Subject Name: Data Structures using C++ Class : II CSE

Subject Code: CS630202 Semester :III

UNIT 1: INTRODUCTION TO OOPS AND C++			
Object oriented programming paradigm – Basic concepts of object oriented programming – Benefits of OOP – A simple C++ program - Basics of C++: Tokens – Keywords – Identifiers and constants – Basic data types - Declaration of variables – Operators in C++ – Scope resolution operator – Operator precedence.			
BL –Bloom's Level (1-Remembering, 2-Understanding, 3–Applying, 4–Analyzing, 5–Evaluating, 6- Creating)			
Part–A(Two Marks)			
Q.No	Question	BT Level*	Competence*
1	What is the need for the object oriented programming paradigm? Object-oriented programming paradigm methods enable us to create a set of objects that work together to produce software that is better understandable and models their problem domains than produced using traditional techniques.	2	Understanding
2	Define Class. The entire set of data and code of an object can be made a user defined data type with the help of a class. Objects are variables of the type class.Thus a class is a collection of objects of similar type. Eg: fruit mango; where fruit is the class name and mango is the object name	1	Remembering
3	Define Object. The instances of class is called as objects. Objects are the basic run time entities in an object oriented system. with each other by passing message. Each object contains data and code to manipulate the data. Each object contains data and code to manipulate the data	1	Remembering

4	What is Data Abstraction and Encapsulation? The wrapping up of data and functions into a single unit (called class) is known as encapsulation. Abstraction refers to the act of representing the essential features without including the background details such as size, weight and cost. The variables are called as data members and the function is called as methods or member function.	2	Understanding

5	How could you define Inheritance? The process of forming a new class from an existing class is known as inheritance. The newly created class is called as derived class The old class is called as base class. It reduces code size It provides the idea of reusability	2	Understanding
6	What is Polymorphism and its types? It is derived from a Greek word which means the ability to take more than one form. It is classified into two types. • Compile Time Polymorphism ✔ Operator Overloading ✔ Function Overloading • Runtime Polymorphism	1	Remembering
7	Define Binding. Binding refers to the linking of a procedure call to the code to be executed in response to the function call. • Dynamic Binding or late binding means that the code to be used for a function call is not s related to inheritance and polymorphism	2	Understanding
8	What are specifying parameters for Message Passing? Message passing involves specifying • Object name • Function name • Information to be sent.	1	Remembering
9	Define Basics of C++. C++ is a middle-level programming language developed by Bjarne Stroustrup starting in 1979 at Bell Labs. C++ runs on a variety of platforms, such as Windows, Mac OS, and the various versions of UNIX.	2	Understanding

10	Write a Simple C++ Program.
	#include <iostream> using namespace std;
	
	// main() is where program execution
	begins. int main()
	{ cout << "Hello World"; // prints Hello World
	return 0; }
	Output: Hello World
		2	Understanding

known until the time of the call at run time. It i

11	Define Tokens in C++. Tokens are the smallest individual units of a program. C++ is the superset of C and so most constructs of C are legal in C++ with their meaning and usage unchanged. So tokens, expressions, and data types are similar to that of C.	1	Remembering
12	Mention C++ Tokens? - Keywords - Identifiers - Constants - Variables - Operators	2	Understanding
13	Write the Syntax of Variable? A variable is a meaningful name of data storage location in computer memory. When using a variable you refer to memory address of computer. Syntax to declare a variable: [data_type] [variable_name];	1	Remembering
14	What are Basic Datatypes of C++? - Basic Types - Enumerated types - The type void - Derived types	2	Understanding
15	What are the three kinds of situations in Void type? - Function returns as void - Function arguments as void - Pointers to void	2	Understanding
16	Define Operator. C++ operator is a symbol that is used to perform mathematical or logical manipulations	2	Understanding
17	What are the Types of Operators in C++? - Arithmetic Operators, - Increment and Decrement Operators, - Relational Operators, - Logical Operators, - Bitwise Operators, - Assignment Operators, - Misc Operators	2	Understanding

18	Define Scope Resolution Opeartor. The scope resolution operator (::) is used for several reasons. For example: If the global variable name is same as local variable name, the scope resolution operator will be used to call the global variable. It is also used to define a function outside the class and used to access the static variables of class.	2	Understandi n g
19	What is Opeartor Precedence? Operator precedence determines the grouping of terms in an expression. The associativity of an operator is a property that determines how operators of the same precedence are grouped in the absence of parentheses. This affects how an expression is evaluated. Certain operators have higher precedence than others; for example, the multiplication operator has higher precedence than the addition operator:	1	Remembering
20	Mention some Bitwise Operators? OperatorDescription << Binary Left Shift Operator != Is not equal to >> Binary Right Shift Operator ~ Binary One's Complement Operator	2	Understandi n g

Part–B (13 Marks)

Q. No	Question	BT Level*	Competence*
1	Explain in detail about Basic concepts of OOPs?	2	Understanding
2	Discuss briefly about Data Types used in C++?	3	Applying
3	Illustrate the concept of Opertors used in C++?	3	Applying
4	Expalin in detail about Various types of Operators in C++?	2	Understanding
5	Outline the details of Operator Precedence with neat example?	3	Applying
6	Discuss briefly about Scope Resolution Operators?	3	Applying
7	Write a Simple C++ Program with tokens and identifiers?	2	Understanding

8	Explain in detail about Tokens used in C++?	3	Applying

UNIT 2: CONTROL STRUCTURES AND STATEMENTS IN C++			
Control structures in C++: if statement – switch statement – do-while statement – while statement – else statement - for statement - Functions in C++: Introduction - The main function – Function prototyping – Call by reference – Return by reference – Inline functions - Method overloading – friend and virtual functions – Math library functions.			
Part–A(Two Marks)			
Q. No	Question	BT Level*	Competence*
1	Define Selection Structure. Selection structure refers to the execution of instruction according to the selected condition, which can be either true or false. There are two ways to implement selection structures, by "ifelse statements" or by "switch case statements".	2	Understanding
2	Define Loop Structure. Loop structure refers to the execution of an instruction in a loop until the condition gets false	2	Understanding
3	Define Sequence Structure. Sequence structure refers to the sequence in which program execute instructions one after another.	2	Understanding
4	Mention the types of Conditional Statements in C++? □ if Statement □ if-else Statement □ switch statement	1	Remembering
5	Mention the types of Loop Statements in C++? □ while Loop □ do – while statement □ for loop □ break □ continue	1	Remembering

6	Define Function in C++. A function is a set of statements that take inputs, do some specific computation, and produce output. The idea is to put some commonly or repeatedly done tasks together and make a function so that instead of writing the same code again and again for different inputs, we can call	2	Understanding

	the function. In simple terms, a function is a block of code that only runs when it is called		
7	What is Need for functions? • Modularity and Clarity: Functions helps to decompose the large program into small segments which makes programmer easy to understand, maintain and debug. • Reusability: If repeated code occurs in a program. Function can be used to include those codes and execute when needed by calling that function. • Portability: programs can be written that are system independent. • Programmer working on large project can divide the workload by making different functions.	4	Analyzing
8	List out the types of functions? • Library function: Library functions are the in-built function in C programming system. For example: strcat() • User defined function: C allows programmer to define their own function according to their requirement. These types of functions are known as user defined functions. For example: sum().	2	Understanding
9	What are the Elements of user defined function? • Function Declaration • Function Definition • Function Call	1	Remembering

10	Define Arguments. An argument is referred to the values that are passed within a function when the function is called. These values are generally the source of the function that require the arguments during the process of execution. These values are assigned to the variables in the definition of the function that is called. The type of the values passed in the function is the same as that of the variables defined in the function definition. These are also called Actual arguments or Actual Parameters.	2	Understanding

11	Define Parameters. The parameter is referred to as the variables that are defined during a function declaration or definition. These variables are used to receive the arguments that are passed during a function call. These parameters within the function prototype are used during the execution of the function for which it is defined. These are also called Formal arguments or Formal Parameters.	2	Understanding
12	Write the syntax of return statement? A function may or may not return a value. A return statement returns a value to the calling function and assigns to the variable in the left side of the calling function. If a function does not return a value, the return type in the function definition and declaration is specified as void. Syntax: return ; OR return (expr/value);	1	Remembering
13	Define Call by Value. This method copies the actual value of an argument into the formal parameter of the function. In this case, changes made to the parameter inside the function have no effect on the argument.	2	Understanding
14	Define Call by Reference. This method copies the reference of an argument into the formal parameter. Inside the function, the reference is used to access the actual argument used in the call. This means that changes made to the parameter affect the argument.	2	Understanding
15	Define Inline function. □ Member functions are the functions that are declared inside the class definition and works on the data members of the class. □ If the function is defined inside the class definition it is called as inline function which can be defined directly.	2	Understanding

16	What is Function overloading? Function overloading is a concept in programming languages like C++ where programmers can define multiple functions with the same name but different parameter lists. These functions can have the same name, but they must have different types or numbers of parameters.	1	Remembering

17	What is friend Function in C++? • If a function is defined as a friend function then, the private and protected data of a class can be accessed using the function. • The complier knows a given function is a friend function by the use of the keyword friend.	1	Remembering
18	Define Virtual Function. • When the same function name is used in both base class and derived class the function in the base is declared as virtual • A virtual function is a member function in a base class that is declared using the keyword virtual.	2	Understanding
19	List out the Properties of virtual functions? • Virtual functions must be member of some class. • They cannot be static members and they are accessed by using object pointers. • Virtual function in a base class must be defined. • Prototypes of base class version of a virtual function and all the derived class versions must be identical. • If a virtual function is defined in the base class, it need not be redefined in the derived class • Virtual constructors cannot be defined but Virtual destructors can be defined	3	Applying

20	What is Pure Virtual Function?
	A pure virtual function is a function declared in a base class but has no definition (i.e., virtual function with no body). The declaration is assigned the value of 0. A class containing a pure virtual function is called abstract base class.
	The function should be redefined in the derived class. class shape
	{
	public:
	void calc_area()=0;
	}
	;
		1	Remembering

Part–B(16 Marks)			
1	Explain in detail about Control Structures in C++?	3	Applying
2	Illustrate the concept of switch statement – do statement with example?	3	Applying

3	Discuss briefly about Function prototyping in C++?	2	Understanding

4	Explain the concept of Call by reference & return by reference with neat example?	3	Applying
5	Outline the concept of else statement – for statement with example?	3	Applying
6	Discuss briefly about Method overloading in C++?	2	Understanding
7	Explain in detail about Math library functions used in C++?	4	Analyzing
8	Explain in detail about friend and virtual functions used in C++ with neat example?	2	Understanding

UNIT 3: BASIC OOPS CONCEPTS IN C++

Specifying a class – Defining member functions – A C++ program with class – Parameterized constructors – Multiple constructors in a class – Constructors with default arguments – Copy constructor – Destructors - Operator overloading – Inheritance - Single inheritance – Multilevel inheritance – Multiple inheritance – Hierarchical inheritance – Hybrid inheritance.

Part–A(Two Marks)

Q. No	Question	BTLevel*	Competence*
1	Define Class. A class is a blueprint or template for creating objects. It defines the properties (attributes or data members) and behaviors (methods or member functions) that the objects of that class will have.	2	Understanding

2	Write Syntax to define a class in C++?
	class ClassName {
	access_specifier:
	// Data members (attributes)
	dataType attributeName;
	// Member functions (methods)
	returnType methodName(parameters);
	// ...
	};
		3	Applying

3	Define Object. An object is an instance of a class. It represents a specific entity or data structure created from the class's blueprint. Each object has its own set of attributes and can invoke the class's methods	2	Understanding

4	Write a Syntax to create an object in C++?
	ClassName objectName;
	int main() {
	// Creating objects of the Person class
	Person person1;
	Person person2;
	// Setting attributes of the objects
	person1.name = "Alice";
	person1.age = 30;
	person2.name = "Bob";
		3	Applying

	person2.age = 25;
	// Calling the method on objects
	person1.displayInfo(); //
	Output: Name: Alice, Age: 30
	person2.displayInfo(); // Output: Name: Bob, Age: 25
	return 0;
	}		

5	Define Member Functions.
	- Member functions are the functions that are declared inside the class definition and works on the data members of the class.
	- A member function can be defined in two way
	  ✔ Inside the class
	  ✔ Outside the class
	- If the function is defined inside the class definition it is called as inline function which can be defined directly.
		2	Understanding

6	Define Public members: **Public members** are accessible from anywhere outside the class but within the main program.	2	Understanding
7	Define Private members: - **Private members** cannot be accessed or even cannot view from outside the class. - Only the members of the class can be accessed. - By default all the members of a class would be private.	2	Understanding
8	What is Static Data Member? - A variable which is a part of the class but not a part of an object is called as a static member. - When a data member is declared as static then only one copy of the data is maintainer for all objects of the class. It is automatically initialized to 0, when the first object of the class is created. - It must be initialized always outside the class using scope resolution operator.	1	Remembering
9	What is Static Member Function? - A function that needs access to members of a class, without invoking for a particular object is defined as static member function - It can be declared by prefixing it with the keyword **static**. **Syntax:** static data_type variable_name;	1	Remembering

10	Mention the Properties of static member function? A static function has the following properties • A static function can access only static members or static function declared in the same Class • A static function will be called using the class name as follows:	3	Applying

	class_name::function_name(); - A static function is common to the entire class		
11	Define Constructor. - It is a special member function - The constructors task is to initialize the objects of its class. - It is special because its name is the name as the class name. - It is invoked when an object is created.	2	Understanding
12	List out the rules for defining a constructor? - The rules to be followed - It should be in public section - It should have the same name as that of the class name - It has no return type - It cannot be virtual - It cannot be inherited	3	Applying
13	Mention the Type of constructor? - Default Constructor - Parameterized Constructor - Copy Constructor	1	Remembering

14	Write the Syntax of Default Constructor?
	- A constructor with no arguments is called as default constructor.
	- It is mainly used to initialize the variables to a default value.
	- It is called when an object is created with no arguments.
	Syntax:
	class_name()
	{
	Constructor definition
	}
		3	Applying

15	Write the Syntax of Parameterized Constructor?
	- A constructor with arguments is called as parameterized constructor.
	- It is invoked when an object is created with matching arguments.
	Syntax:
	class_name(argument)
	{
	Constructor definition
	}
		3	Applying

16	Write the Syntax of Copy Constructor?
	- It is used to declare and initialize an objects from another object.
	Syntax:
	class_name(class_name &t)
	{
	Constructor definition
	}
		3	Applying

17	Define Destructor. • A destructor is used to destroy the objects that have been created by a constructor • It is also a member function whose name is the same as the class name but is preceded by • a tilde(~) operator. • It is invoked when an object is destroyed. • It cannot be overloaded. • It has no return type and no arguments	2	Understanding
18	How could you define this pointer in C++? • C++ uses a unique keyword called **this** to represent an object that invokes a member function. • This is a pointer that points to the object for which this was called. • This unique pointer is automatically passed to a member function when it is called. • It act as an implicit argument to all the member functions.	1	Remembering
19	Define Operator Overloading. • C++ has the ability to provide the operators with a special meaning for a data type. • The mechanism of giving such special meanings to an operator is known as operator overloading. • To define an additional task to an operator, what it means in relation to the class to which the operator is applied must be specified. This is done with the help of a special function which describes the task.	2	Understanding
20	List out the Types of Inheritance? • Single Inheritance • Multi-level Inheritance • Multiple Inheritance • Hierarchical Inheritance • Hybrid Inheritance	1	Remembering
	Part–B(16 Marks)		
1	Write a Simple C++ program with class concept?	3	Applying
2	Discuss briefly about Parameterized constructors with neat example?	3	Applying
3	Explain in detail about Multiple constructors in a class with neat example?	3	Applying
4	Illustrate the concept of Constructors with default Arguments with neat example?	4	Analyzing
5	Discuss in detail about Operator overloading in C++?	3	Applying

6	Explain in detail about Types of Inheritance in C++?	2	Understanding

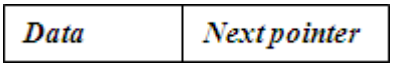
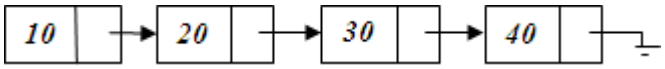
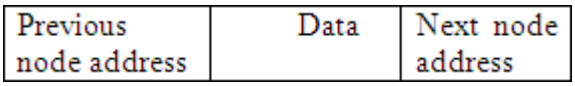
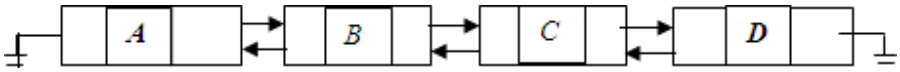
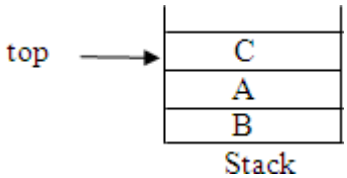
7	Outline the concept of Hierarchical inheritance & Hybrid inheritance with neat example?	2	Understanding
8	Discuss shortly about Copy constructor & Destructors with neat example?	2	Understanding

UNIT 4: BASIC DATA STRUCTURES AND SORTING

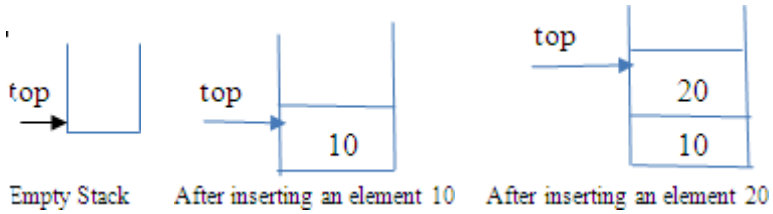
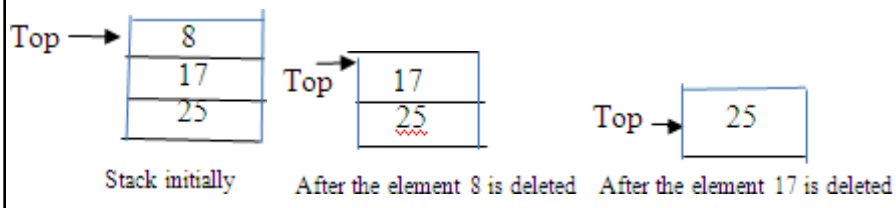
Algorithm – Analysis – List ADT – Stack ADT – Queue ADT – Priority Queue – Stack implementation – Basic operations on stack – Application of stack – Queue: Introduction – Definition of Queue –implementation of Queue – Operation on a Queue – Applications of Queue – Sorting : bubble sort-Insertion sort - Merge sort - Quick sort-Searching - hashing

Part–A(2 Marks)

Q.No	Question	BTLevel*	Competence*
1	Define Data structures & Classify it: A data structure is a systematic way of organizing and accessing data. **Classification of Data structure:** **Linear Data structure**: A data structure is said to be linear In which the data items are stored in sequence order. Example: Array, Linked list, Stacks, Queues. **Nonlinear Data structure**: A data structure is said to be nonlinear if its element are not in sequence Example: Tree, Graph	B1	Remember
2	What is Abstract Data type? An **Abstract data type** is defined as a mathematical model of the object that makeup a data type as well as the functions that operate on these objects. An abstract data type is a set of operations Object such as lists, sets and graph along with their operations can be viewed as abstract data types.	B1	Remember
3	Define List. **Li**st is an ordered set of elements or List is a linear collection of data items. The general form of the list is of the formA1, A2, A3, ,AN A1 - First element of the list AN - Last element of the list N - Size of the list If the element at position i is Ai then its successor is Ai+1 and its predecessor is Ai-1.Empty list is of size 0.	B1	Remember
4	What are the various operations performed on List? 1. Insert (X, 5) - Insert the element X after the position 5. 2. Delete (X) - The element X is deleted 3. Find (X) - Returns the position of X. 4. Next (i) - Returns the position of its successor element i+1. 5. Previous (i) - Returns the position of its predecessor i-1. 6. Print list - Contents of the list is displayed. 7. Makeempty - Makes the list empty.	B2	Understand
5	Define Singly Linked List. . How to represent it? A singly linked list is a linked list , a node is connected to the next node by a single link. A node in this type of linked list contains two types of fields Data: It hold a list of element	B1	Remember

	Next: It stores a link to the next node in the list Representation of a node  Representation of a singly linked list  		
6	Define Doubly Linked List. How to represent it? A Doubly linked list is a two way linked list uses double set of pointers, one pointing to the next item and other pointing to the preceding item. It can be traversed in two direction either from the beginning of the list or in the backward direction from end of the list to the beginning. Representation of a node  Each node contains 3 parts An information field which contains the data A pointer field previous which contains the location of the preceding node in the list A pointer field next which contains the location of the next node in the list Representation of a Doubly linked list  	B1	Remember
7	What can you point out about Stack ADT A stack is a linear data structure which follows Last In First Out (LIFO) principle, in which both insertion and deletion occur at only one end of the list called the **top**.  BASIC OPERATIONS ON STACK 1. Push (for insertion) 2. Pop (for deletion)	B4	Analyze

8	What can you point out PUSH operation? The process of inserting a new element to the top of the stack. For every push operation the top is incremented by 1.	B4	

	Eg.  		Analyze
9	What can you point out POP operation. The process of deleting an element from the top of stack is called pop operation. After every pop operation the top pointer is decremented by 1.  	B4	Analyze
10	What can you point out Prefix notation(Polish notation), Infix notation&Postfix notation(Reverse polish notation) **Prefix notation(Polish notation):** The arithmetic operator is placed before the two operands to which it applies. Also called as polish notation. ((A/B) + C) For example : - +/ABC In **Infix notation**, The arithmetic operator appears between the two operands to which it is being applied. For example : A / B + C **Postfix notation(Reverse polish notation):** The arithmetic operator appears directly after the two operands to which it applies. Also called reverse polish notation. For example : AB / C +	B4	Analyze

11	What can you point out Queue ADT? A Queue is a linear data structure which follows First In First Out (FIFO) principle, in which insertion is performed at rear end and deletion is performed at front end. Example : Waiting Line in Reservation Counter Dequeue (Q) Queue Q Enqueue(Q) **Operations on Queue:** The fundamental operations performed on queue are 1. Enqueue 2. Dequeue	B4	Analyze



	Enqueue :The process of inserting an element in the queue. Dequeue :The process of deleting an element from the queue.		
12	Define Priority Queues& List out its Applications . Priority Queue is a Queue in which inserting an item or removing an item can be performed from any position based on some priority. **Applications of Queue:** * Batch processing in an operating system * To implement Priority Queues. * Priority Queues can be used to sort the elements using Heap Sort. * Simulation. * Mathematics user Queueing theory. * Computer networks where the server takes the jobs of the client as per the queue Strategy.	B1	Remember
13	What can you point out Time complexity & Space Complexity of Bubble sort, Insertion Sort, Merge sort & Quick sort. Time Complexity of **Bubble Sort**: $O(n^2)$ in the worst and average cases, O(n) in the best case (when the input array is already sorted) Space complexity: O(1) Time Complexity of **Insertion Sort**: $O(n^2)$ in the worst and average cases, O(n) in the best case (when the input array is already sorted) Space complexity: O(1) Time Complexity of **Merge sort**: Best case , Average case, Worst case : O (n log n) Space Complexity : O(n) Time Complexity of **Quick sort**: Best case & Average cases: O (n log n) Worst case : O (n2) Space Complexity : O(n log n)	B4	Analyze
14	Define Sorting& its types **Sorting** is a technique to rearrange the elements of a list in ascending or descending order, whichcan be numerical, lexicographical, or any user-defined order. Sorting is a process through whichthedataisarranged in ascendingor descendingorder.Sortingcanbeclassifiedintwo types: 1. **Internal Sorts** - uses only the primary memory during sorting process. All dataitems are held in main memory and no secondary memory is required this sorting process -3 types ofinternalsorts. SELECTIONSORT :- Ex:-Selectionsort,HeapSort INSERTIONSORT:-Ex:-Insertionsort ,Shell Sort EXCHANGESORT:- Ex:-BubbleSort,Quicksort 2. **External Sorts** - Sorting large amount of data requires external or secondary memory.Ex:- Merge Sort	B1	Remember
15	Define Merge Sort.	B1	

	Merge sortis based on the divide-and-conquer paradigm. The basic concept of merge sort is divides the list into two smaller sub-lists of approximately equal size. Recursively repeat this procedure till only one element is left in the sub-list. After this, various sorted sub-lists are merged to form sorted parent list. This process goes on recursively till the original sorted list arrived.		Remember
16	Define Quick Sort. **Quick sort** is based on partition. It is also known as partition exchange sorting. The basic concept of quick sort process is pick one element from an array and rearranges the remaining elements around it. This element divides the main list into two sub lists. This chosen element is called pivot. Once pivot is chosen, then it shifts all the elements less than pivot to left of value pivot and all the elements greater than pivot are shifted to the right side. This procedure of choosing pivot and partition the list is applied recursively until sub-lists consisting of only one element.	B1	Remember
17	Define Insertion Sort. **Insertion Sort:** Both the selection and bubble sorts exchange elements. But insertion sort does not exchange elements. In insertion sort the element is inserted at an appropriate place similar to card insertion. Here the list is divided into two parts sorted and unsorted sub-lists. In each pass, the first element of unsorted sub list is picked up and moved into the sorted sub list by inserting it in suitable position	B1	Remember
18	Define Bubble Sort. In **Bubble sort** method the list is divided into two sub-lists sorted and unsorted. The smallest element is bubbled from unsorted sub-list. After moving the smallest element the imaginary wall moves one element ahead. The bubble sort was originally written to bubble up the highest element in the list. But there is no difference whether highest / lowest element is bubbled. This method is easy to understand but time consuming. In this type, two successive elements are compared and swapping is done. Thus, step-by-step entire array elements are checked. Given a list of 'n' elements the bubble sort requires up to n-1 passes to sort the data	B1	Remember
19	List out the different categories of Searching **Linear Search** A linear search scans one item at a time, without jumping to any item. **Binary Search** A binary search however, cut down your search to half as soon as you find middle of a sorted list. The middle element is looked to check if it is greater than or less than the value to be searched. Accordingly, search is done to either half of the given list	B1	Remember
20	What is Hashing? **Hashing** is a process which uses a function to get the key and using the key it quickly identifies the record, without much strain. The values returned by a hash function are called hash values. Hash table is data structure in which key values are place in array location	B1	Remember
Part–B(13/15 Marks)			

1	Formulate an algorithm to insert an element and to delete an element in a Singly linked list?	B6	Create

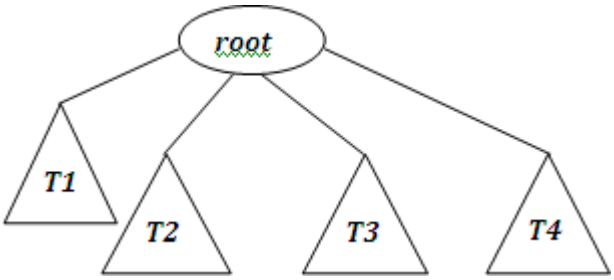
2	Explain with an algorithm to insert an element and to delete an element in a Doubly linked list?	B6	Create
3	Formulate an algorithm to insert an element and to delete an element in a Queue?	B6	Create
4	Discuss the algorithms for push and pop operations on a Stack	B6	Create
5	Why do you think about any three application of stacks?	B4	Analyze
6	Write an algorithm for converting infix expression to postfix expression with an example of (A+(B*C-(D/E^F)*G)*H)	B5	Evaluate
7	Explain the Merge Sort with example	B4	Analyze
8	Explain the Bubble Sort with example	B4	Analyze
9	Explain the Insertion Sort with example	B4	Analyze
10	Explain the Quick Sort with example	B4	Analyze

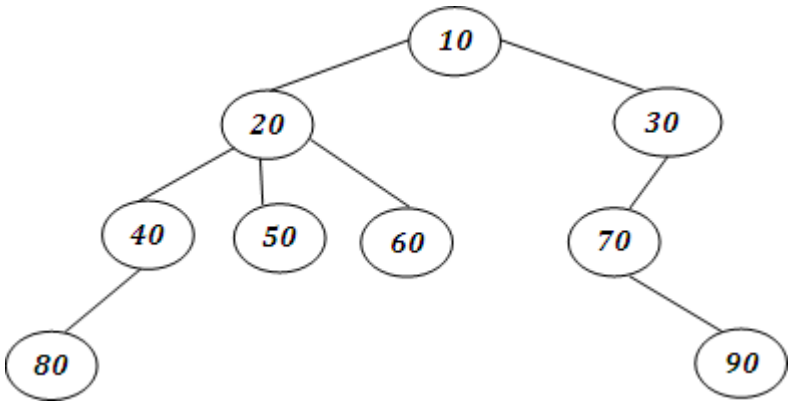
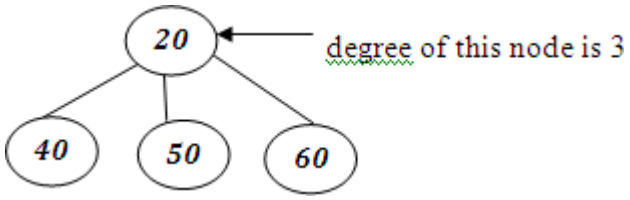
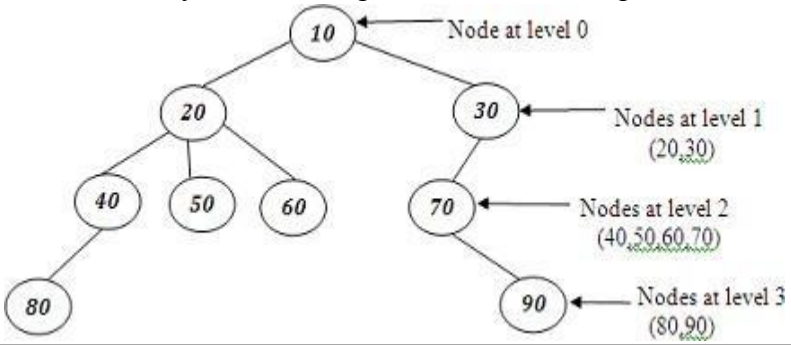
UNIT 5: GRAPH AND TREES

Trees: Introduction – Tree – Basic elements of a tree – Binary tree – Representation of binary tree – Operations on binary tree – AVL Tree – Operations on AVL Tree - Graph: Representation - Shortest path algorithm: Dijikstra's algorithm - Minimum spanning tree: Prim's Algorithm

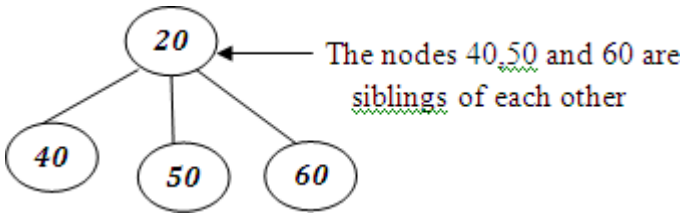
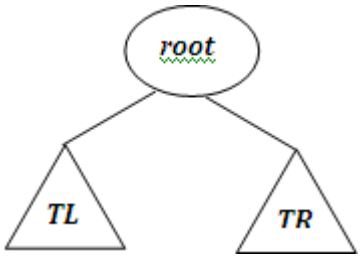
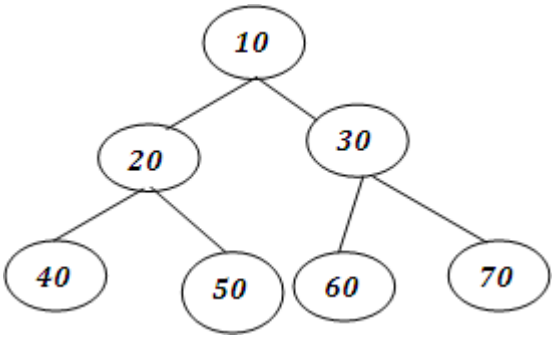
Part–A(Two Marks)

Q.No	Question	BTLevel*	Competence*
1	What is the need for non-linear data structure? • Linear data structures such as stacks, queues, linked list and arrays. • In linear data structure all the elements are arranged in a linear sequence. • But there some complex applications for which use of linear data structure is not possible • Example trees,graphs, multi-link polynomials are some data structure which are hard to represent using linear data structure. • Hence there is a need for nonlinear data structure.	B2	Understand

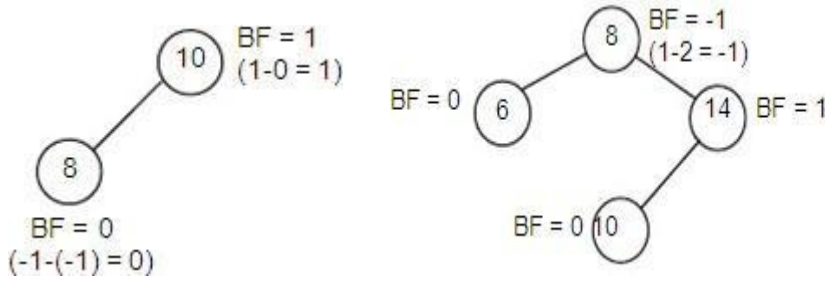
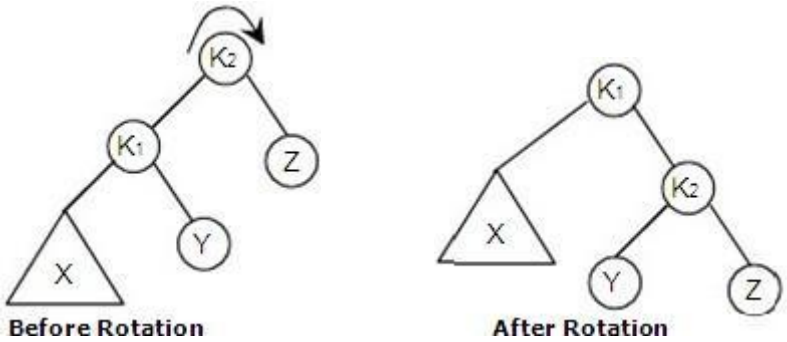
2	What is your opinion about Tree? A tree is a collection of nodes. The tree consists of a distinguished nodes R called the root, and zero or more non empty sub trees $T_1, T_2 \ldots\ldots T_n$, each of whose roots are connected by a directed edge from root R.  Example:	B4	Analyze

	 *Root* • The node at the top of the tree (or) the node that has no parent is called the root. • In above Fig : node 10 is a root node. • *Parent node* • The node having further sub-branches is called parent node. • In above Fig : node 20 is the parent node of 40,50,60. *Leaf node* • Nodes with no children are called leaves In above Fig : 80,50,60,90 are the leaf nodes		
3	What is your opinion about degree of the node? • The total number of subtrees attached to that node is called the **degree of a node**. When the branch is directed toward a node, it is an indegree branch; when the branch is directed away from the node, it is an outdegree branch. The sum of indegree and outdegree branches is the degree of the node.The indegree of the root is by definition is zero.  	B4	Analyze
4	What is your opinion about level of the tree? • The root node is always considered at level zero. • The level of any node is one plus the level of the parent.  	B4	Analyze

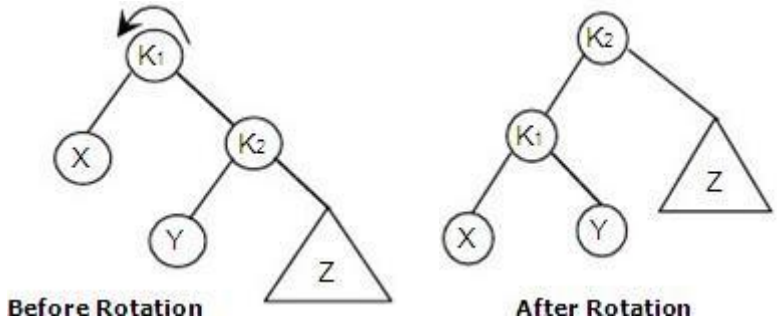
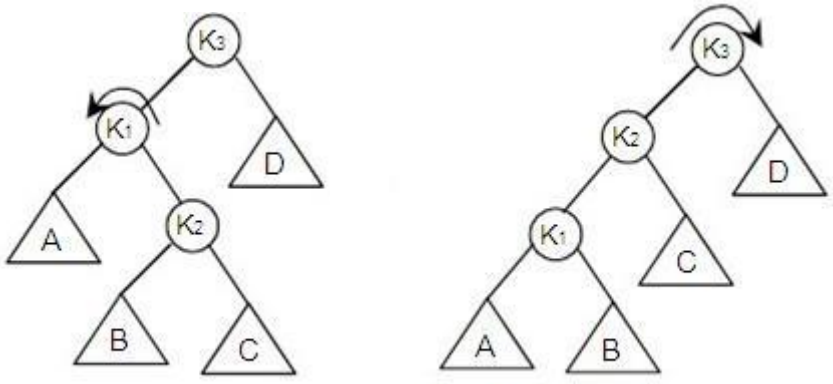
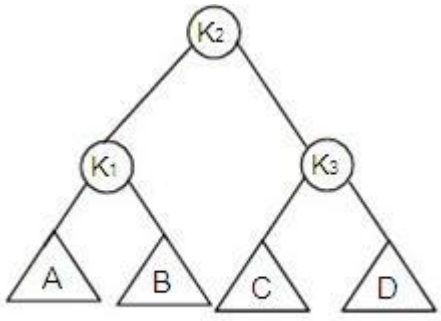
5	Define Height of the tree & Height of the node. Height of the tree • The maximum level is the height of the tree. • In above Fig: the height of the tree is 3		

	Sometimes height of the tree is also called depth of tree. Height of the node The height of node is the length of the longest path from node to a leaf. All the leaves are at height 0. Height(40)=1, height(30)=2	B2	Understand
6	What is Sibling? The node with same parent are called siblings. 	B2	Understand
7	Define Binary Tree ADT. A binary tree is a tree in which no node can have more than two children 	B2	Understand
8	What is your opinion about Complete binary tree? A complete binary is a full binary tree in which leaves are at the same depth.  The total no. of nodes:2^{h+1} - 1 No. of leaf nodes:2^{h} Eg. Total no. of nodes : 7 No. of leaf nodes : 4	B4	Analyze

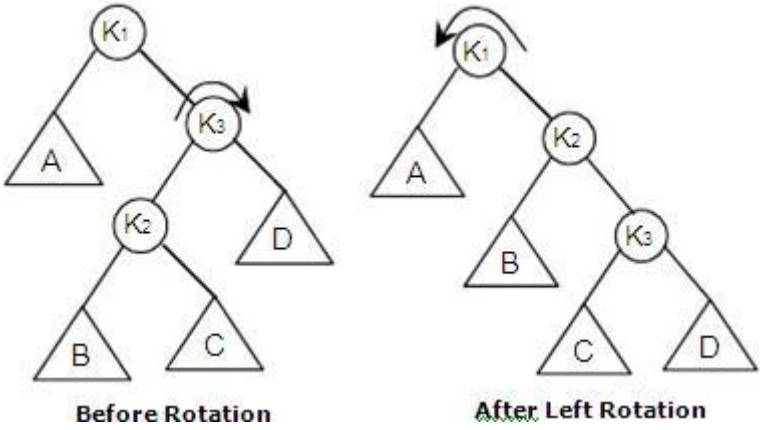
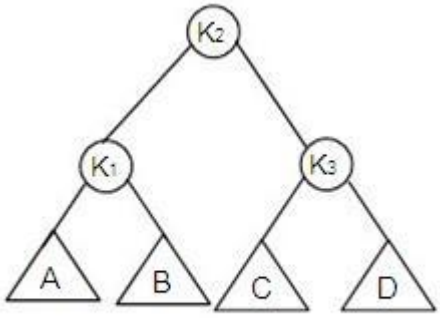
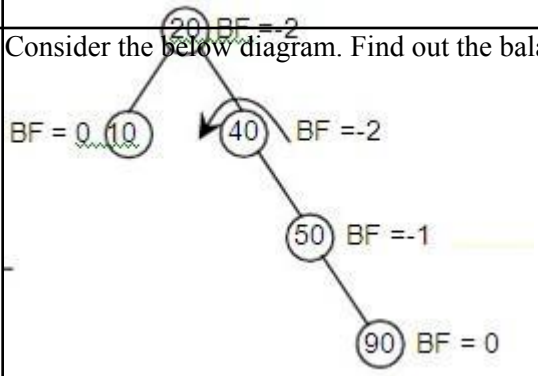
9	What are the ways for representation of binary tree? The binary tree can be represented by using the following two ways: • Linear representation(sequential) • Linked representation	B1	Remember

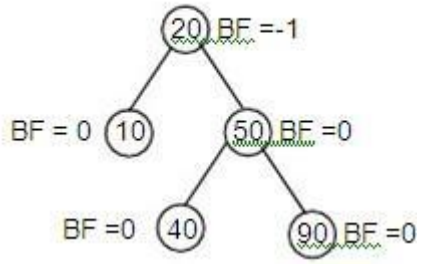
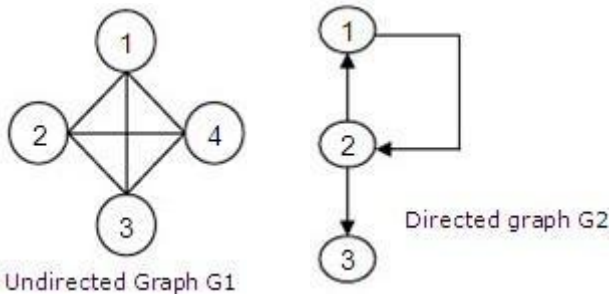
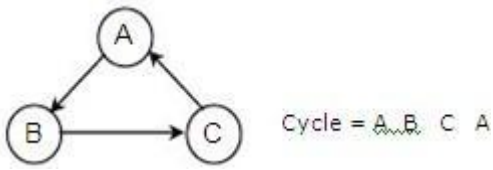
10	What are the applications of trees? - Binary search trees - Expression trees - Threaded trees	B1	Remember
11	Define AVL trees. Adelson – Velskii –and Landis introduced a binary tree structure that is balanced with respect to heights of sub trees. The order of retrieval, insertion and deletion is only O (log n). This tree structure is called AVL tree. The criteria that is used to determine the "level" of "balanced-tree" is the difference between the heights of sub trees of a root in the tree. The "height of tree is the "number levels" in the tree. To be more formal, the height of a tree is defined as follows: 1. The height of a tree with no elements is 0 2. The height of a tree with 1 element is 1 3. The height of tree with > 1 element is equal to 1 + height of its tallest sub tree An AVL tree is a binary tree in which the difference between the height of the right and left sub trees or the root node is never more than one.  	B2	Understand
12	What is your opinion about single rotation LL with example? **Single Rotation with Left - Left Rotation(Left of Left):** The left child (K1) of the root (K2) is promoted as root node. The root (K2) is promoted as the right child of the new root(K1). The right child of the K1 is converted to the left child of K2  	B4	Analyze

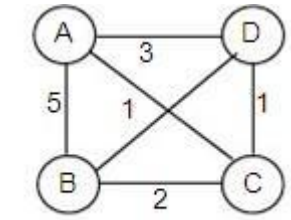
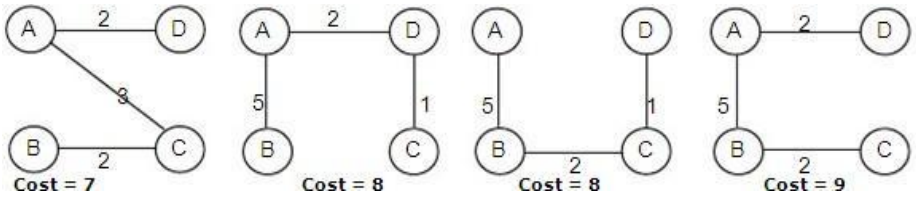
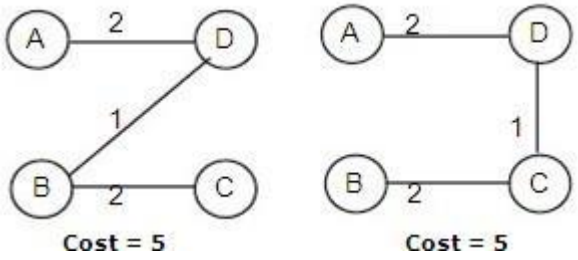
13	What is your opinion about single rotation RR with example? **Single Rotation with Right – Right Rotation(Right of Right):** The right child(K2) of the root(K1) is promoted as root node. Its(K2) left child(Y) is converted to right child of the previous root node(K1).The previous root node(K1) is considered as the left child of		

	new root node(K2).  	B4	Analyze
14	What is your opinion about Double Rotation with Left (Insertion in the Right Subtree of the Left Subtree i.e. Left-Right Rotation) with example? **Routine to perform double rotation with right and then Left:** 1.Apply single Right rotation at K1, as a result K2 becomes the left child of the root node K3 and K1 becomes the left child of K2. 2.Apply single Left rotation at the root node K3, as a result K2 becomes the root node and K1 becomes the left child of it and K3 becomes right child of it. Now the tree is balanced.   After Left Rotation	B4	Analyze

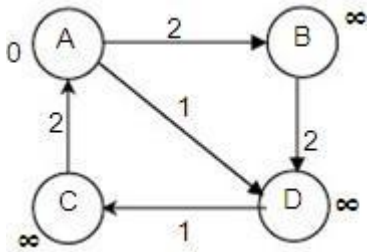
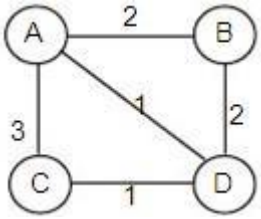
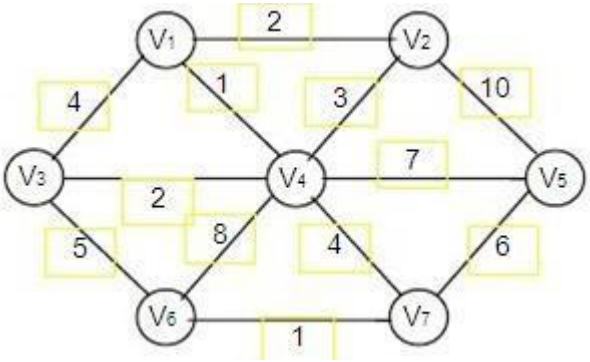
15	What is your opinion about Double Rotation with Right(Insertion in the Left Subtree of the Right Subtree Right-Left Rotation) with example? **Routine to perform double rotation with right and then left:**	B4	Analyze

	1. Apply single Left rotation at K3, as a result K2 becomes the right child of the root node K1 and K3 becomes the right child of K2. 2. Apply single Right rotation at the root node K1, as a result K2 becomes the root node and K1 becomes the left child of it and K3 becomes right child of it. Now the tree is balanced.  Before Rotation After Left Rotation  After Right Rotation		
16	Consider the below diagram. Find out the balanced tree.  BF – Balance Factor Answer:	B5	Evaluate

	After the Right Rotation:  		
17	Define Graph A graph G= (V, E) is an ordered pair of finite sets V and E. The elements of V are called as vertices are also called as nodes and points. The elements of E are called edges are also called arcs and lines. Each edge in E joins two different vertices of V and is denoted by the tuple (i,j), where i and j are the two vertices joined by E. --  	B1	Understand
18	Define Cycle: Cycle in a directed graph is a simple cycle in which no vertex is repeated except that the first and last are identical.  	B2	Remember
19	What is meant by Shortest Path Algorithm? The Shortest path algorithm determines the minimum cost of the path from source to every other vertex. The cost of the path $V_1, V_2, \ldots V_N$ is $\sum_{i=1}^{n-1} C_{i,i+1}$ This is referred as weighted path length. The unweighted path length is the number of the edges on the path, namely N - 1. Two types of shortest path problems are 1.Single source shortest path problem 2. All pairs shortest path problem	B2	Remember
20	What did you observe minimum spanning tree?	B2	Remember

	Spanning Tree with minimum cost is considered as Minimum Spanning Tree. A tree is a connected graph with no cycles. A spanning tree is a subgraph of G that has the same set of vertices of G and is a tree. A minimum spanning tree of a weighted connected graph G is its spanning tree of the smallest weight, where the weight of a tree is defined as the sum of the weights on all its edges. The total number of edges in Minimum Spanning Tree (MST) is |V|-1 where V is the number of vertices.  **Connected Graph G**   		
Part–B(13/15 Marks)			
1	Describe different ways of representation of a binary trees with example.	B4	Analyze
2	Describe the different types of tree traversals with an application?	B4	Analyze
3	Explain with example , how to perform Single rotation in AVL tree	B6	Create
4	Explain with example , how to perform Double rotation in AVL tree	B6	Create
5	How would you demonstrate the different ways of Representations for Graphs	B4	Analyze
6	Explain how to find shortest path using Dijkstra's algorithm with an example	B4	Analyze

7	Predict the outcome for Finding the shortest path of the following weighted graph.	B5	Evaluate

			
8	Explain about Minimum spanning tree using Prim's algorithm9 	B5	Evaluate
9	Predict the outcome from the following graph, construct MST using Prim's algorithm. 	B5	Evaluate
10	Explain the different applications of Graphs with neat sketch.	B4	Analyze

.