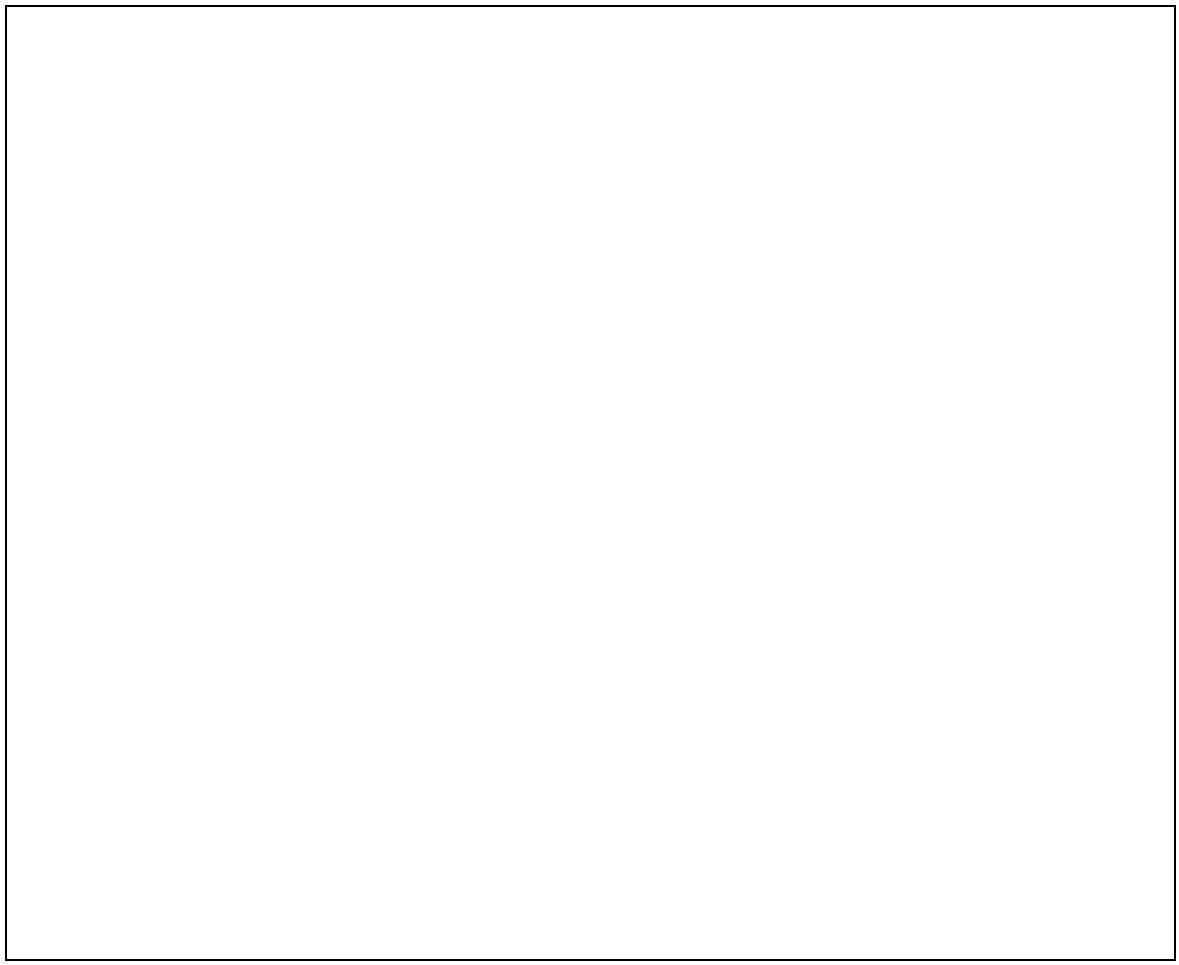
J. Program Source Code

1) Implement algorithm to delete a node from beginning in singly linked list.

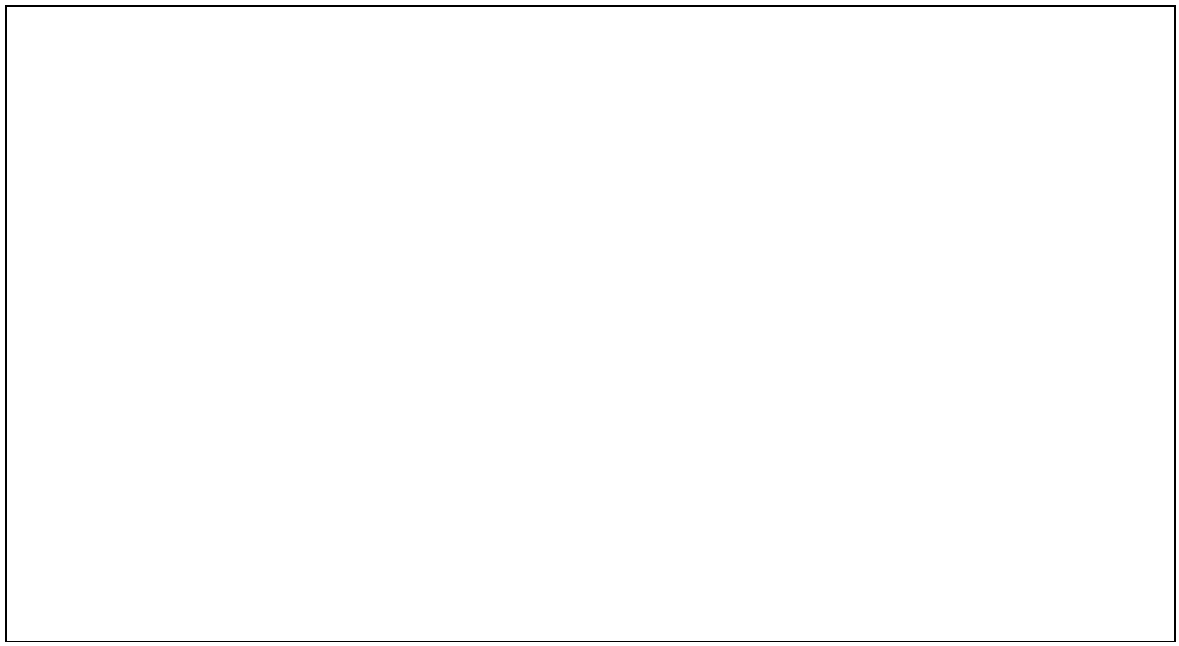


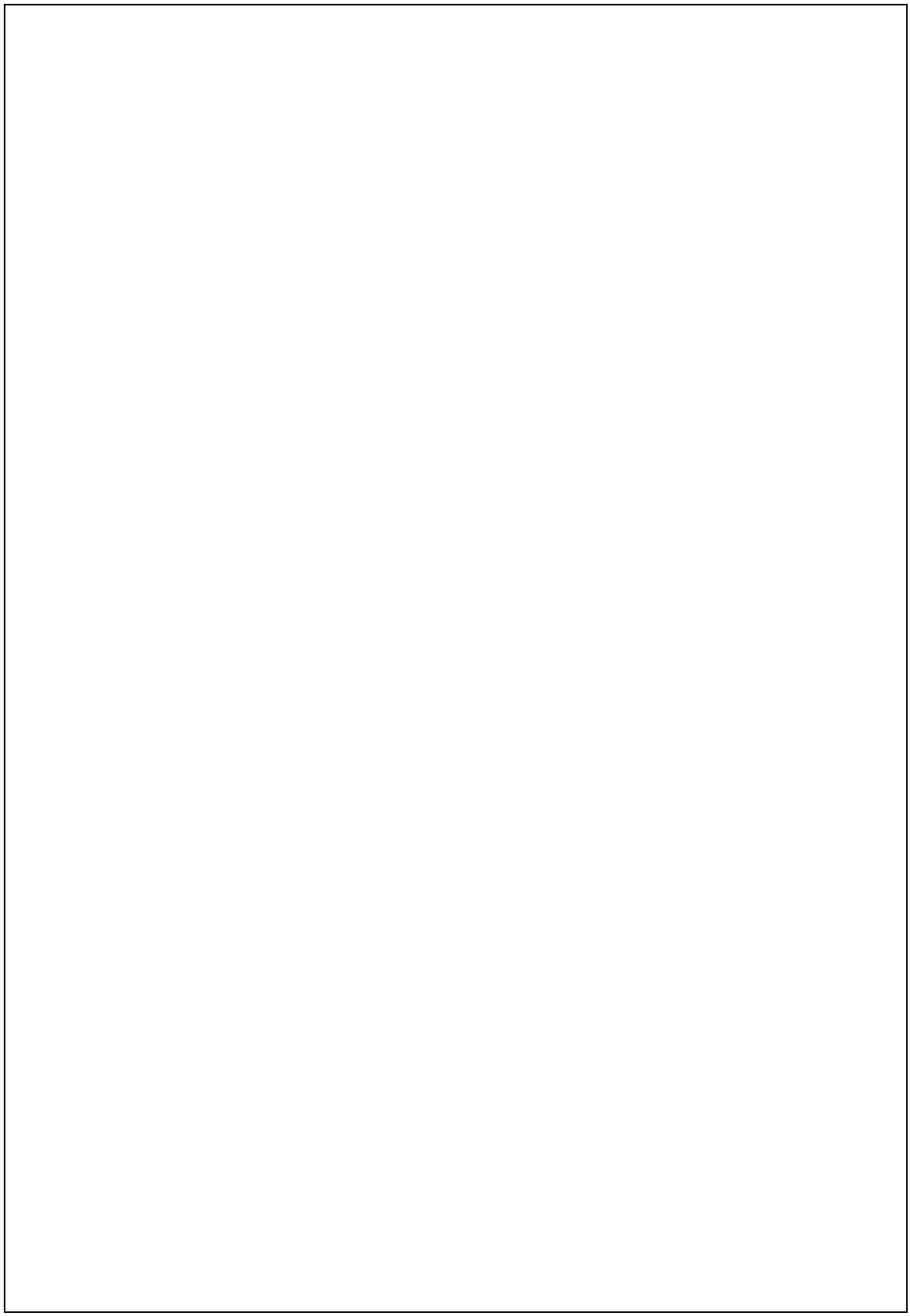


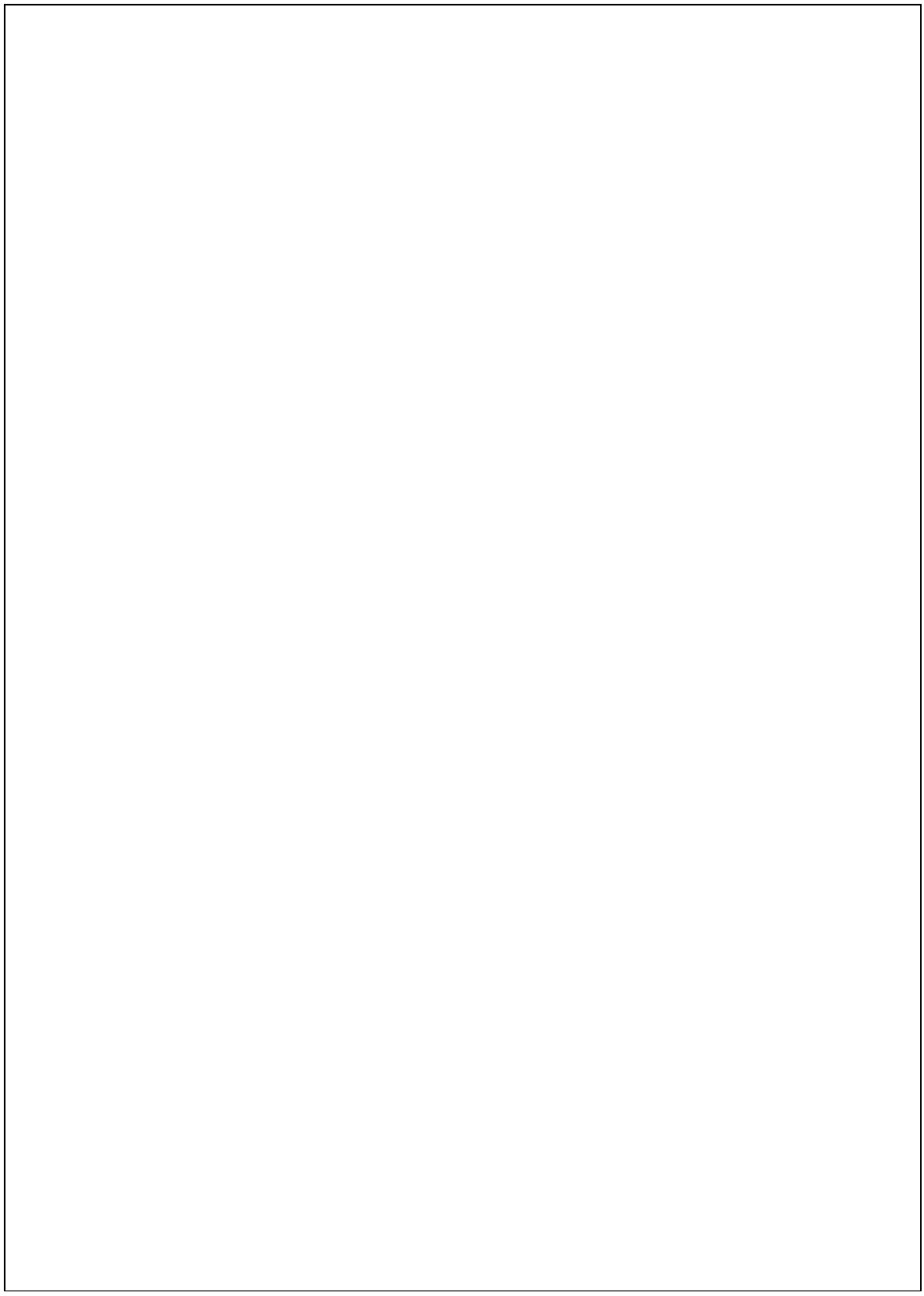
K. (1) Input – Output



2) Implement algorithm to delete a node at the end in singly linked list.







K. (2) Input – Output

L. References

I. https://www.programiz.com/dsa/linked-list
II. https://examradar.com/linked-list-concepts/
III. https://chat.openai.com/
IV. https://www.freecodecamp.org/news/data-structures-explained-with-examples-linked-list/
V. https://www.youtube.com/watch?v=f1r_jxCyOl0

M. Assessment Rubrics

Practical no. 16: Implement algorithm to delete a node in singly linked list.				
Performance indicators for PROs	**Weighted Marks**	**Max Marks**	**Rubrics**	**Obtained Marks**
Correctness of program	**20%**	**5**	5 to 3 – Correct 2 – Partial correct 1 to 0 – Poor work	
Readability and documentation of the program/Quality of input and output displayed (messaging and formatting)	**20%**	**5**	5 to 4 – Excellent 3 to 2 – Good 1 to 0 – Poor	
Code efficiency	**10%**	**3**	3 – Excellent 2 – Good 1 to 0 – Poor	
Debugging ability	**20%**	**5**	5 to 4 – Excellent 3 to 2 – Good 1 to 0 – Poor	
Program execution/answer to questions	**30%**	**7**	7 to 5 – Proper execution 4 to 2 – Run with some errors 1 to 0 – Not run	
			Total Marks: (Out of 25)	

Signature with date

Date: _____

Practical No. 17: Implement construction of Binary Search Tree (BST). Implement pre-order, in-order, post-order traversal methods in binary search tree.

A Objective:

- Main objective of learning binary search tree is to optimize insertion, deletion and searching operation in tree data structure also trees serve as a foundation for other advanced data structures and algorithms.

B Expected Program Outcomes (POs):

1. **Basic and discipline specific knowledge:**
   Apply knowledge of basic mathematics, science and engineering fundamentals and engineering specialization to solve the engineering problems.
2. **Problem analysis:**
   Identify and analyse well-defined engineering problems using codified standard methods.
3. **Design/ development of solutions:**
   Design solutions for engineering well-defined technical problems and assist with the design of systems components or processes to meet specified needs.

C Expected Skills to be developed based on competency:

This practical is expected to develop the following skills for the industry-identified competency:

- ✓ Write algorithm for construction of BST.
- ✓ Write code for the given problem.
- ✓ Gain knowledge about traversal methods of BST.
- ✓ Follow coding guidelines.

D Expected Course Outcomes (COs):

Illustrate algorithms to insert, delete and searching a node in tree.

E Practical Outcome (PRo):

Sd Implement construction of Binary Search Tree (BST). Implement pre-order, in-order, post-order traversal methods in binary search tree.

F Expected Affective Domain Outcomes (ADOs):

1. Follow safety practices.
2. Practice good housekeeping.
3. Demonstrate working as a leader/a team member.
4. Maintain tools and equipment.
5. Follow ethical practices.

G. Prerequisite theory:

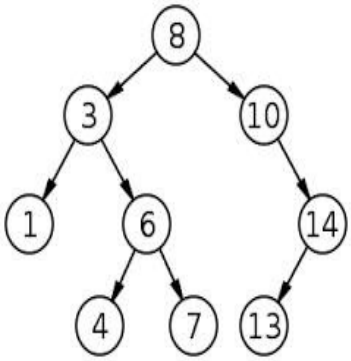
A tree is called Binary Search tree (BST) if each and every node can have at most two branches. And it should contain following characteristics.

- ✓ All the nodes to the left of the any node have values less than the value of that node and
- ✓ All the nodes to the right of any node have values greater than the value of that node.

Operations on Binary Search Tree

- Insertion
- Deletion
- Searching
- Traversal

A simple example of BST shown in below figure.



Defining a node in BST

```
struct node {
  int data;
  struct node *leftChild;
  struct node *rightChild;
};
```

Search operation in BST

1. START

2. Check whether the tree is empty or not
3. If the tree is empty, search is not possible
4. Otherwise, first search the root of the tree.
5. If the key does not match with the value in the root, search its subtrees.
6. If the value of the key is less than the root value, search the left subtree
7. If the value of the key is greater than the root value, search the right subtree.
8. If the key is not found in the tree, return unsuccessful search.
9. END

Insert operation in BST

1. START
2. If the tree is empty, insert the first element as the root node of the tree. The following elements are added as the leaf nodes.
3. If an element is less than the root value, it is added into the left subtree as a leaf node.
4. If an element is greater than the root value, it is added into the right subtree as a leaf node.
5. The final leaf nodes of the tree point to NULL values as their child nodes.
6. END

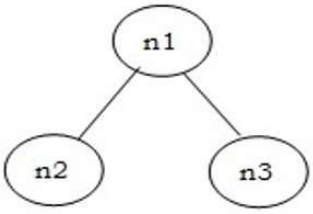
Tree traversal

As we have seen that traversal is a method of processing every node in the tree exactly once in a systematic manner. There are three ways for binary search tree traversing.

Pre – Order Traversal (Root – Left – Right)

In – Order Traversal (Left – Root – Right)

Post – Order Traversal (Left – Right – Root)



Pre-Order: n1 n2 n3
In-Order: n2 n1 n3
Post-Order: n2 n3 n1

Algorithm: PREORDER

- This function traverses the tree in pre-order
- T is a pointer which points to the root node of the tree

- LPTR and RPTR denotes to the left pointer and right pointer of the particular node.

```
Step1. [process the root node]
        if (T != NULL)
        then    write (DATA (T))
        else    write ("EMPTY TREE")
                return ( )
step2. [process the left sub tree]
        if ( LPTR (T) != NULL )
        then    call PREORDER (LPTR(T))
step3.  [process the right sub tree]
        if ( RPTR (T) != NULL )
        then    call PREORDER (RPTR(T))
step4.  [finished]
        return( )
```

```
1. START
2. Traverse the root node first.
3. Then traverse the left subtree, recursively
4. Later, traverse the right subtree, recursively.
5. END
```

Algorithm: INORDER

- This function traverses the tree in in-order.
- T is a pointer which points to the root node of the tree.
- LPTR and RPTR denotes to the left pointer and right pointer of the particular node.

```
Step1. [check for empty tree]
        if (T = NULL)
        then    write ("EMPTY TREE")
                return( )
step2. [process the left sub tree]
        if ( LPTR (T) != NULL )
        then    call INORDER (LPTR(T))
step3. [process the root node]
        write (DATA (T))
step4. [process the right sub tree]
        if ( RPTR (T) != NULL )
        then    call INORDER (RPTR(T))
```

```
1. START
2. Traverse the left subtree, recursively
3. Then, traverse the root node
4. Traverse the right subtree, recursively.
5. END
```

```
step5. [finished]
        return( )
```

Algorithm: POSTORDER

- This function traverses the tree in post-order
- T is a pointer which points to the root node of the tree
- LPTR and RPTR denotes to the left pointer and right pointer of the particular node.

```
Step1. [check for empty tree]
        if ( T = NULL )
        then    write ("EMPTY TREE")
                return( )
Step2. [process the left sub tree]
        if LPTR (T) != NULL
        then    call POSTORDER (LPTR(T))
step3. [process the right sub tree]
        if ( RPTR (T) != NULL )
        then    call POSTORDER (RPTR(T))
step4. [process the root node]
        write (DATA (T))
step5. [finished]
        return( )
```

```
1. START
2. Traverse the left subtree, recursively
3. Traverse the right subtree, recursively.
4. Then, traverse the root node
5. END
```

H. Resource / Equipment required

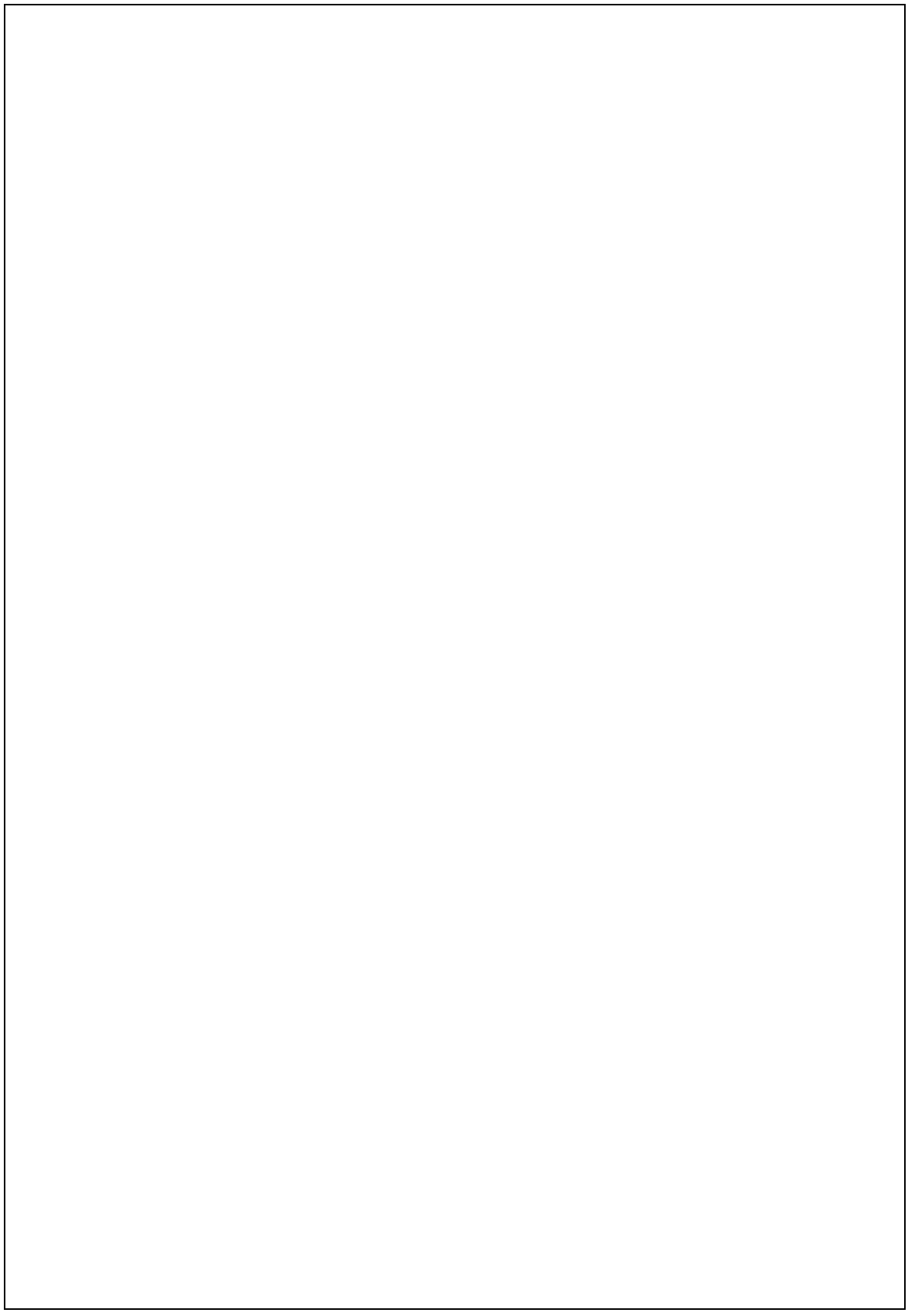
Sr. No.	Equipment / Software resources	Specifications
1	Computer system (Hardware Tool)	Computer having latest configuration with Windows or Unix OS
2	C compiler (Software Tool)	Open-source software

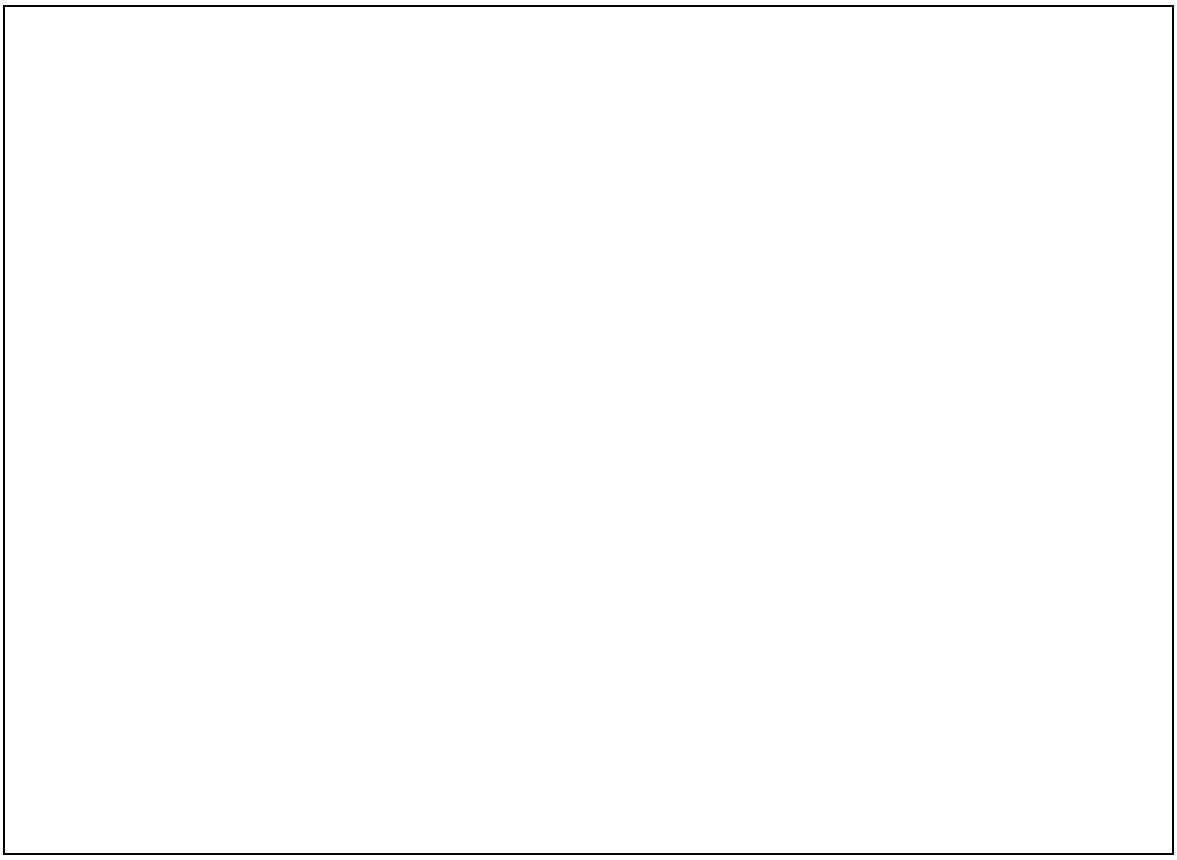
I. Safety and necessary precautions followed

- Shutdown computer system properly once the Lab hours are finished.
- Strictly follow the instructions given by laboratory in-charge.

J. Program Source Code

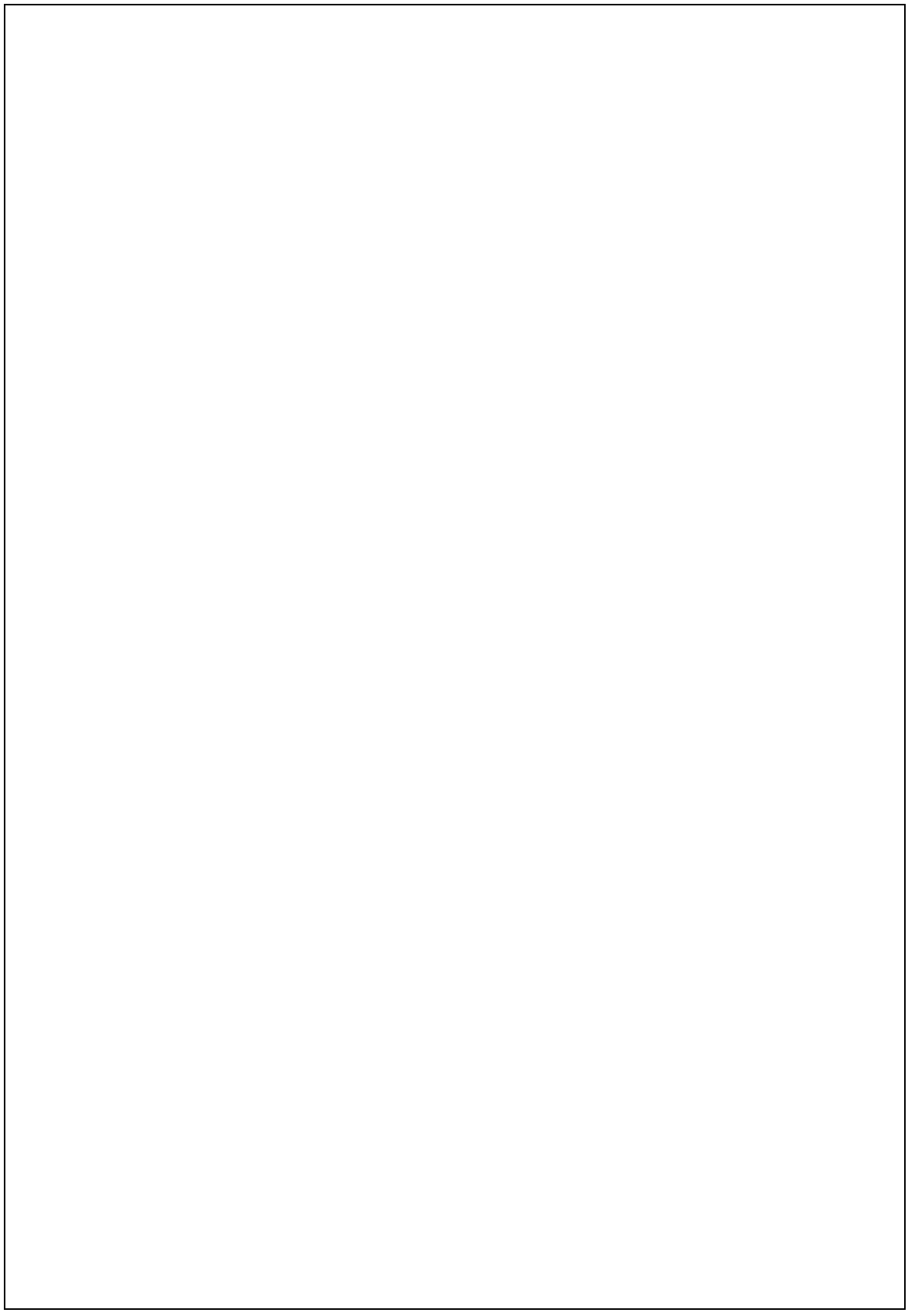
1) Implement construction of BST.

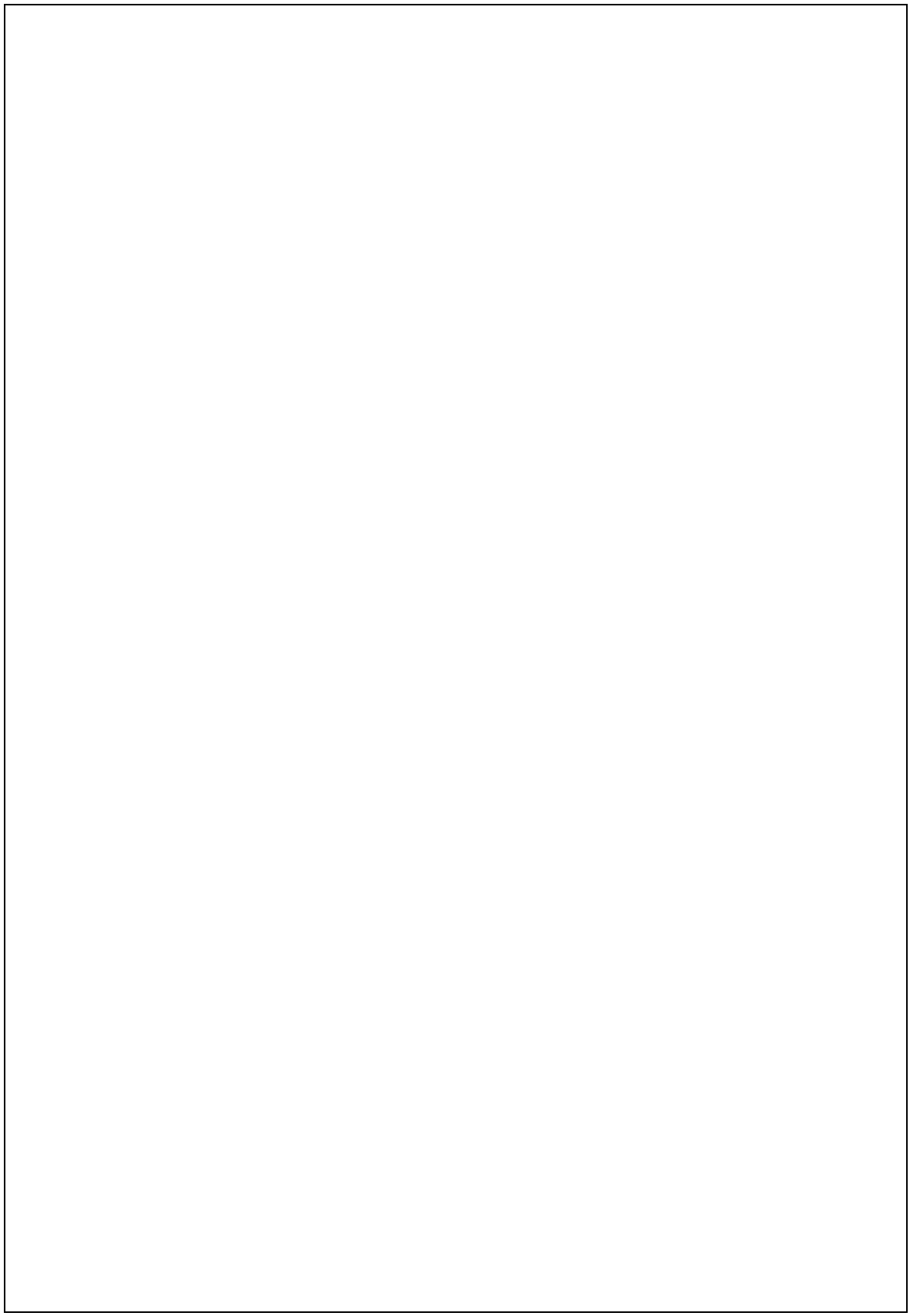


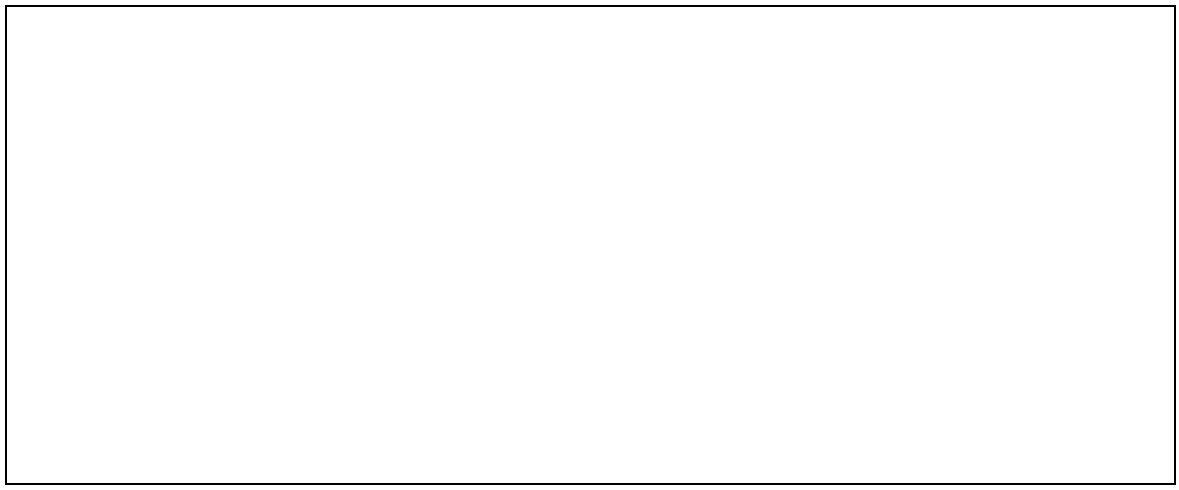


K (1). Input – Output

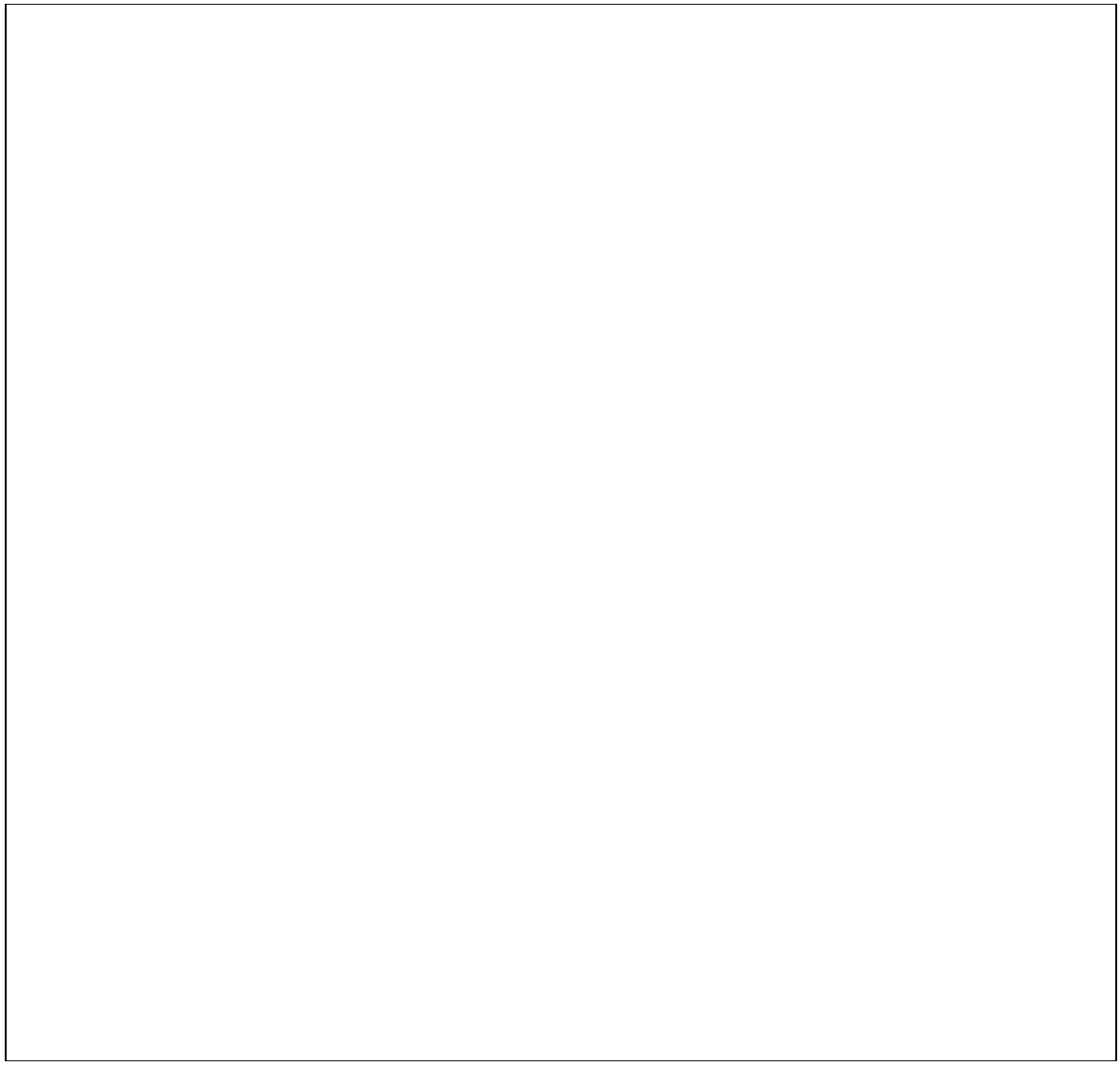
2) Write a source code to implement pre-order, in-order and post-order traversal in BST.







K (2). Input – Output

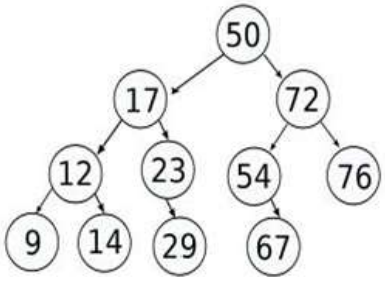


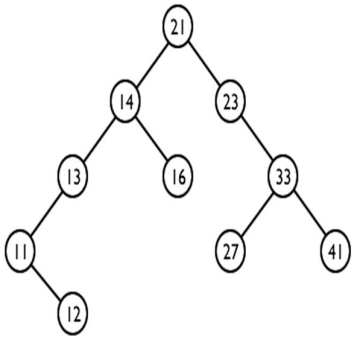
L. Practical related Quiz

1. Define following terms:

- Tree	- Graph	- BST	- Forest
- Root node	- Indegree	- Terminal node	- Sibling
- Path	- Height of tree	- Strict binary tree	- Complete binary tree
- M-ary tree	- Depth of tree	- Outdegree	- Directed tree

2. Find pre-order, in-order and post-order traversal for the following trees.





3. Fill in the blanks for the following.
   I. The topmost node in a tree is called the _____.
   II. A node that has no children is called a _____.
   III. A tree where each node has at most two children is called a _____.
   IV. A _____ is a type of tree traversal that visits all the nodes of a tree in a specific order → root, left subtree, right subtree.
   V. In a binary tree, the maximum number of nodes at level 'h' is _____.
   VI. The _____ of a node in a tree refers to the sequence of nodes from the root to that particular node.

4. State TRUE (T) / FALSE (F) for the following.

I. A graph is acyclic tree. ☐

II. A binary tree can have multiple nodes with the same value. ☐

III. A binary tree can be both complete and full at the same time. ☐

IV. In a binary tree, the depth of a node is always equal to its level. ☐

V. In a BST, the minimum value is always found in the leftmost node. ☐

M. References

I. https://www.geeksforgeeks.org/introduction-to-tree-data-structure-and-algorithm-tutorials/
II. https://www.tutorialspoint.com/data_structures_algorithms/tree_data_structure.htm
III. https://chat.openai.com/
IV. https://www.simplilearn.com/tutorials/data-structure-tutorial/trees-in-data-structure
V. https://www.youtube.com/watch?v=XRcC7bAtL3c

N. Assessment Rubrics

Practical no. 17: Implement construction of Binary Search Tree (BST). Implement pre-order, in-order, post-order traversal methods in binary search tree.				
Performance indicators for PROs	**Weighted Marks**	**Max Marks**	**Rubrics**	**Obtained Marks**
Correctness of program	**20%**	**5**	5 to 3 – Correct 2 – Partial correct 1 to 0 – Poor work	
Readability and documentation of the program/Quality of input and output displayed (messaging and formatting)	**20%**	**5**	5 to 4 – Excellent 3 to 2 – Good 1 to 0 – Poor	
Code efficiency	**10%**	**3**	3 – Excellent 2 – Good 1 to 0 – Poor	
Debugging ability	**20%**	**5**	5 to 4 – Excellent 3 to 2 – Good 1 to 0 – Poor	
Program execution/answer to questions	**30%**	**7**	7 to 5 – Proper execution 4 to 2 – Run with some errors 1 to 0 – Not run	
			Total Marks: (Out of 25)	

Signature with date

Date: _____

Practical No. 18: Implement searching algorithm in Binary Search Tree (BST).

Note: ***consider BST constructed in Practical No. 17***

A Objective:

- Main objective of learning binary search tree is to optimize insertion, deletion and searching operation in tree data structure also trees serve as a foundation for other advanced data structures and algorithms.
- Another objective is to search the element efficiently from the given data.

B Expected Program Outcomes (POs):

1. **Basic and discipline specific knowledge:**
   Apply knowledge of basic mathematics, science and engineering fundamentals and engineering specialization to solve the engineering problems.
2. **Problem analysis:**
   Identify and analyse well-defined engineering problems using codified standard methods.
3. **Design/ development of solutions:**
   Design solutions for engineering well-defined technical problems and assist with the design of systems components or processes to meet specified needs.

C Expected Skills to be developed based on competency:

This practical is expected to develop the following skills for the industry-identified competency:

- ✓ Write algorithm for search operation of BST.
- ✓ Write code for the given problem.
- ✓ Gain knowledge about search the node element in BST.
- ✓ Follow coding guidelines.

D Expected Course Outcomes (COs):

Illustrate algorithms to insert, delete and searching a node in tree.

E Practical Outcome (PRo):

Implement searching algorithm in Binary Search Tree (BST).

F Expected Affective Domain Outcomes (ADOs):

1. Follow safety practices.

2. Practice good housekeeping.
3. Demonstrate working as a leader/a team member.
4. Maintain tools and equipment.
5. Follow ethical practices.

G. Prerequisite theory:

A tree is called Binary Search tree (BST) if each and every node can have at most two branches. And it should contain following characteristics.

Algorithm: BST_SEARCH

This algorithm will search the KEY node in the BST.

- T is a pointer which points to the root node.
- LPTR and RPTR are left and right pointer.

```
Step1. [check for empty tree]
        if (T = NULL)
        then    write ("Tree is empty")
                return (0)
Step2. [Compare the KEY with root node]
        if (KEY = DATA(T))
        then    return (T)
Step3. [search in left or right sub tree]
        if (KEY < DATA(T))
        then    call BST_SEARCH(LPTR(T), KEY)
        else    call BST_SEARCH(RPRT(T), KEY)
Step4. [finished]
        return( )
```

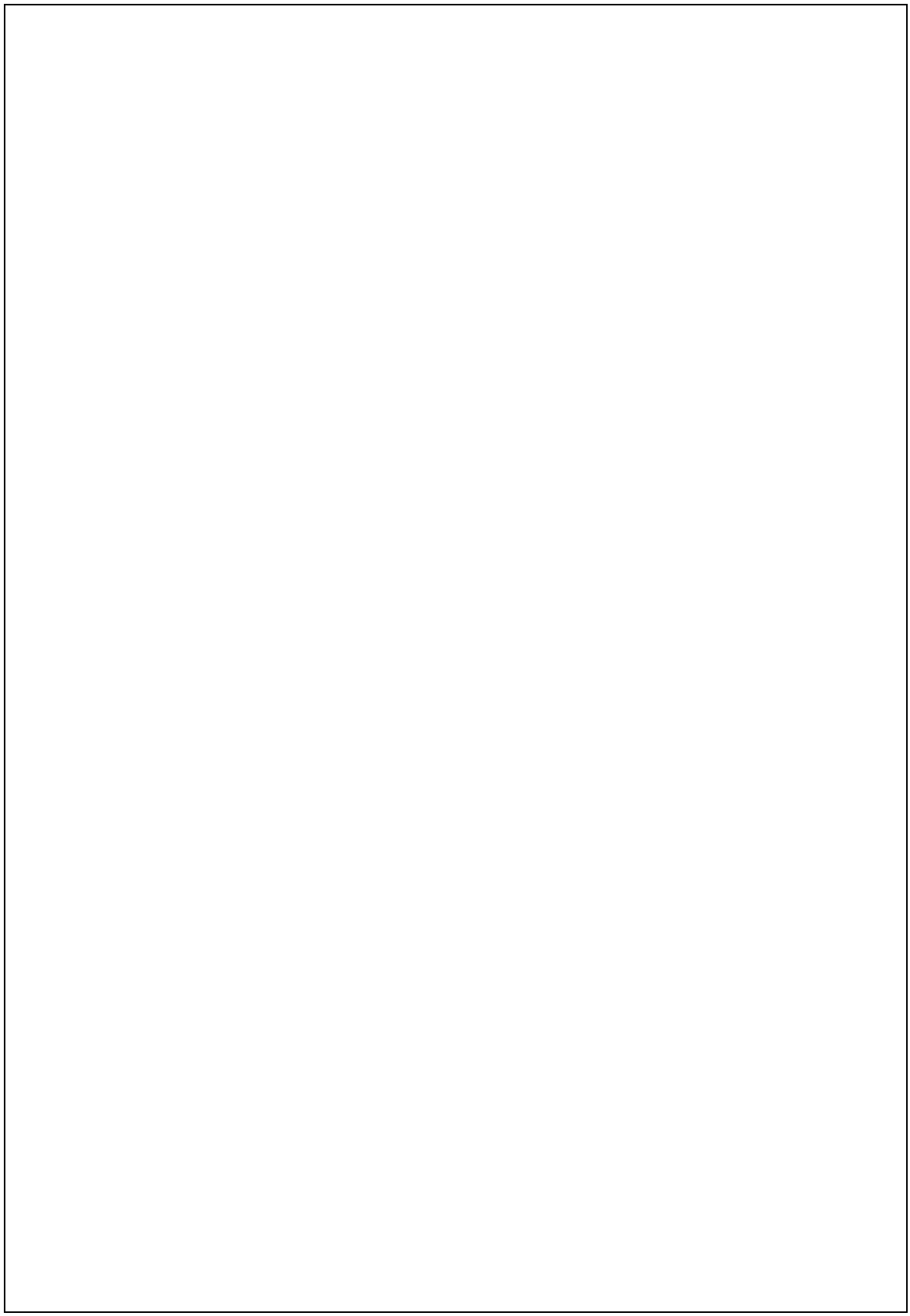
H. Resource / Equipment required

Sr. No.	Equipment / Software resources	Specifications
1	Computer system (Hardware Tool)	Computer having latest configuration with Windows or Unix OS
2	C compiler (Software Tool)	Open-source software

I. Safety and necessary precautions followed

- Shutdown computer system properly once the Lab hours are finished.
- Strictly follow the instructions given by laboratory in-charge.

J. Program Source Code (Implement search operation on BST).



K. Input – Output

L. References

I. https://www.geeksforgeeks.org/introduction-to-tree-data-structure-and-algorithm-tutorials/
II. https://www.tutorialspoint.com/data_structures_algorithms/tree_data_structure.htm
III. https://chat.openai.com/
IV. https://www.youtube.com/watch?v=OKXI2woGoeg

M. Assessment Rubrics

Practical no. 18: Implement searching algorithm in BST.				
Performance indicators for PROs	**Weighted Marks**	**Max Marks**	**Rubrics**	**Obtained Marks**
Correctness of program	**20%**	**5**	5 to 3 – Correct 2 – Partial correct 1 to 0 – Poor work	
Readability and documentation of the program/Quality of input and output displayed (messaging and formatting)	**20%**	**5**	5 to 4 – Excellent 3 to 2 – Good 1 to 0 – Poor	
Code efficiency	**10%**	**3**	3 – Excellent 2 – Good 1 to 0 – Poor	
Debugging ability	**20%**	**5**	5 to 4 – Excellent 3 to 2 – Good 1 to 0 – Poor	
Program execution/answer to questions	**30%**	**7**	7 to 5 – Proper execution 4 to 2 – Run with some errors 1 to 0 – Not run	
			Total Marks: (Out of 25)	

Signature with date

Date: _____

Practical No. 19: Implement Bubble sort algorithm.

A Objective:

- To understand and apply a simple sorting algorithm that can arrange elements in a list or array in ascending or descending order.
- To visualize and analyze sorting process.

B Expected Program Outcomes (POs):

1. **Basic and discipline specific knowledge:**
   Apply knowledge of basic mathematics, science and engineering fundamentals and engineering specialization to solve the engineering problems.
2. **Problem analysis:**
   Identify and analyse well-defined engineering problems using codified standard methods.
3. **Design/ development of solutions:**
   Design solutions for engineering well-defined technical problems and assist with the design of systems components or processes to meet specified needs.

C Expected Skills to be developed based on competency:

This practical is expected to develop the following skills for the industry-identified competency:

- ✓ Write and edit code for bubble sort method.
- ✓ Understand the concepts of sorting.
- ✓ Debug program to fix errors.
- ✓ Follow coding guidelines.

D Expected Course Outcomes (COs):

Apply different sorting and searching algorithms to the small data sets.

E Practical Outcome (PRo):

Implement bubble sort algorithm.

F Expected Affective Domain Outcomes (ADOs):

1. Follow safety practices.
2. Practice good housekeeping.
3. Demonstrate working as a leader/a team member.
4. Maintain tools and equipment.

5. Follow ethical practices.

G. Prerequisite theory:

A tree is called Binary Search tree (BST) if each and every node can have at most two branches. And it should contain following characteristics.

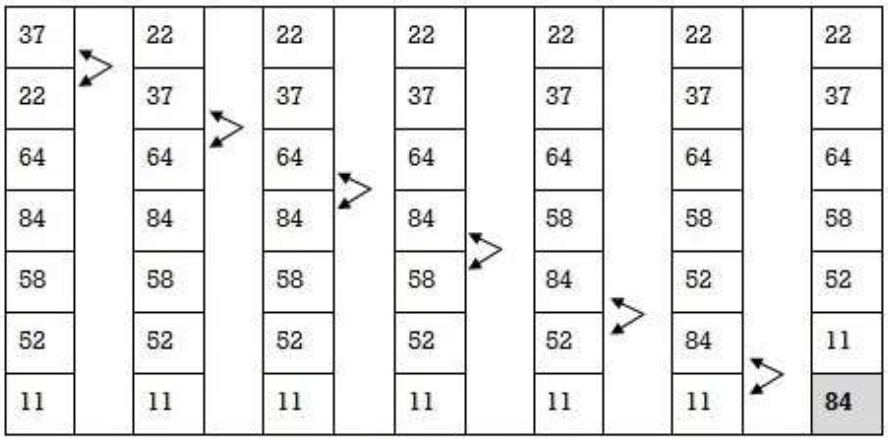
Bubble sort is the easiest and frequently used sorting algorithm among all other algorithms. Bubble sort focuses on successive adjacent pairs of elements in the array, compares them, and either swaps them or not.

Each pass through the list places the next largest value in its proper place. In essence, each item "bubbles" up to the location where it belongs. Maximum number of passes required is n – 1. (Where n is the total number of elements).

Example of Bubble sort:

Given data: 37, 22, 64, 84, 58, 52, 11

- Tracing of bubble sort is given below according to passes.



FIRST PASS

Finally get the result after six passes.

H. Resource / Equipment required

Sr. No.	Equipment / Software resources	Specifications
1	Computer system (Hardware Tool)	Computer having latest configuration with Windows or Unix OS
2	C compiler (Software Tool)	Open-source software

I. Safety and necessary precautions followed

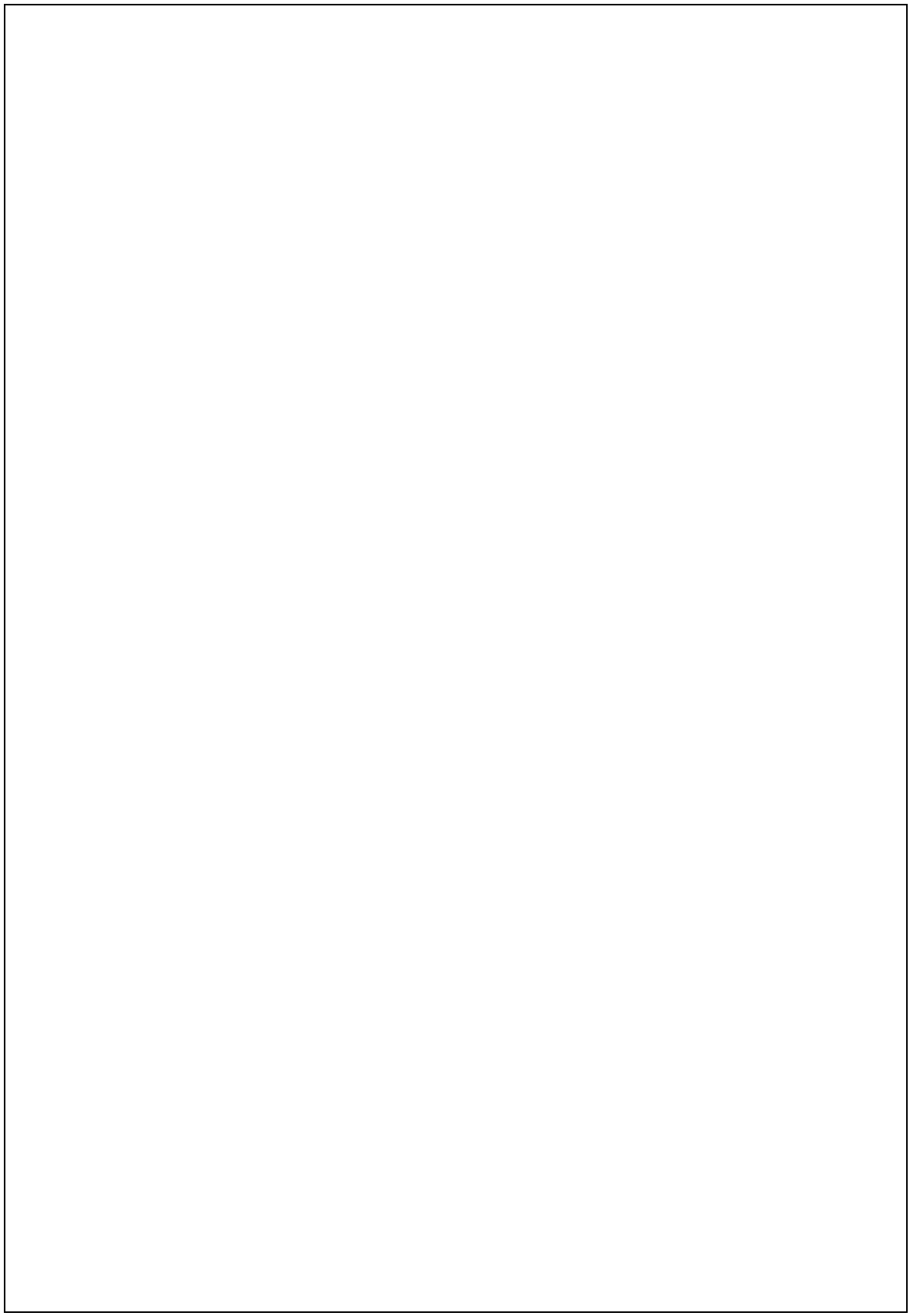
- Shutdown computer system properly once the Lab hours are finished.
- Strictly follow the instructions given by laboratory in-charge.

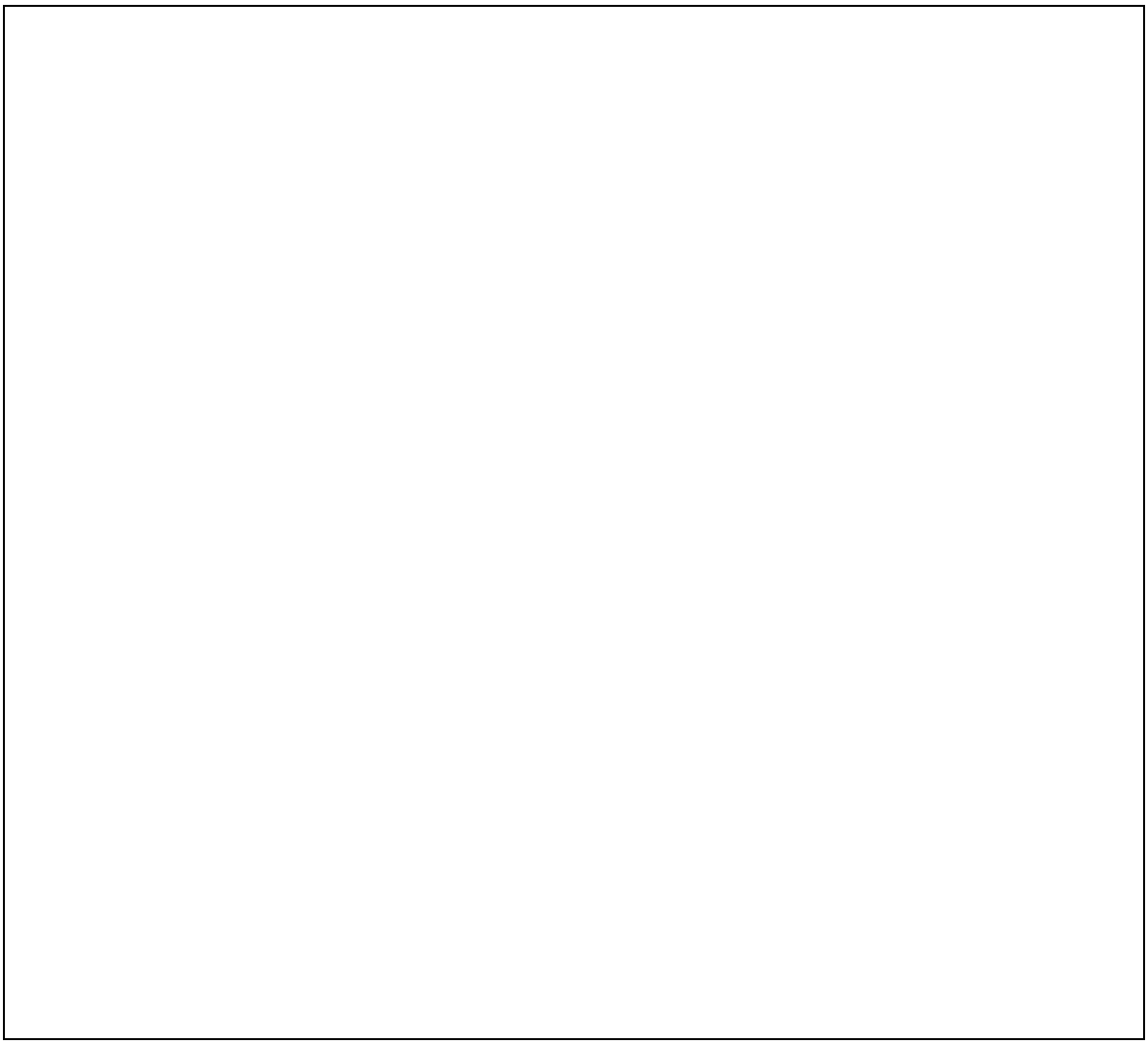
J. Algorithm and Program Source Code and Output

1) Write an algorithm for Bubble sort method.

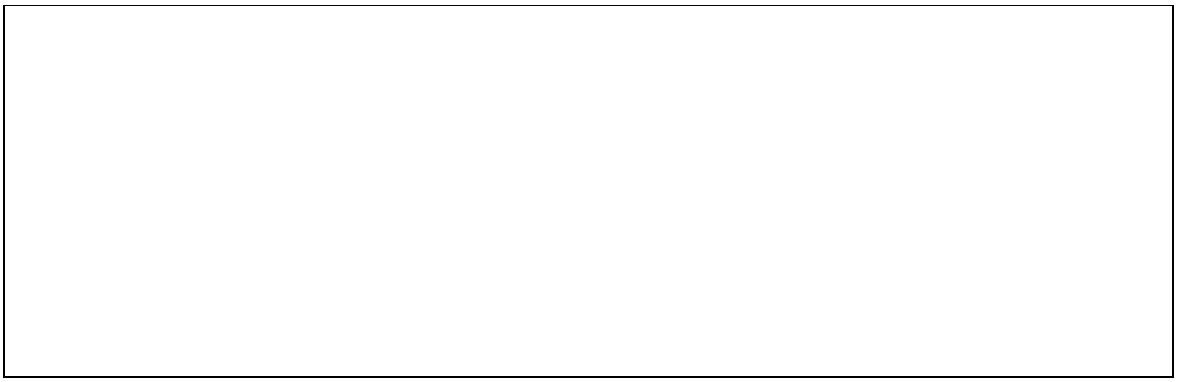
Algorithm

2) Write a Source Code to implement Bubble sort algorithm.





K. Input – Output



L. Practical related Quiz

Define sorting. List various sorting methods.

1. Fill in the blanks for the following.
   I. Bubble sort has a worst-case time complexity of _____.
   II. In bubble sort, the number of iterations required to sort an array of size 'n' is _____.
   III. In bubble sort method, after the first pass, the _____ element placed at its final position.
2. State TRUE (T) / FALSE (F) for the following.
   I. Bubble sort is an iterative and comparison-based sorting algorithm. ☐
   II. Bubble sort is performed only on ascending ordered data. ☐
   III. Bubble sort is the best sorting algorithm in all. ☐

M. References

I. https://www.javatpoint.com/bubble-sort
II. https://www.youtube.com/watch?v=re9ytVtt5zg
III. https://chat.openai.com/
IV. https://www.scaler.com/topics/data-structures/bubble-sort/
V. https://www.youtube.com/watch?v=xcPFUCh0jT0

N. Assessment Rubrics

Practical no. 19: Implement Bubble sort algorithm.				
Performance indicators for PROs	**Weighted Marks**	**Max Marks**	**Rubrics**	**Obtained Marks**
Correctness of program	**20%**	**5**	5 to 3 – Correct 2 – Partial correct 1 to 0 – Poor work	
Readability and documentation of the program/Quality of input and output displayed (messaging and formatting)	**20%**	**5**	5 to 4 – Excellent 3 to 2 – Good 1 to 0 – Poor	
Code efficiency	**10%**	**3**	3 – Excellent 2 – Good 1 to 0 – Poor	
Debugging ability	**20%**	**5**	5 to 4 – Excellent 3 to 2 – Good 1 to 0 – Poor	
Program execution/answer to questions	**30%**	**7**	7 to 5 – Proper execution 4 to 2 – Run with some errors 1 to 0 – Not run	
			Total Marks: (Out of 25)	

Signature with date

Date: _____

Practical No. 20: Implement Selection sort algorithm.

A Objective:

- To understand and apply a simple sorting algorithm that can arrange elements in a list or array in ascending or descending order.
- To visualize and analyze sorting process.

B Expected Program Outcomes (POs):

1. **Basic and discipline specific knowledge:**
   Apply knowledge of basic mathematics, science and engineering fundamentals and engineering specialization to solve the engineering problems.
2. **Problem analysis:**
   Identify and analyse well-defined engineering problems using codified standard methods.
3. **Design/ development of solutions:**
   Design solutions for engineering well-defined technical problems and assist with the design of systems components or processes to meet specified needs.

C Expected Skills to be developed based on competency:

This practical is expected to develop the following skills for the industry-identified competency:

- ✓ Write and edit code for selection sort method.
- ✓ Understand the concepts of sorting.
- ✓ Debug program to fix errors.
- ✓ Follow coding guidelines.

D Expected Course Outcomes (COs):

Apply different sorting and searching algorithms to the small data sets.

E Practical Outcome (PRo):

Implement selection sort algorithm.

F Expected Affective Domain Outcomes (ADOs):

1. Follow safety practices.
2. Practice good housekeeping.
3. Demonstrate working as a leader/a team member.
4. Maintain tools and equipment.

5. Follow ethical practices.

G. Prerequisite theory:

As the name suggests, selection sort selecting the smallest element in each iteration and put it into its proper position.

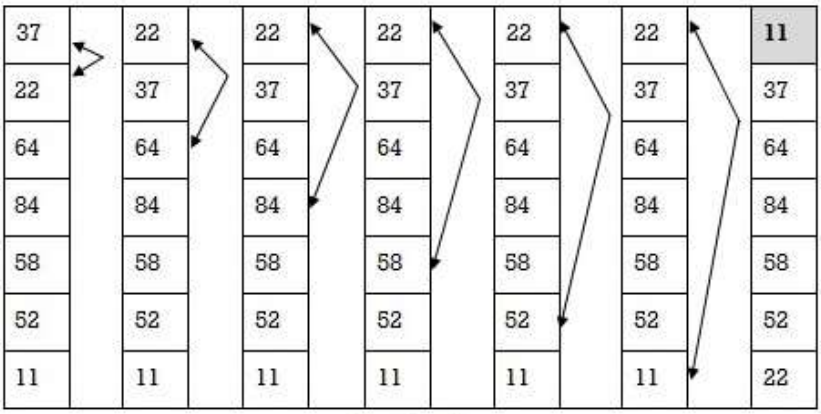
In each iteration, the key value is compared with the entire list until the smallest element is found. Then the smallest element is swapped with the key value. (Initially, the key value is the first element of items.)

Then, take second element as key value and find the second smallest element in the entire list. Do the same procedure up to the last element.

Example: sort the given data using selection sort.

- Trace of selection sort is given below.

  Given data: 37, 22, 64, 84, 58, 52, 11



Iteration 1

All other iterations are described in below figure.

Given Data	Iteration 1	Iteration 2	Iteration 3	Iteration 4	Iteration 5	Iteration 6
37	**11**	**11**	**11**	**11**	**11**	**11**
22	37	**22**	**22**	**22**	**22**	**22**
64	64	64	**37**	**37**	**37**	**37**
84	84	84	84	**52**	**52**	**52**
58	58	58	64	84	**58**	**58**
52	52	52	58	64	84	**64**
11	22	37	52	58	64	**84**

(Bold numbers are placed at their proper places. And after each iteration, smaller elements are placed at their proper places)

H. Resource / Equipment required

Sr. No.	Equipment / Software resources	Specifications
1	Computer system (Hardware Tool)	Computer having latest configuration with Windows or Unix OS
2	C compiler (Software Tool)	Open-source software

I. Safety and necessary precautions followed

- Shutdown computer system properly once the Lab hours are finished.
- Strictly follow the instructions given by laboratory in-charge.

J. Algorithm and Program Source Code

1) Write an algorithm for Selection sort method.

2) Write a Source Code to implement Selection sort algorithm.

K. Input – Output

L. Practical related Quiz

1. Fill in the blanks for the following.
   I. Selection sort has a worst-case time complexity of _____.

II. In selection sort, the number of iterations required to sort an array of size 'n' is _____.

III. In selection sort method, after the first pass, the _____ element placed at its final position.

IV. Selection sort has a best-case time complexity of _____.

2. State TRUE (T) / FALSE (F) for the following.

I. Selection sort is an iterative and comparison-based sorting algorithm. ☐

II. Bubble sort is performed best on large dataset. ☐

III. Selection sort always performs the same number of comparisons regardless of the input. ☐

M. References

I. http://www.btechsmartclass.com/data_structures/selection-sort.html

II. https://www.youtube.com/watch?v=_lqhHiPmg6g

III. https://techvidvan.com/tutorials/selection-sort/

IV. https://www.edureka.co/blog/selection-sort-in-c/

N. Assessment Rubrics

Practical no. 20: Implement Selection sort algorithm.				
Performance indicators for PROs	**Weighted Marks**	**Max Marks**	**Rubrics**	**Obtained Marks**
Correctness of program	**20%**	**5**	5 to 3 – Correct 2 – Partial correct 1 to 0 – Poor work	
Readability and documentation of the program/Quality of input and output displayed (messaging and formatting)	**20%**	**5**	5 to 4 – Excellent 3 to 2 – Good 1 to 0 – Poor	
Code efficiency	**10%**	**3**	3 – Excellent 2 – Good 1 to 0 – Poor	
Debugging ability	**20%**	**5**	5 to 4 – Excellent 3 to 2 – Good 1 to 0 – Poor	
Program execution/answer to questions	**30%**	**7**	7 to 5 – Proper execution 4 to 2 – Run with some errors 1 to 0 – Not run	
			Total Marks: (Out of 25)	

Signature with date

Date: _____

Practical No. 21: Implement Quick sort algorithm.

A Objective:

- The primary objective is to understand and be able to implement an efficient sorting algorithm that operates on a list or an array of elements.
- To understand the concept of divide and conquer methodology.

B Expected Program Outcomes (POs):

1. **Basic and discipline specific knowledge:**
   Apply knowledge of basic mathematics, science and engineering fundamentals and engineering specialization to solve the engineering problems.
2. **Problem analysis:**
   Identify and analyse well-defined engineering problems using codified standard methods.
3. **Design/ development of solutions:**
   Design solutions for engineering well-defined technical problems and assist with the design of systems components or processes to meet specified needs.

C Expected Skills to be developed based on competency:

This practical is expected to develop the following skills for the industry-identified competency:

- ✓ Understand the concepts of sorting.
- ✓ Write and edit code for quick sort method.
- ✓ Debug program to fix errors.
- ✓ Gain the ability of algorithmic thinking and problem solving.

D Expected Course Outcomes (COs):

Apply different sorting and searching algorithms to the small data sets.

E Practical Outcome (PRo):

Implement quick sort algorithm.

F Expected Affective Domain Outcomes (ADOs):

1. Follow safety practices.
2. Practice good housekeeping.
3. Demonstrate working as a leader/a team member.

4. Maintain tools and equipment.
5. Follow ethical practices.

G. Prerequisite theory:

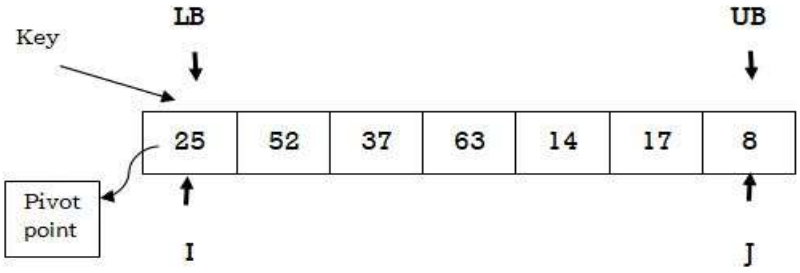
This technique works on the method of partitioning, so it is also called **partition exchange sort.** It is based on the rule of 'Divide and conquer' technique.

In each iteration (pass), one element is selected, which is called **pivot point**. And at the end of each iteration, pivot point is placed in its final position. At that time, two sub lists are created. (One on the left side of the pivot having less value than the value of pivot point and second list on the right side of the pivot point having high value than the value of the pivot point.

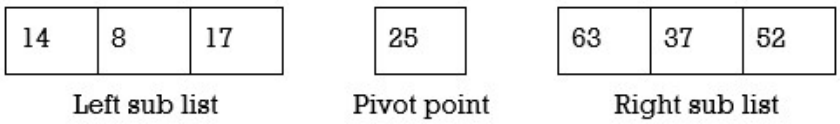
So, at the end of each iteration, this method divides the list into three main parts:

1. Elements less than pivot element
2. Pivot element
3. Elements greater than pivot element

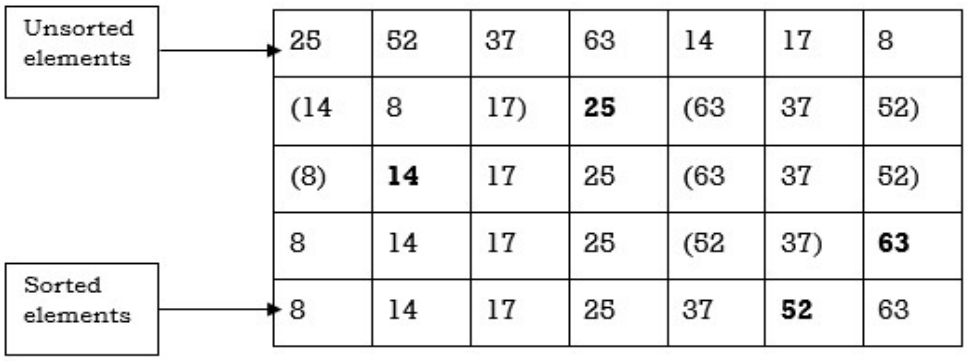
Then again same procedure (as above) is applied in both sub lists by taking an element as pivot element. This is why quick sort method is called recursive method as, the same procedure recursively applied; until we find the given data in sorted form.



At the end of this iteration array is partitioned like this



Full trace of quick sort given below.



25	52	37	63	14	17	8
(14	8	17)	**25**	(63	37	52)
(8)	**14**	17	25	(63	37	52)
8	14	17	25	(52	37)	**63**
8	14	17	25	37	**52**	63

H. Resource / Equipment required

Sr. No.	Equipment / Software resources	Specifications
1	Computer system (Hardware Tool)	Computer having latest configuration with Windows or Unix OS
2	C compiler (Software Tool)	Open-source software

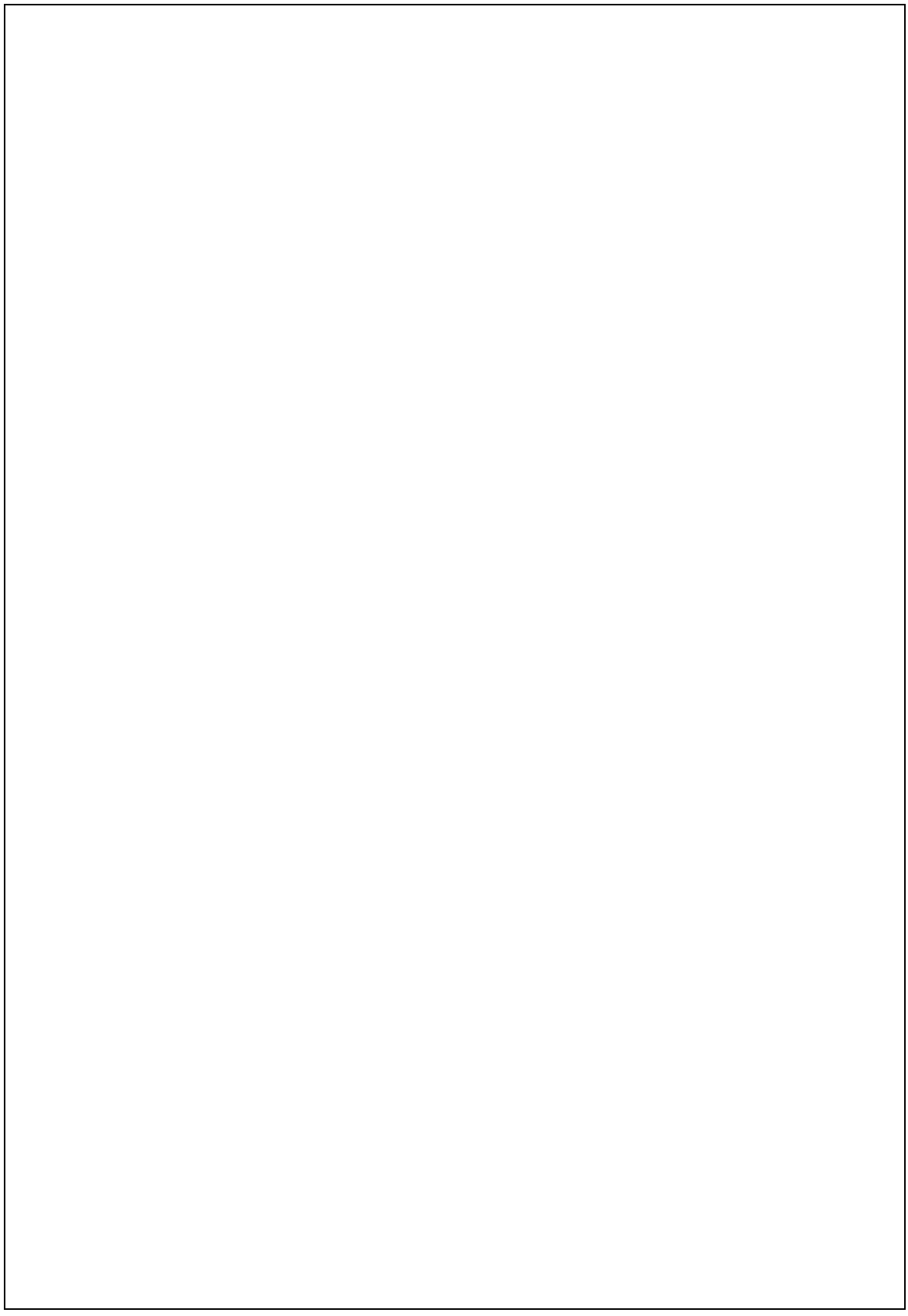
I. Safety and necessary precautions followed

- Shutdown computer system properly once the Lab hours are finished.
- Strictly follow the instructions given by laboratory in-charge.

J. Algorithm and Program Source Code

1) Write an algorithm for Quick sort method.

2) Write a Source Code to implement Quick sort algorithm.



K. Input – Output

L. Practical related Quiz

1. Fill in the blanks for the following.
   I. Quick sort has a worst-case time complexity of _____.
   II. The process of rearranging the elements around the pivot in quicksort is known as _____.
   III. Quick sort is also known as _____.
2. State TRUE (T) / FALSE (F) for the following.
   I. Quick sort method follows divide and conquer approach. ☐
   II. Quick sort performs recursion in the process of sorting. ☐
   III. Quick sort always performs the same number of comparisons regardless of the input. ☐

M. References

I. https://www.geeksforgeeks.org/quick-sort/
II. https://www.programiz.com/dsa/quick-sort
III. https://towardsdatascience.com/an-overview-of-quicksort-algorithm-b9144e314a72?gi=90df387e0796
IV. https://www.youtube.com/watch?v=tWCaFVJMUi8

N. Assessment Rubrics

Practical no. 21: Implement Quick sort algorithm.				
Performance indicators for PROs	**Weighted Marks**	**Max Marks**	**Rubrics**	**Obtained Marks**
Correctness of program	**20%**	**5**	5 to 3 – Correct 2 – Partial correct 1 to 0 – Poor work	
Readability and documentation of the program/Quality of input and output displayed (messaging and formatting)	**20%**	**5**	5 to 4 – Excellent 3 to 2 – Good 1 to 0 – Poor	
Code efficiency	**10%**	**3**	3 – Excellent 2 – Good 1 to 0 – Poor	
Debugging ability	**20%**	**5**	5 to 4 – Excellent 3 to 2 – Good 1 to 0 – Poor	
Program execution/answer to questions	**30%**	**7**	7 to 5 – Proper execution 4 to 2 – Run with some errors 1 to 0 – Not run	
			Total Marks: (Out of 25)	

Signature with date

Date: _____

Practical No. 22: Implement Insertion sort algorithm.

A Objective:

- The primary objective is to understand and be able to implement a simple and efficient sorting algorithm that operates on a list or an array of elements.
- To understand the concept of an incremental sorting algorithm.

B Expected Program Outcomes (POs):

1. **Basic and discipline specific knowledge:**
   Apply knowledge of basic mathematics, science and engineering fundamentals and engineering specialization to solve the engineering problems.
2. **Problem analysis:**
   Identify and analyse well-defined engineering problems using codified standard methods.
3. **Design/ development of solutions:**
   Design solutions for engineering well-defined technical problems and assist with the design of systems components or processes to meet specified needs.

C Expected Skills to be developed based on competency:

This practical is expected to develop the following skills for the industry-identified competency:

- ✓ Understand the concepts of sorting with incremental approach.
- ✓ Write and edit code for insertion sort method.
- ✓ Debug program to fix errors.

D Expected Course Outcomes (COs):

Apply different sorting and searching algorithms to the small data sets.

E Practical Outcome (PRo):

Implement insertion sort algorithm.

F Expected Affective Domain Outcomes (ADOs):

1. Follow safety practices.
2. Practice good housekeeping.
3. Demonstrate working as a leader/a team member.
4. Maintain tools and equipment.
5. Follow ethical practices.

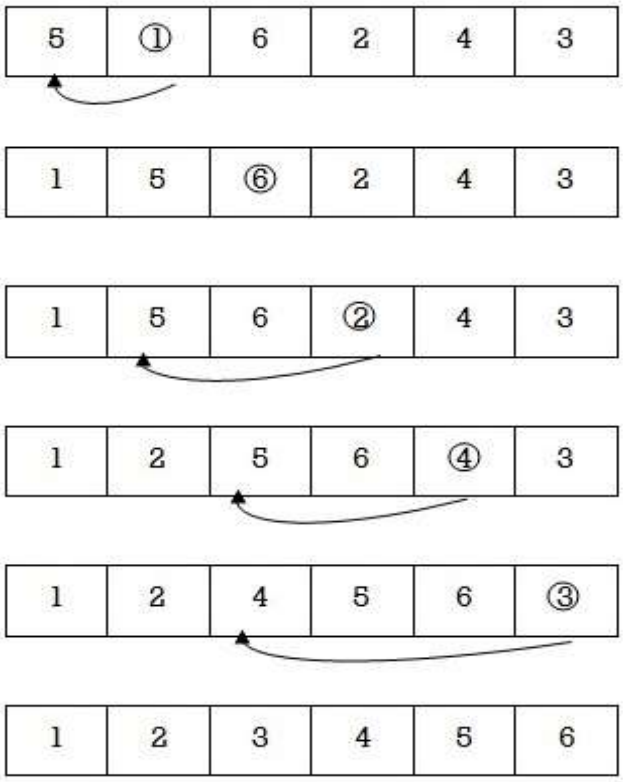
G. Prerequisite theory:

A very simple example of insertion sort is → when we are playing a game of cards; we are taking cards one by one and arranging them accordingly.

Example: sort the given data using insertion sort method.

Given data: 5, 1, 6, 2, 4, 3

(Always start with the second element as key)



H. Resource / Equipment required

Sr. No.	Equipment / Software resources	Specifications
1	Computer system (Hardware Tool)	Computer having latest configuration with Windows or Unix OS
2	C compiler (Software Tool)	Open-source software

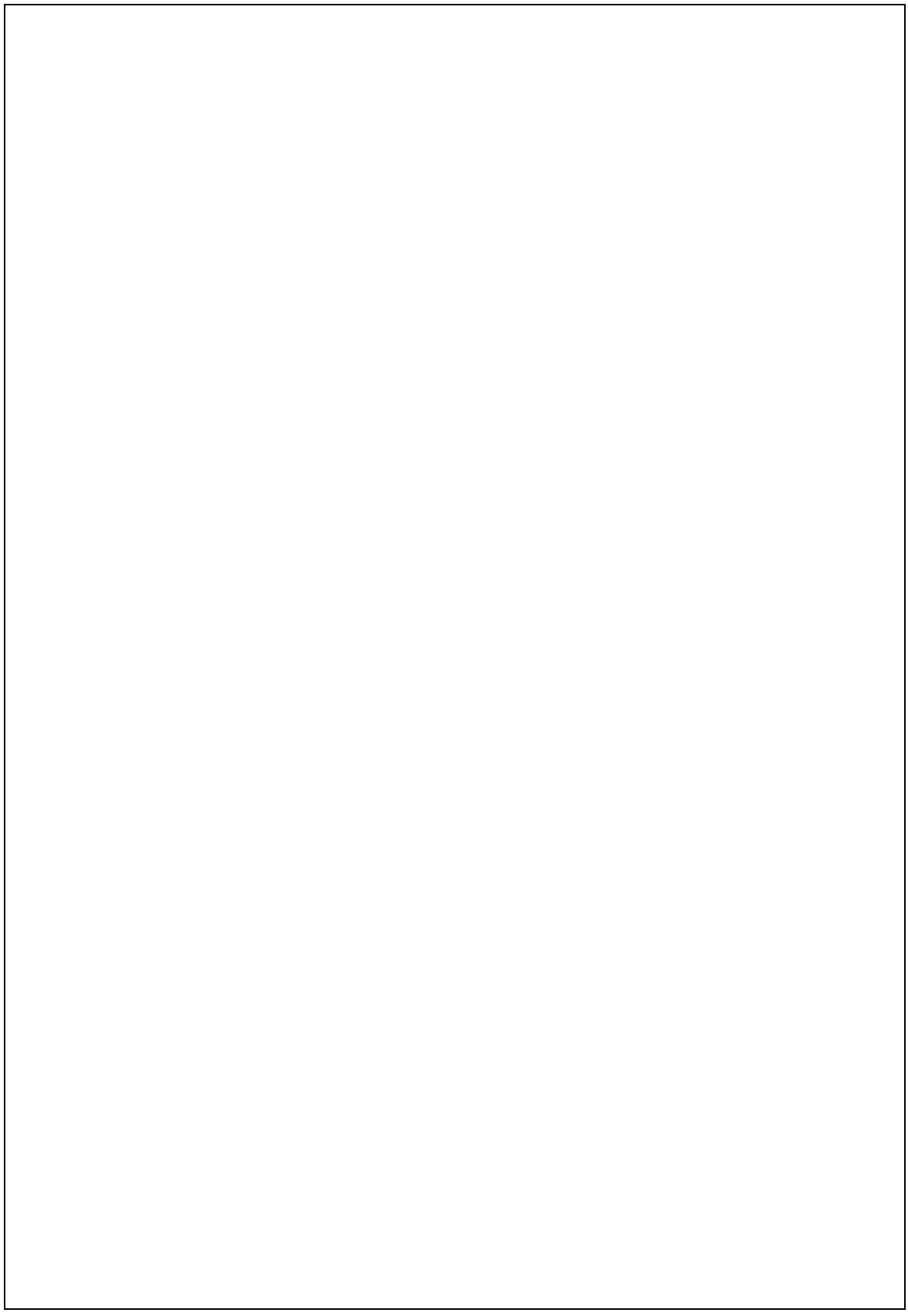
I. Safety and necessary precautions followed

- Shutdown computer system properly once the Lab hours are finished.
- Strictly follow the instructions given by laboratory in-charge.

J. Algorithm and Program Source Code

1) Write an algorithm for Insertion sort method.

2) Write a Source Code to implement insertion sort algorithm.



K. Input – Output

L. Practical related Quiz

1. Fill in the blanks for the following.
   I. Insertion sort has a worst-case time complexity of _____.
   II. _____ number of passes are required to sort 8 elements using insertion sort method.
2. State TRUE (T) / FALSE (F) for the following.
   I. Insertion sort is a kind of external sort.
   II. Insertion sort is efficient when given data is small.

M. References

I. https://www.javatpoint.com/insertion-sort

II. https://www.khanacademy.org/computing/computer-science/algorithms/insertion-sort/a/insertion-sort
III. https://www.hackerearth.com/practice/algorithms/sorting/insertion-sort/tutorial/
IV. https://www.youtube.com/watch?v=3GC83dh4cf0

N. Assessment Rubrics

Practical no. 22: Implement Insertion sort algorithm.				
Performance indicators for PROs	**Weighted Marks**	**Max Marks**	**Rubrics**	**Obtained Marks**
Correctness of program	**20%**	**5**	5 to 3 – Correct 2 – Partial correct 1 to 0 – Poor work	
Readability and documentation of the program/Quality of input and output displayed (messaging and formatting)	**20%**	**5**	5 to 4 – Excellent 3 to 2 – Good 1 to 0 – Poor	
Code efficiency	**10%**	**3**	3 – Excellent 2 – Good 1 to 0 – Poor	
Debugging ability	**20%**	**5**	5 to 4 – Excellent 3 to 2 – Good 1 to 0 – Poor	
Program execution/answer to questions	**30%**	**7**	7 to 5 – Proper execution 4 to 2 – Run with some errors 1 to 0 – Not run	
			Total Marks: (Out of 25)	

Signature with date

Date: _____

Practical No. 23: Implement Merge sort algorithm.

A Objective:

- The primary objective is to understand and be able to implement a simple and efficient sorting algorithm that operates on a list or an array of elements.
- To understand divide and conquer approach.

B Expected Program Outcomes (POs):

1. **Basic and discipline specific knowledge:**
   Apply knowledge of basic mathematics, science and engineering fundamentals and engineering specialization to solve the engineering problems.
2. **Problem analysis:**
   Identify and analyse well-defined engineering problems using codified standard methods.
3. **Design/ development of solutions:**
   Design solutions for engineering well-defined technical problems and assist with the design of systems components or processes to meet specified needs.

C Expected Skills to be developed based on competency:

This practical is expected to develop the following skills for the industry-identified competency:

- ✓ Understand the concepts of merge sorting.
- ✓ Write and edit code for merge sort method.
- ✓ Debug program to fix errors.

D Expected Course Outcomes (COs):

Apply different sorting and searching algorithms to the small data sets.

E Practical Outcome (PRo):

Implement merge sort algorithm.

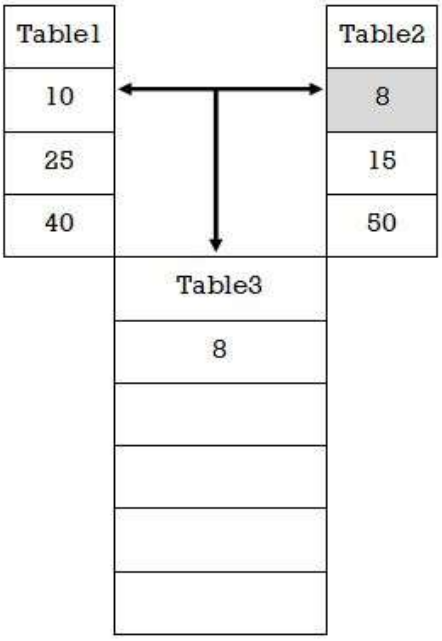
F Expected Affective Domain Outcomes (ADOs):

1. Follow safety practices.
2. Practice good housekeeping.
3. Demonstrate working as a leader/a team member.
4. Maintain tools and equipment.
5. Follow ethical practices.

G. Prerequisite theory:

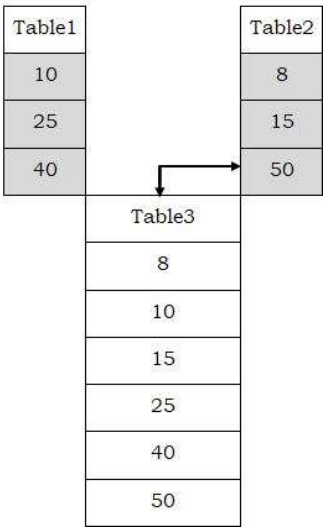
Merge sort is used to merge (combine) two ordered lists into third list. So that we need to have two sorted array and combine them into third array in sorted manner.

Suppose we have two sorted tables table1 and table2. Then in this method, we have to compare the elements of both tables one by one and the element which has the smaller value, it will be placed in a third table.



This process is continued until last element. In every step, each element from both the tables are compared and put the lower element in Table3.

Finally, after completion of all the comparisons, we will have following data:



H. Resource / Equipment required

Sr. No.	Equipment / Software resources	Specifications
1	Computer system (Hardware Tool)	Computer having latest configuration with Windows or Unix OS
2	C compiler (Software Tool)	Open-source software

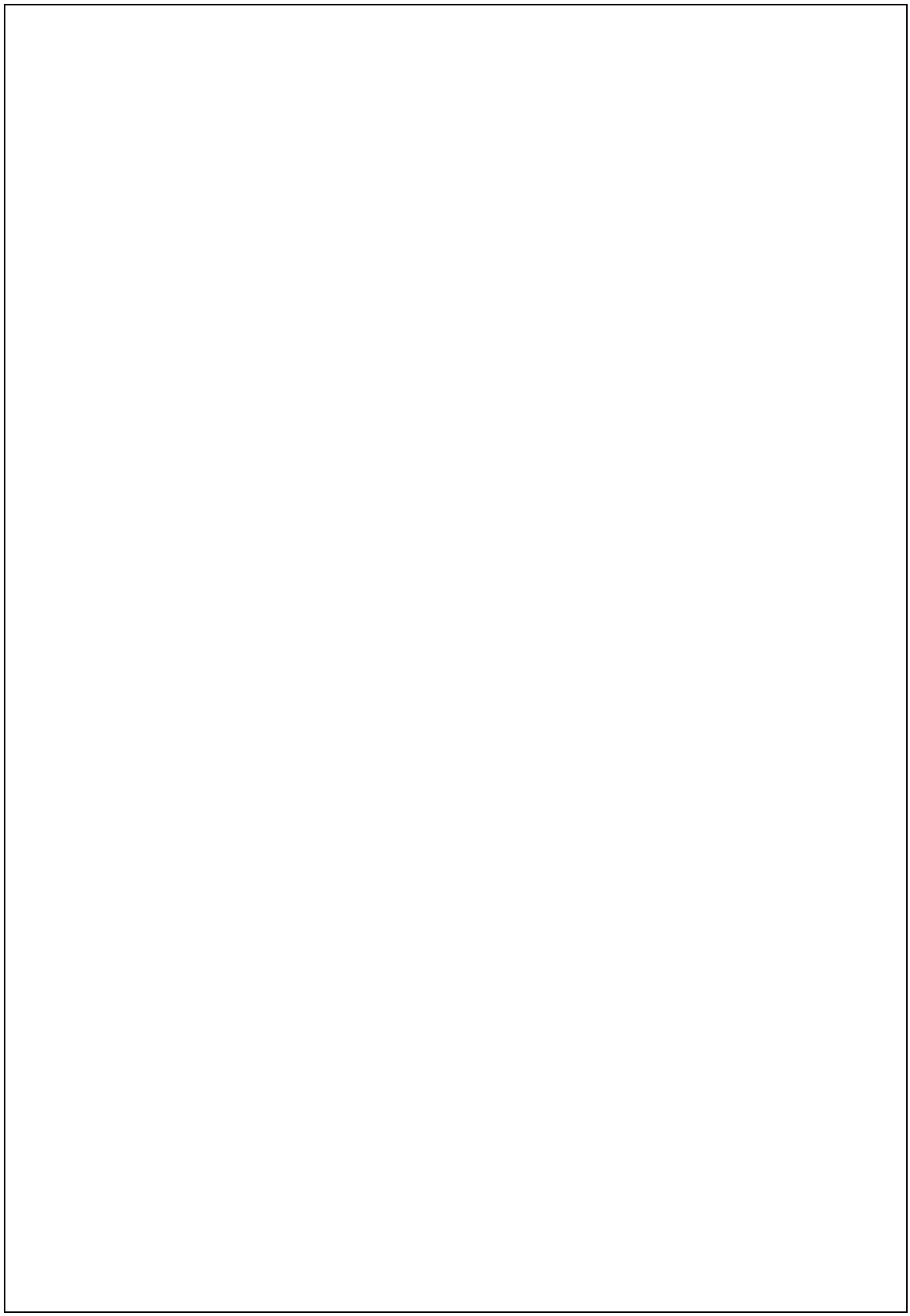
I. Safety and necessary precautions followed

- Shutdown computer system properly once the Lab hours are finished.
- Strictly follow the instructions given by laboratory in-charge.

J. Algorithm and Program Source Code

1) Write an algorithm for Merge sort method.

2) Write a Source Code to implement Merge sort algorithm.



K. Input – Output

L. Practical related Quiz

1. Fill in the blanks for the following.
   I. Merge sort has a worst-case time complexity of _____.
   II. _____ number of passes are required to sort 8 elements using insertion sort method.
2. State TRUE (T) / FALSE (F) for the following.
   I. Merge sort follows divide and conquer approach. ☐
   II. Merge sort is a recursive sorting algorithm. ☐

M. References

I. https://www.gatevidyalay.com/data-structures/
II. https://takeuforward.org/data-structure/merge-sort-algorithm/
III. https://www.digitalocean.com/community/tutorials/merge-sort-algorithm-java-c-python
IV. https://www.youtube.com/watch?v=KF2j-9iSf4Q

N. Assessment Rubrics

Practical no. 23: Implement Merge sort algorithm.				
Performance indicators for PROs	**Weighted Marks**	**Max Marks**	**Rubrics**	**Obtained Marks**
Correctness of program	**20%**	**5**	5 to 3 – Correct 2 – Partial correct 1 to 0 – Poor work	
Readability and documentation of the program/Quality of input and output displayed (messaging and formatting)	**20%**	**5**	5 to 4 – Excellent 3 to 2 – Good 1 to 0 – Poor	
Code efficiency	**10%**	**3**	3 – Excellent 2 – Good 1 to 0 – Poor	
Debugging ability	**20%**	**5**	5 to 4 – Excellent 3 to 2 – Good 1 to 0 – Poor	
Program execution/answer to questions	**30%**	**7**	7 to 5 – Proper execution 4 to 2 – Run with some errors 1 to 0 – Not run	
			Total Marks: (Out of 25)	

Signature with date

Date: _____

Practical No. 24: Solve hash table example using division method, mid square method and folding method.

A Objective:

- The primary objective is to understand and apply a data structure that provides efficient access and retrieval of data.
- Able to resolve collision and get efficient data storage.

B Expected Program Outcomes (POs):

1. **Basic and discipline specific knowledge:**
   Apply knowledge of basic mathematics, science and engineering fundamentals and engineering specialization to solve the engineering problems.
2. **Problem analysis:**
   Identify and analyse well-defined engineering problems using codified standard methods.
3. **Design/ development of solutions:**
   Design solutions for engineering well-defined technical problems and assist with the design of systems components or processes to meet specified needs.

C Expected Skills to be developed based on competency:

This practical is expected to develop the following skills for the industry-identified competency:

- ✓ Understand the concepts of hashing.
- ✓ Solve examples using division method, mid-square method and folding method.

D Expected Course Outcomes (COs):

Apply different sorting and searching algorithms to the small data sets.

E Practical Outcome (PRo):

Solve hash table example using division method, mid square method and folding method.

F Expected Affective Domain Outcomes (ADOs):

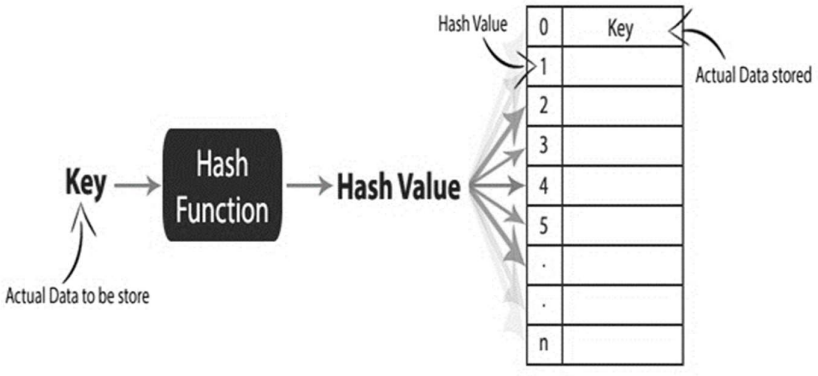
1. Follow safety practices.
2. Practice good housekeeping.

3. Demonstrate working as a leader/a team member.
4. Maintain tools and equipment.
5. Follow ethical practices.

G. Prerequisite theory:

Hashing is a searching technique performed in constant time. Because Hashing provides direct access of record from the file no matter where the record is stored in the file.

With the help of good hash function, operations like insertion, deletion and search can be performed with constant time complexity (i.e. O(1)).



There are different Hash table methods available to build various Hash functions, like:

The division method

Mid square method

Folding method

H. Practical related Quiz

1. Define hashing. List various hashing functions.
2. Apply Division method on following data: 11. 20, 35, 76, 87, 999, 245, 56
   Table size: 11 (Use linear probing to avoid collision)
3. Apply Mid square method on following data:
   a) 123456
   b) 456789
   c) 154674
4. Apply folding method on following data:
   a) 356942781 (Make partition of 3 digit)
   b) 123456 (Make partition of 2 digit)

c) 345678912654 (Make partition of 4 digit)

5. Fill in the blanks for the following.
   I. The output of a hashing algorithm is called a _____.
   II. In hashing, a _____ is a unique identifier that represents a specific input or data.
   III. The time complexity of a good hashing algorithm for average-case scenarios is usually _____.
   IV. _____ is a method used to distribute keys more uniformly across the slots of a hash table, reducing the likelihood of collisions.
6. State TRUE (T) / FALSE (F) for the following.
   I. Hashing is a process of taking an input (or message) and transforming it into a fixed-size string of characters. ☐
   II. The size of a hash value is typically fixed regardless of the size of the input. ☐
   III. Folding method is the best among all hashing hash table methods. ☐
   IV. Hashing is often used for fast and efficient retrieval of data. ☐
   V. Hashing guarantees a unique hash value for each unique input. ☐
   VI. Hashing is a reversible process, meaning it is possible to obtain the original input from its hash value. ☐

I. References

I. https://courses.cs.washington.edu/courses/cse332/17wi/lectures/hashing-2/hashing-2-6up.pdf
II. https://www.knowledgehut.com/blog/programming/hashing-in-data-structure
III. https://www.javatpoint.com/hash-table
IV. https://www.tutorialride.com/data-structures/hashing-in-data-structure.htm
V. https://www.youtube.com/watch?v=W5q0xgxmRd8

J. Assessment Rubrics

Practical no. 24: Solve hash table example using division method, mid square method and folding method.				
Performance indicators for PROs	**Weighted Marks**	**Max Marks**	**Rubrics**	**Obtained Marks**
Correctness of program	**20%**	**5**	5 to 3 – Correct 2 – Partial correct 1 to 0 – Poor work	
Readability and documentation of the program/Quality of input and output displayed (messaging and formatting)	**20%**	**5**	5 to 4 – Excellent 3 to 2 – Good 1 to 0 – Poor	
Code efficiency	**10%**	**3**	3 – Excellent 2 – Good 1 to 0 – Poor	
Debugging ability	**20%**	**5**	5 to 4 – Excellent 3 to 2 – Good 1 to 0 – Poor	
Program execution/answer to questions	**30%**	**7**	7 to 5 – Proper execution 4 to 2 – Run with some errors 1 to 0 – Not run	
			Total Marks: (Out of 25)	

Signature with date