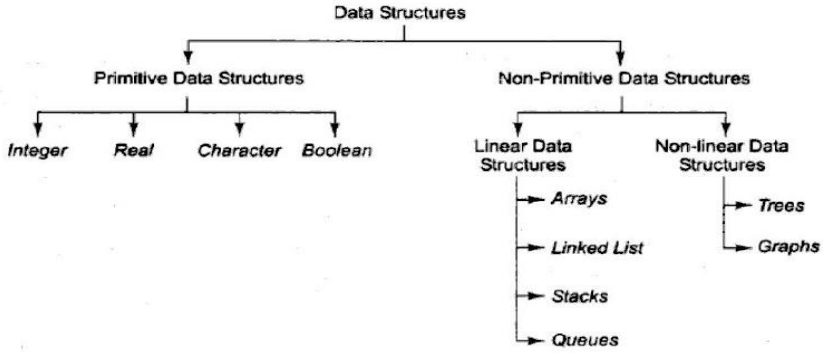
BHANDARKARS' ARTS AND SCIENCE COLLEGE KUNDAPURA
COMPUTER SCIENCE DEPARTMENT
II – BCA DATA STRUCTURE (2 MARKS QUESTION AND ANSWER)

1. **Define data structure.**

Data may be organized in many different ways; **the logical or mathematical model of a particular organization of data is called a data structure**.

2. **Write the classification of data structure**

Data structures are generally classified into primitive and non-primitive data structures.
non-primitive data structures are further classified into linear and non-linear.



3. **Name the 2 classification of data structure**

Data structures are generally classified into primitive and non-primitive data structures.
Basic data types such as integer, real, character and boolean are known as primitive data structures.

Non-primitive data structures are derived from the primitive data structures.

4. **Define primitive data structure. Give an example.**

Basic data types such as integer, real, character and boolean are known as primitive data structures. These data types consist of characters that cannot be divided, and hence they are also called simple data types.

5. Define non-primitive data structure. Give an example.

Non-primitive data structures are derived from the primitive data structures.The simplest example of non-primitive data structure is the processing of complex numbers. Very few computers are capable of doing arithmetic on complex numbers. Linked-lists, stacks, queues, trees and graphs are examples of non-primitive data structures.

6. Write the classification of non-primitive data structure.

Non-primitive data structures are further classified into linear and non-linear data structure.

A data structure is said to be linear if its elements form a sequence or a linear list.
A data structure is said to be non-linear if the data is not arranged in sequence.

7. Differentiate linear and non-linear data structure.

A data structure is said to be linear if its elements form a sequence or a linear list. In linear data structures, the data is arranged in a linear fashion although the way they are stored in memory need not be sequential. Arrays, linked lists, stacks and queues are examples of linear data structures.
A data structure is said to be non-linear if the data is not arranged in sequence. The insertion and deletion of data is therefore not possible in a linear fashion. Trees and graphs are examples of non-linear data structures.

8. Name any 4 operations performed by data structure.

Define sorting and merging.
Define traversing and searching.
Traversal : Processing each element in the list.
Search.: Finding the location of the element with a given value or the record with a given key.
Insertion.: Adding a new element to the list.
Deletion. : Removing an element from the list.
Sorting.: Arranging the elements in some type of order.
Merging.: Combining two lists into a single list.

9. Write any two algorithmic notations with examples.

Variable Names

Variable names will use capital letters, as in MAX and DATA.

Assignment Statement

Our assignment statements will use the dots-equal notation: = that is used in Pascal. For example,

Max: = DATA[1]

Input and Output

Data may be input and assigned to variables by means of a Read statement with the following form:

Read: Variables names.

Similarly, messages, placed in quotation marks, and data in variables may be output by means of a Write or Print statement with the following form:

Write: Messages and/or variable names.

10. Name any 2 types of control structures used in algorithms.

1. Sequence logic or sequential flow

2. Selection logic or conditional flow

3 .Iteration logic or repetitive flow

11. Write a note on sequence logic.

Unless instructions are given to the contrary, the modules are executed in the obvious Sequence. The sequence may be presented explicitly, by means of numbered steps, or implicitly, by the order in which the modules are written.

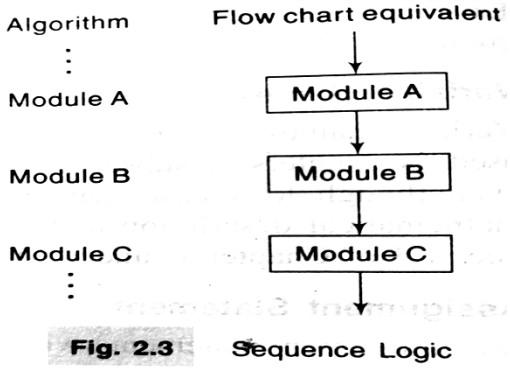


Fig. 2.3 Sequence Logic

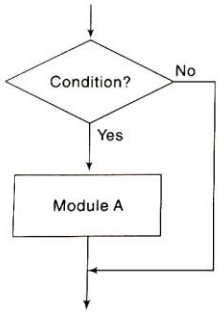
12. Write the structure of single alternative.

Single Alternative. This structure has the form

If condition, then:

[Module A]

[End of If structure.]

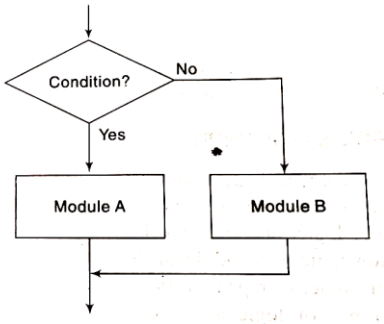


(a) Single alternative.

11. Write the structure of double alternative.

Double Alternative. This structure has the form

If condition, then:
[Module A]
Else:
[Module B]
[End of If structure.]



(b) Double alternative.

12. Write the structure of multiple alternatives.

Multiple Alternatives. This structure has the form

If condition(1), then:
[Module A_1]
Else if condition(2), then:
[Module A_2]

.
.

.

Else if condition (M), then:
[Module A_M]
Else:
[Module B]
[End of If structure]

13. Write a note on iteration logic.

begins with a Repeat statement and is followed by a module, called the body of the loop.

The repeat-for loop uses an index variable, such as K, to control the loop. The loop will usually have the form:

Repeat for K = R to S by T:
[Module]
[End of loop.]

The repeat-while loop uses a condition to control the loop.The loop will usually have the form

Repeat while condition:
[Module]
[End of loop.]

14. **Define linear array. Write the formula to obtain number of data** elements in an array.

A data structure is said to be linear if its elements form a sequence, or, in other words, a linear list.

Length = UB -LB +1

13. What is sparse matrix? Give example.

Matrix with relatively a high proportion of zero entries are called sparse matrix.

Two general types of n-square sparse matrices are

Triangular matrix
Tridiagonal matrix

14. What is lower-triangular matrix? Give an example.

This is the matrix where all the entries above the main diagonal are zero or equivalently where non-zero entries can only occur on or below the main diagonal is called a (lower) Triangular matrix.

15. What is upper-triangular matrix? Give an example.

This is the matrix where all the entries bellow the main diagonal are zero or equivalently where non-zero entries can only occur on or above the main diagonal is called a (upper) Triangular matrix.

16. What is tri-diagonal matrix? Give an example.

This is the matrix where non-zero entries can only occur on the diagonal or on elements immediately above or below the diagonal is called a Tridiagonal matrix.

Unit – 2

1. Define searching. Name 2 types of searching techniques.

Searching in data structure refers to the process of finding location LOC of an element in a list.

- Sequential Search. Sequential search is also called as Linear Search. Sequential search starts at the beginning of the **list** and checks every element of the **list**. ...
- **Binary** Search. **Binary** Search is used for **searching** an element in a sorted array.

2. What do you mean by successful search and unsuccessful search?

All **successful searches** terminate when the object of the **search** is found. Therefore, all **successful searches** terminate at an internal node.

All successful **searches** terminate when the object of the **search** is found. In contrast, all **unsuccessful searches** terminate at an external node.

3. **What do you mean by sequential search?**

In computer science, a **linear search** or **sequential search** is a method for finding an element within a list. It **sequentially** checks each element of the list until a match is found or the whole list has been searched.

4. **What do you mean by binary search?**

Binary search is an efficient algorithm for finding an item from a sorted list of items. It works by repeatedly dividing in half the portion of the list that could contain the item, until **you**'ve narrowed down the possible locations to just one. Write one advantage and disadvantage of binary search.

5. **What do you mean by sorting? Write any 2 sorting techniques.**

Let A be a list of n numbers. **Sorting A refers to the operation of rearranging the elements of A So they are in increasing order, i.e., so that,**

A[1]< A[2]< A[3] <... <A[N]

Sorting Techniques

- Bubble sort
- Selection sort
- Insertion sort
- Merge sort
- Shell sort
- Radix sort

6. **What is merging?**

Merging.: Combining two lists into a single list.

7. **What is diminishing increment sort?**

The shell **sort**, **sometimes called the** "**diminishing increment sort**," improves on the insertion **sort** by breaking the original list into a number of smaller sublists, each of which is **sorted** using an insertion **sort**.

8. **What is linked list?**

A linked list is a non-sequential collection of data items, **there is an associated pointer that would give the memory location of the next data item in the linked list**. The **data items in the linked list are not in a consecutive memory locations**.

9. **What is free storage list?**

10. What is garbage collection?

The operating system of a computer may periodically collect all the deleted space onto the free storage list. Any technique which does this collection is called garbage collection.

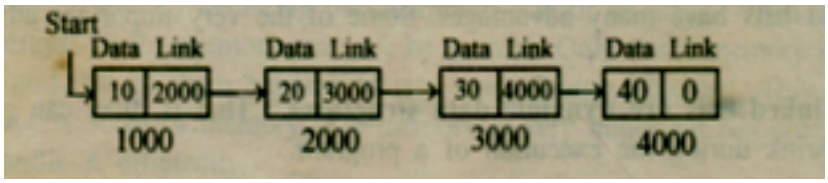
11. What role does the AVAIL list play in a linked list?

The unused memory cells in the arrays will also be linked together to form a linked list using AVAIL as its list pointer variable. (Hence this free-storage list will also be called the AVAIL list.) Such a data structure will frequently be denoted by writing

LIST (INFO, LINK, START, AVAIL)

12. How is singly linked list terminated? Give diagram.

The **link field of the last node contains zero** rather than a valid address. It is a **null pointer** and **indicates the end of the list**.



13. What is underflow and overflow with respect to linked list?

The term underflow refers to the situation where one wants to delete data from a data structure that is empty. Observe that **underflow will occur with our linked lists when START = NULL and there is a deletion**.

Sometimes **new data are to be inserted into a data structure but there is no available space,** i.e., the free-storage list is empty. **This situation is usually called overflow.**

14. Define circularly linked list. Give its diagram.

It is just a singly linked list in which the link field of the last node contains the address of the first node of the list. That is, the link field of the last node does not point to NULL rather it points back to the beginning of the linked list.

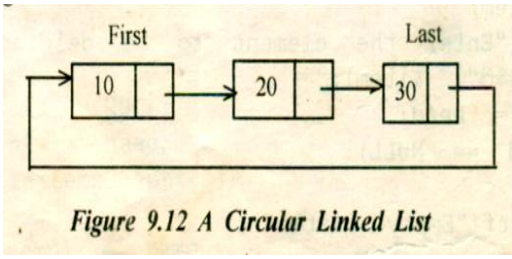
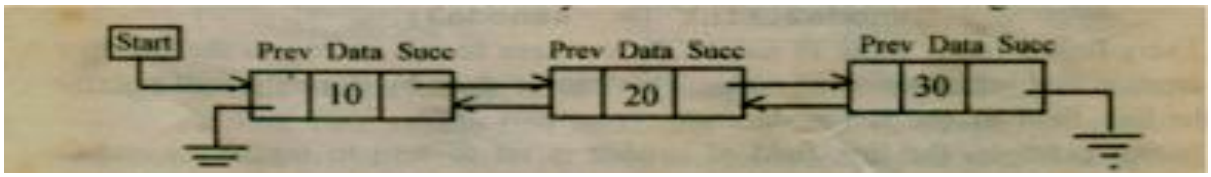


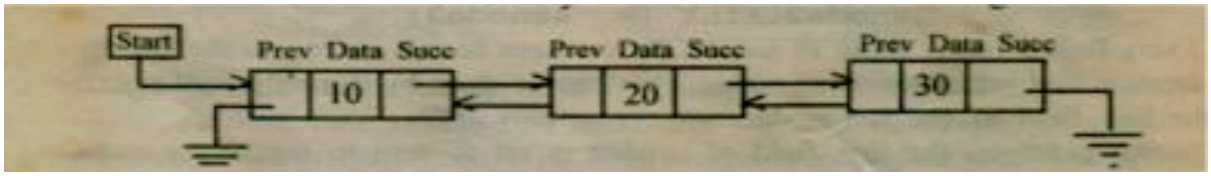
Figure 9.12 A Circular Linked List

15. **What is two-way list? Give its diagram. OR What is doubly linked** list? Give its diagram.

A doubly linked list is one in which all **nodes are linked together by multiple links which help in accessing both the successor node (next node) and predecessor node** (previous node) for any arbitrary node within the list. Therefore each node in a doubly linked list has two link fields (pointers) to point to the left node (previous) and the right node(next).



16. Write the node diagram of two-way list.



17. What is a null pointer? What is its significance?

A **null pointer** is a **pointer** which points nothing.

Some uses of **the null pointer** are:

a) To initialize a **pointer** variable when that **pointer** variable isn't assigned any valid memory address yet.

b) To pass a **null pointer** to a function argument when we don't want to pass any valid memory address.

18. Mention any 2 operations performed on two-way lists.

Following are the basic operations supported by a list.

- **Insertion** − Adds an element at the beginning of the list.
- **Deletion** − Deletes an element at the beginning of the list.
- **Insert Last** − Adds an element at the end of the list.
- **Delete Last** − Deletes an element from the end of the list.
- **Insert After** − Adds an element after an item of the list.
- **Delete** − Deletes an element from the list using the key.
- **Display forward** − Displays the complete list in a forward manner.
- **Display backward** − Displays the complete list in a backward manner

Unit -3

1. **What is stack? Name the basic operations performed on stack.**

A stack is a list of elements in which an element may be inserted or deleted only at one end, called the top of the stack

Operations on a Stack

The basic operations that can be performed on a stack are listed in this table

Operation	Description
PUSH	This adds(or pushes) a new data item on to the stack
POP	This deletes the data item from the top of the stack
TOP	It returns the *value* of the item at the top of the stack
Stack_empty	It returns *true* if the stack is empty. Otherwise it returns false
Stack_full	It returns *true* if the stack is Full. Otherwise it returns false

2) What is stack? Why stack is called LIFO list?

A stack is a list of elements in which an element may be inserted or deleted only at one end, called the top of the stack.

LIFO :This means, in particular, **element are removed from a stack in the reverse order of that in which they were inserted into the stack.**

3) What is LIFO list? Mention any 2 its applications.

A stack is a LIFO list.

Applications of Stack

- Recursion Functions
- Expression Conversion
- Expression Evaluation

5) What do you mean by PUSH and POP operation?

(a) "Push" is the term used to insert an element into a stack.

(b) "Pop" is the term used to delete an element from a stack.

6) Expand LIFO and FIFO.

LIFO : LAST IN FIRST OUT.

FIFO : FIRST IN FIRST OUT.

7) What is a queue? Why queue is called FIFO list?

A queue is a linear list of elements in which deletions can take place only at one end, called the front, and insertions can take place only at the other end, called the rear.

Queues are also called first-in first-out (FIFO) lists, since the first element in a queue will be the first element out of the queue. In other words, the order in which elements enter a queue is the order in which they leave.

8) How does Stack differ from Queue?

Difference between Stack and Queue Data Structures

Stacks	Queues
Stacks are based on the LIFO principle, i.e., the element inserted at the last, is the first element to come out of the list.	Queues are based on the FIFO principle, i.e., the element inserted at the first, is the first element to come out of the list.
Insertion and deletion in stacks takes place only from one end of the list called the top.	Insertion and deletion in queues takes place from the opposite ends of the list. The insertion takes place at the rear of the list and the deletion takes place from the front of the list.
Insert operation is called push operation.	Insert operation is called enqueue operation.
Delete operation is called pop operation.	Delete operation is called dequeue operation.
In stacks we maintain only one pointer to access the list, called the top, which always points to the last element present in the list.	In queues we maintain two pointers to access the list. The front pointer always points to the first element inserted in the list and is still present, and the rear pointer always points to the last inserted element.
Stack is used in solving problems works on recursion.	Queue is used in solving problems having sequential processing

9) Write an algorithm for PUSH operation using arrays.

PUSH(STACK, TOP, MAXSTK, ITEM)

This procedure pushes an ITEM onto a stack.

1. [Stack already filled?]
   If TOP = MAXSTK, then: Print: OVERFLOW, and Return.
2. Set TOP: = TOP + 1. [Increases TOP by 1.]

3. Set STACK [TOP] := ITEM. [Inserts ITEM in new TOP position.]
4. Return.

10) Write an algorithm for POP operation using arrays.

POP (STACK, TOP, ITEM)

This procedure deletes the top element of STACK and assigns it to the variable ITEM.

1. [Stack has an item to be removed?]
   If TOP = 0, then: Print: UNDERFLOW, and Return.
2. Set ITEM := STACK[TOP]. [Assigns TOP element to ITEM.]
3. Set TOP := TOP - 1. [Decreases TOP by 1.]
4. Return.

11) Define i) infix notation ii) prefix notation iii) postfix notation. (any 2)

INFIX: It is the form of an arithmetic expression in which we fix place the arithmetic operator in between the two operands.

Example: Let A and B are two operands then the infix form of expression for arithmetic operators +,-, *and/are given below. A+B, A– B,A * B, A / B

POSTFIX: It is the form of an arithmetic expression in which we fix (place) the arithmetic operator after its two operands.

the postfix expression **ABC*+**

PREFIX: It is the form of an arithmetic expression, in which we fix(place) the arithmetic operator before (pre) its two operands.

+*AB/CD.

12) What do you mean by infix notation? Give an example.

Infix notation: X + Y. Operators **are** written in-between their operands. This is the usual way **we** write expressions. An **expression** such as A * (B + C) / **D** is usually taken to **mean** something like: "First add B and C together, then multiply the result by A, then divide by **D** to **give** the final answer."

13) What do you mean by prefix notation? Give an example.

In this **notation**, operator is prefixed to operands, i.e. operator is written ahead of operands. For **example**, +ab. This is equivalent to its infix **notation** a + b. **Prefix notation** is also known as **Polish Notation**.

14) What do you mean by postfix notation? Give an example.

In a **postfix expression**, • an operator is written after its operands. • the infix **expression** 2+3 is 23+ in **postfix notation**. • For **postfix** expressions, operations **are** performed in the order in which **they are** written (left to right).

15) What do you mean by polish and reverse polish notation?

Polish notation is a mathematical notation in which operators *precede* their operands, in contrast to the more common infix notation, in which operators are placed *between* operands, as well as reverse Polish notation (RPN), in which operators *follow* their operands.

Reverse Polish notation (**RPN**), also known as **Polish postfix notation** is a mathematical notation in which operators *follow* their operands, in contrast to Polish notation (PN), in which operators *precede* their operands. It does not need any parentheses as long as each operator has a fixed number of operands.

16. Write an algorithm to find factorial of a number N using recursion.

FACTORIAL(FACT, N)

This procedure calculates N! and returns the value in the variable FACT.

1. If N= 0, then: Set FACT := 1, and Return.
2. Call FACTORIAL(FACT, N - 1).
3. Set FACT := N*FACT.
4. Return

17. Write an algorithm to generate Fibonacci series using recursion.

FINONACCI(FIB,N)

This procedure calculate Fn and returns the value in first parameter FIB

1. If N=0 OR N=1 then

   SET FIB:=N and Return;

   [End of If]
2. CALL FIBONACCI(FIBA,N-2)
3. CALL FIBONACCI(FIBB,N-1)
4. SET FIB:=FIBA+FIBB
5. Return

23) Define queue. Name the primary queue operations.

A queue is a linear list of elements in which deletions can take place only at one end, called the front, and insertions can take place only at the other end, called the rear.

24) Differentiate queue and circular queue.

S.no.	Linear Queue	Circular Queue
1.	Arranges the data in a linear pattern.	Arranges the data in a circular order where the rear end is connected with the front end.
2.	The insertion and deletion operations are fixed i.e, done at the rear and front end respectively.	Insertion and deletion are not fixed and it can be done in any position.
3.	Linear queue requires more memory space.	It requires less memory space.
4.	In the case of a linear queue, the element added in the first position is going to be deleted in the first position. The order of operations performed on any element is fixed i.e, FIFO.	In the case of circular queue, the order of operations performed on an element may change.
5.	It is inefficient in comparison to a circular queue.	It is more efficient in comparison to linear queue.

25) What is circular queue?

A circular queue is similar to a linear queue as it is also based on the FIFO (First In First Out) principle except that the last position is connected to the first position in a circular queue that forms a circle.

26) Define dequeue. What are its types?

A dequeue is a linear list in which elements can be added or removed at either end but not in the middle. The term dequeue is a contraction of the name double-ended queue.

There are two variations of a deque-namely, an input-restricted deque and an output-restricted deque

27) What do you mean by input-restricted deque and output-restricted deque.

An **input-restricted deque** is a deque which allows insertions at only one end of the list but allows deletions at both ends of the list; and

An **output-restricted deque** is a deque which allows deletions at only one end of the list but allows insertions at both ends of the list.

28) What is meant by priority queue? What

A priority queue is a collection of elements such that each element has been assigned a priority and such that the order in which elements are deleted and processed comes from the following rules:

1. An element of higher priority is processed before any element of lower priority.
2. Two elements with the same priority are processed according to the order in which they were added to the queue .

18. Write any 2 applications of queues.

- Categorizing data: This is useful in many problems.
- Queue simulation: This can be used to study the performance of any queue application.

Unit -4

1. **Define binary tree. Give an example.**

A binary tree is a finite set of data items which is either empty or con ist of a single item called the root and two disjoint binary trees called the **left subtree** and **right subtree**.

3) Define complete binary tree. Give an example.

A binary tree with n nodes and of depth d is a strictly binary tree all of whose terminal nodes are at level d. In a complete binary tree. There is exactly one node at level 0, two nodes at level 1, and four nodes at level 2 and so on.

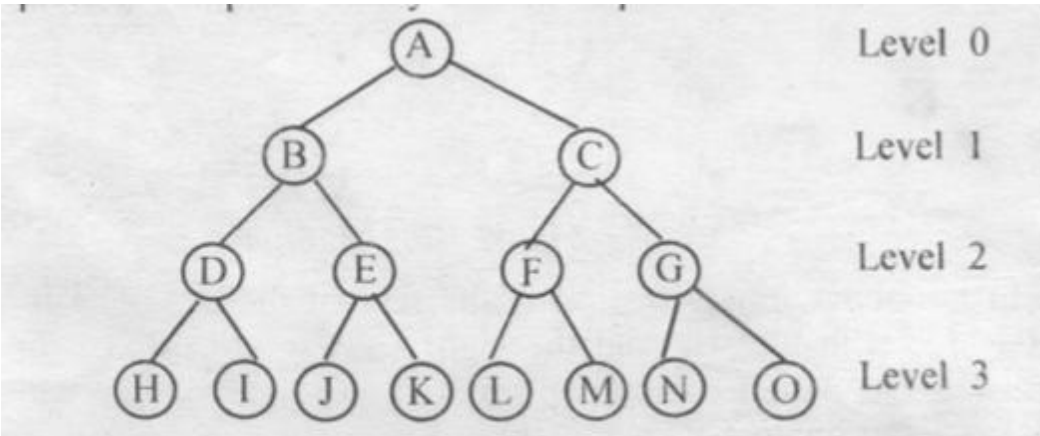
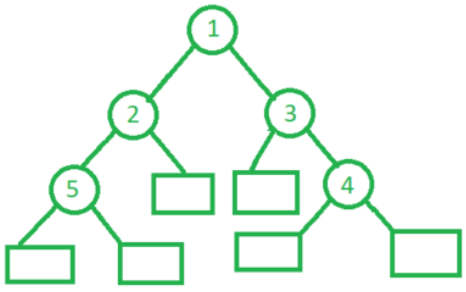


Figure 10.4 A complete binary tree with depth 4

4) What is extended binary tree? Give an example.

Extended binary tree is a type of binary tree in which all the null sub tree of the original tree are replaced with special nodes called **external nodes** whereas other nodes are called **internal nodes**



Extended Binary Tree

5) What do you mean by internal nodes and external nodes?

An **internal node** (also known as an inner **node**, inode for short, or branch **node**) is any **node** of a tree that has child **nodes**. Similarly, an **external node** (also known as an **outer node**, leaf **node**, or terminal **node**) is any **node** that **does** not have child **nodes**.

6) Name the 2 ways of representing trees in memory.

Binary trees can be represented either using an ***array representation*** or using ***a linked list representation***.

Array representation of Binary trees An array can be used to store the nodes of a binary tree. The nodes stored in an array are accessible sequentially. In C, arrays start with index 0 to (MAXSIZE - 1). Here, numbering of binary tree nodes starts from 0 rather than 1. The maximum number of nodes is specified by MAXSIZE.

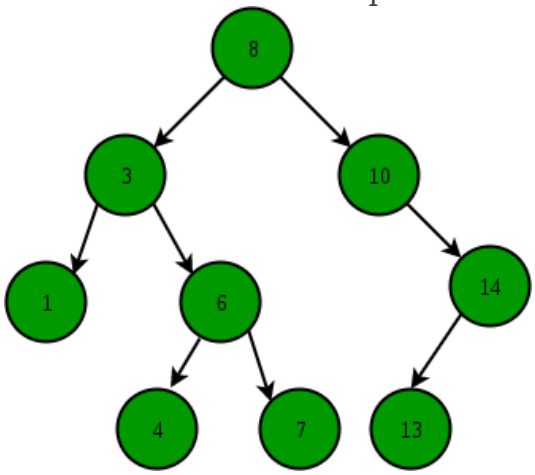
The array representation of this binary tree is as follows.

	BT
BT [0]	A
BT [1]	B
BT [2]	C

10) What is a binary search tree? Give an example.

Binary Search Tree is a node-based binary tree data structure which has the following properties:

- The left subtree of a node contains only nodes with keys lesser than the node's key.
- The right subtree of a node contains only nodes with keys greater than the node's key.
- The left and right subtree each must also be a binary search tree. There must be no duplicate nodes.



11) Give any two applications of binary tree.

Binary Search Tree - Used in many search applications that constantly show and hide data, such as data. For example, map and set objects in many libraries.

B-Tree and B+ Tree : They are used to implement indexing in databases.
Suffix Tree : For quick pattern searching in a fixed text.

13) Define adjacency matrix.

an **adjacency matrix** is a square **matrix** used to represent a finite graph. The elements of the **matrix** indicate whether pairs of vertices are **adjacent** or not in the graph. ... If the graph is undirected (i.e. all of its edges are bidirectional), the **adjacency matrix** is symmetric.

14) Define path matrix.

The concept of **path matrix** of G if G is a simple directed graph. A graph consists of a set of nodes which are also known as vertices and a set of arcs also known as edges. Each graph is specified by a pair of nodes. If the pairs of nodes that make up the arcs are ordered pairs, the graph is said to be a directed graph.

15) List various operation performed on graphs.

Following are basic primary operations of a Graph −

- **Add Vertex** − Adds a vertex to the graph.
- **Add Edge** − Adds an edge between the two vertices of the graph.
- **Display Vertex** − Displays a vertex of the graph.