**UNIT –I -CLO-1**

**Multiple Choice Questions**

1. **The complexity of bubble-sort algorithm is (page No:437)**
   1. O(n2)
   2. O(n)
   3. O(log n)
   4. O(n log n)

**Ans:a**

1. **Binary search is more suitable for: (Page No:426)**
2. array
3. linked list
4. stack
5. Queue

**Ans:a**

1. **The complexity of binary-search algorithm is (Page No:427)**
2. O(log n)
3. O(n log n)
4. O(n)
5. O(n2)

**Ans:a**

1. **The estimated amount of time required in executing an algorithm is referred to as \_\_\_\_\_ of the algorithm. (page No:54)**
2. time complexity
3. space complexity
4. time and space complexity
5. none of the above

**Ans:a**

1. **The searching technique suitable for unsorted arrays: (PageNo:424)**
2. binary search
3. linear search
4. any of these
5. none of these

**Ans:b**

1. **A theoretical measure of algorithm execution, usually the time/ memory needed ,**

**given the problem size n , is referred to as (Page No: 57)**

1. Big O notation
2. Polish notation
3. Time notation
4. space complexity

**Ans:a**

1. **The main measures for the efficiency of an algorithm are (page No: 54)**
2. processor and memory
3. complexity and capacity
4. time and space
5. data and space

**Ans:c**

1. **Which of the following cases does not exist in complexity theory:(Page No: 54)**
2. best case
3. worst case
4. average case
5. Null case

**Ans:d**

1. **The worst case occurs in linear search algorithm when (Page No: 425)**
2. item is in the middle of the array
3. item is not in the array
4. item is the last element in the array
5. item is the last element in the array or not in the array at-all

**Ans:d**

1. **The complexity of linear search algorithm is (Page No:425)**
2. O(n)
3. O(log n)
4. O(n2)
5. O(n log n)

**Ans:a**

1. **Which of the following data structures is not a linear data structure(Page No:45)**
2. arrays
3. linked lists
4. queue
5. graph

**Ans:d**

1. **Which of the following data structures can’t store non-homogeneous data-elements (Page No: 45)**
2. Arrays
3. Records
4. Pointers
5. Stack

**Ans: a**

1. **Which of the following abstract data types is not used by integer abstract data type**

**group? (Page No:50)**

1. short
2. int
3. float
4. long

**Ans:c**

1. **While calculating time-complexity, the program-time which is considered is (Page No:54)**
2. compile time
3. execution time
4. both compile and run-time
5. none of the above

**Ans:c**

1. **The time complexity of the following algorithm is: (Page No:50)**

**sum(a,n){ s=0; for i= 1 to n{s=s+a[i]; } return s;}**

1. 3n+2
2. 2n +3
3. n+1
4. 2n+2

**Ans:a**

**16. A mathematical model with a collection of operations defined on that model is called (Page No:50)**

1. Data structure
2. Abstract Data Type
3. Primitive Data Type
4. Algorithm

**Ans: b**

**17. Representation of data structure in memory is known as: (Page No:50)**

1. Recursive
2. Abstract data type
3. Storage structure
4. File structure

**Ans:b**

**18. Input instance for which algorithm take minimum possible time is called (Page No:54)**

1. Worst case
2. Best case
3. Average case
4. Null case

**Ans: b**

**19 .Input instance for which algorithm take maximum possible time is called (Page No:54)**

1. Worst case
2. Best case
3. Average case
4. Null case

**Ans: a**

**20. Which case analysis appropriate when the response time of the algorithm is critical? (Page No:54)**

1. Worst case
2. Best case
3. Average case
4. null case

**Ans:a**

**21. Which of the following is considered an Abstract Data Type? (Page No:50)**

1. Array
2. reference variable
3. any of the primitive types (e.g., int, double, char)
4. Stack

**Ans: d**

**22. Which of the following data structure is linear data structure? (Page No:45)**

1. Trees
2. Graphs
3. Arrays
4. None of above

**Ans: c**

**23. ........ sorting algorithm is frequently used when n is small where n is total number of elements. (Page No:438)**

1. Heap
2. Insertion
3. Bubble
4. Quick

**Ans: b**

**24. The complexity of sorting algorithm measures the ...... as a function of the number n of items to be sorter. (Page No:437)**

1. average time
2. running time
3. average-case complexity
4. case-complexity

**Ans:b**

**25. Assume that we use Bubble Sort to sort n distinct elements in ascending order. When does the best case of Bubble Sort occur? (Page No: 434)**

a. When elements are sorted in ascending order  
b. When elements are sorted in descending order  
c. When elements are not sorted by any order  
d.  There is no best case for Bubble Sort. It always takes O(n\*n) time

**Ans:a**

**26. Which of the following is not a limitation of binary search algorithm? (Page No:426)**

1. must use a sorted array
2. requirement of sorted array is expensive when a lot of insertion and deletions are needed
3. there must be a mechanism to access middle element directly
4. binary search algorithm is not efficient when the data elements more than 1500.

**Ans: d**

**27. The worst case time required to search a given element in a sorted linked list of length**

**n is (Page No:439)**

1. O(1)
2. O(log2 n)
3. O(n)
4. O(n log2 n)

**Ans:c**

**28. …………… is not the component of data structure. (Page No:43)** a. Operations  
 b. Storage Structures  
 c. Algorithms  
 d. None of above  
 **Ans: d**  
**29. Which of the following is not the part of ADT description? (Page No:45)**

a. Data  
b. Operations  
c. Both of the above  
d. None of the above

Ans:d

**30. Which if the following is/are the levels of implementation of data structure? (Page No:43)**

a. Abstract level  
b. Application level  
c. Implementation level  
d. All of the above

**Ans:d**

**31. After each iteration in bubble sor**t **(Page No:435)**

1. [at least one element is at its sorted position.](about:blank)
2. [one less comparison is made in the next iteration.](about:blank)
3. Both A & B are true.
4. [Neither A or B are true.](about:blank)

**Ans:a**

32. **When determining the efficiency of algorithm, the space factor is measured by** **(Page No:54)**

* 1. Counting the maximum memory needed by the algorithm
  2. Counting the minimum memory needed by the algorithm
  3. Counting Counting the average memory needed by the algorithm
  4. Counting the maximum disk space needed by the algorithm

**Ans: a**

**33. The Average case occur in linear search algorithm (Page No:425)**  
a.    When Item is somewhere in the middle of the array  
b.    When Item is not in the array at all  
c.    When Item is the last element in the array  
d.    When Item is the last element in the array or is not there at all

**Ans:a**

**34.When determining the efficiency of algorithm the time factor is measured by (Page No:55)**  
a.    Counting microseconds  
b.    Counting the number of key operations  
c.    Counting the number of statements  
d.    Counting the kilobytes of algorithm

**Ans:b**

**35.For an algorithm the complexity of the average case is (Page No:54)**  
a.    Much more complicated to analyze than that of worst case  
b.    Much more simpler to analyze than that of worst case  
c.    Sometimes more complicated and some other times simpler than that of worst case  
d.    None or above

**Ans: a**

**36. which of the following is not the required condition for binary search algorithm? (Page No:426)**

a. The list must be sorted  
b. there should be the direct access to the middle element in any sublist  
c. There must be mechanism to delete and/or insert elements in list  
d. none of above  
**Ans:c**

**37. Which of the following is not a limitation of binary search algorithm? (Page No:426)**

a. must use a sorted array  
b. requirement of sorted array is expensive when a lot of insertion and deletions are needed  
c. there must be a mechanism to access middle element directly  
d. binary search algorithm is not efficient when the data elements are more than 1000.  
**Ans:d**

**38. Binary search algorithm can not be applied to (Page No:426)**

a. sorted linked list  
b. sorted binary trees  
c. sorted linear array  
d. pointer array  
**Ans:a**

**39. When new data are to be inserted into a data structure, but there is no available space; this situation is usually called (Page No:44)**

a. underflow  
b. overflow  
c. housefull  
d. saturated  
 **Ans:b**

**40. Operations on a data structures (Page No:49)**

a. insertion

b. deletion

c. construction.

d. all the above.

Ans.d

**4 marks:**

1. **List out the types of data structures.(Page No-45)**

Data Structure is a way of collecting and organizing data in such a way that we can perform operations on these data in an effective way.

**Basic types of Data Structures**

**1.Primitive Data Structures**.

Anything that can store data can be called as a data structure, hence Integer, Float, Boolean, Char etc, all are data structures. They are known as **Primitive Data Structures**.

**2. Non-primitive Data Structures**

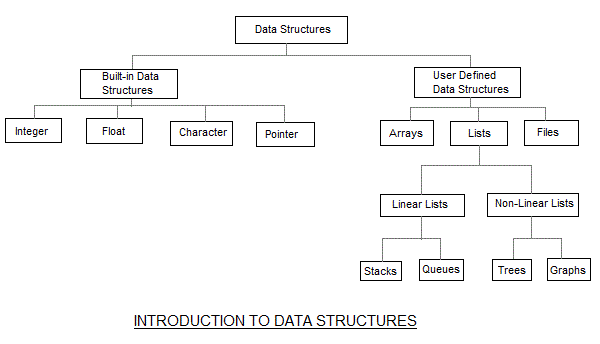
Non-primitive data structures are more complicated data structures and are derived from primitive data structures.Non-primitive data structures can further be classified into two categories: linear and non-linear data structures.

**Linear and Non-linear Structures**

If the elements of a data structure are stored in a linear or sequential order, then it is a linear data structure. Examples include arrays, linked lists, stacks, and queues.

Then we also have some complex Data Structures, which are used to store large and connected data. Some example of **Abstract Data Structure** are:

* Linked List
* Tree
* Graph
* Stack, Queue etc.

All these data structures allow us to perform different operations on data. We select these data structures based on which type of operation is required. We will look into these data structures in more details in our later lessons.

1. **List out the various operations on data structures. (Page No:49)**

The basic operations that are performed on data structures are as follows:  
***Traversing*** It means to access each data item exactly once so that it can be processed. For example, to print the names of all the students in a class.

***Searching*** It is used to find the location of one or more data items that satisfy the given constraint.Such a data item may or may not be present in the given collection of data items. For example, to find the names of all the students who secured 100 marks in mathematics.

***Inserting*** It is used to add new data items to the given list of data items. For example, to add the details of a new student who has recently joined the course.

***Deleting*** It means to remove (delete) a particular data item from the given collection of data items. For example, to delete the name of a student who has left the course.

***Sorting*** Data items can be arranged in some order like ascending order or descending order depending on the type of application. For example, arranging the names of students in a class in an alphabetical order, or calculating the top three winners by arranging the participants’ scores in descending order and then extracting the top three.

***Merging*** Lists of two sorted data items can be combined to form a single list of sorted data items.

1. **Define – Abstract Data Type. Give one example for ADT. (Page No:50)**

Abstract data types or ADTs are a mathematical specification of a set of data and the set of operations that can be performed on the data. They are abstract in the sense that the focus is on the definitions of the constructor that returns an abstract handle that represents the data, and the various operations with their arguments. The actual implementation is not defined, and does not affect the use of the ADT.

For example, rational numbers (numbers that can be written in the form a/b where a and b are integers) cannot be represented natively in a computer. A Rational ADT could be defined as shown below.

**Construction:** Create an instance of a rational number ADT using two integers, a and b, where a represents the numerator and b represents the denominator.

**Operations:** addition, subtraction, multiplication, division, exponentiation, comparison, simplify, conversion to a real (floating point) number

1. **What is the time complexity of an algorithm? Illustrate it with a simple example. (Page No:54)**

An algorithm is said to be efficient and fast, if it takes less time to execute and

consumes less memory space. The performance of an algorithm is measured on the basis

of following properties:

1. Time Complexity
2. Space Complexity

Time Complexity is a way to represent the amount of time needed by the program to run to

completion.

Example: FOR LOOP

for(i=0; i< N; i++)

{statement;}

Here, the statement executes for n times.

T(n) = n

1. **Explain in detail, the significance and limitations of Big O. (Page No:60)**

The Big O notation defines an upper bound of an algorithm, it bounds a function only from above.

If f(n)<==g(n), c>0,n>=n0, then f(n)=O(g(n) and g(n) is an asymptotically tight upper bound for f(n).

***Limitations of Big O Notation***

There are certain limitations with the Big O notation of expressing the complexity of algorithms.These limitations are as follows:

* Many algorithms are simply too hard to analyse mathematically.
* There may not be sufficient information to calculate the behaviour of the algorithm in the
* average case.
* Big O analysis only tells us how the algorithm grows with the size of the problem, not how efficient it is, as it does not consider the programming effort.
* It ignores important constants.

1. **How will you measure the running time of an algorithm? (Page No:55)**

When a program is to be run repeatedly, its efficiency and that of its underlying Efficiency algorithm become important. Generally, we associate efficiency with the time it takes a program to run, although there are other resources that a program sometimes must conserve, such as

* 1. The amount of storage space taken by its variables.
  2. The amount of traffic it generates on a network of computers.

iii)The amount of data that must be moved to and from disks.

For large problems, however, it is the running time that determines whether a given program can be used,. We shall, in fact, take the efficiency of a program to mean the amount of time it takes, measured as a function of the size of its input.

1. **Discuss the complexity of insertion sort. (Page No:439)**

The best case occurs when the array is already sorted. In this case ,the running time of the algorithm, has a linear running time.

The worst case of the first element of the insertion sort algorithm occurs when the array is sorted in reverse order. In this the first element of the unsorted set has to be compared with almost every element in the sorted set. Furthermore, every iteration of the inner loop will have to shift the elements of the sorted set of the array before inserting the next element.

Insertion sort has a quadratic running time(O(n2).

1. **Discuss the complexity of bubble sort. (Page No:437)**

The complexity of any sorting algorithm depends upon the number of comparisons.

There are N-1 passes in total. In the first pass, N-1 comparisons are made to place the highest element in its correct position. Then, in pass 2, there are N-2 comparisond and the second highest element is placed in its positions.Therefore, to compute complexity of bubble sort, we need to calculate the total number of comparisons.

F(n)=(n-1)+(n-2)+(n-3)+----+3+2+1

F(n)=n(n-1)/2

F(n)=n2/2

F(n)=O(n2)

1. **Discuss the complexity of binary search. (Page No:427)**

The complexity of the binary search algorithm can be expressed as

f(n), where n is the number of elements in the array. The complexity of any algorithm depends upon the number of comparisons.the size of the segment where has to be made is reduced to half. Thus,we can say that, in order to locate a particular value in the array, the total number of comparisons that will made is given as,

F(n)=log n

1. **What is the need of using data structures? (Page No:44)**

* Data structures organize data
  + This gives ***more efficient programs.***
* More powerful computers encourage more complex applications.
* More complex applications demand more calculations.

Complex computing tasks are unlike our everyday experience

* Any organization for a collection of records can be searched, processed in any order, or modified.
  + The choice of data structure and algorithm can make the difference between a program running in a few seconds or many days.
* A solution is said to be ***efficient*** if it solves the problem within its ***resource constraints***.
  + Space
  + Time

The ***cost*** of a solution is the amount of resources that the solution consumes

**12 Marks**

1. Write an algorithm for binary search with an example. **(Page No:426)**
2. Explain in detail, bubble sort with an example**(Page No:434)**.
3. Explain in detail, insertion sort algorithm with example**(Page No:438)**.
4. Write a program to sort a array using insertion sort. **(Page No:440)**
5. Sort the following sequence of numbers in ascending order using bubble and show the each pass { 42,34,75,23,21,18,19,67,78}**(Page No:435)**
6. Apply binary search algorithm in a program to search an element in the data structure. using binary . Demonstrate the set of numbers { 10,8,2,7,3,4,9,1,6,5}**(Page No:426)**
7. Explain in detail, the complexity of an algorithm. **(Page No:54)**