J. BASKARAN M.Sc., B.Ed. (C.S)
jbaskaran89@gmail.com
Puducherry.

J. ILAKKIA M.Sc., M.Phil., B.Ed. (C.S)
jilakkia@gmail.com
Puducherry.

4. ALGORITHMIC STRATEGIES

Section – A

Choose the best answer **(1 Mark)**

1. The word comes from the name of a Persian mathematician Abu Ja'far Mohammed ibn-i Musa al Khowarizmi is called?

(A) Flowchart (B) Flow **(C) Algorithm** (D) Syntax

2. From the following sorting algorithms which algorithm needs the minimum number of swaps?

(A) Bubble sort (B) Quick sort (C) Merge sort **(D) Selection sort**

3. Two main measures for the efficiency of an algorithm are

(A) Processor and memory (B) Complexity and capacity

(C) Time and space (D) Data and space

4. The complexity of linear search algorithm is

(A) O(n) (B) O(log n) (C) O(n2) (D) O(n log n)

5. From the following sorting algorithms which has the lowest worst case complexity?

(A) Bubble sort (B) Quick sort **(C) Merge sort** (D) Selection sort

6. Which of the following is not a stable sorting algorithm?

(A) Insertion sort **(B) Selection sort** (C) Bubble sort (D) Merge sort

7. Time complexity of bubble sort in best case is

(A) θ (n) (B) θ (nlogn) (C) θ (n2) (D) θ (n(logn) 2)

8. The Θ notation in asymptotic evaluation represents

(A) Base case **(B) Average case** (C) Worst case (D) NULL case

9. If a problem can be broken into subproblems which are reused several times, the problem possesses which property?

(A) Overlapping subproblems (B) Optimal substructure

(C) Memoization (D) Greedy

10. In dynamic programming, the technique of storing the previously calculated values is called ?

(A) Saving value property (B) Storing value property

(C) Memoization (D) Mapping

Section-B

Answer the following questions **(2 Marks)**

1. What is an Algorithm?

- An algorithm is a finite set of instructions to accomplish a particular task.
- It is a step-by-step procedure for solving a given problem.

2. Define Pseudo code.

- **Pseudo code** is a methodology that allows the programmer to represent the implementation of an algorithm.
- It has no syntax like programming languages and thus can't be compiled or interpreted by the computer.

3. Who is an Algorist?

- An Algorist is a person skilled in the design of algorithms
- An algorithmic artist

4. What is Sorting?

- Sorting is a process of arranging group of items in an ascending or descending order.
- Bubble Sort, Quick Sort, Heap Sort, Merge Sort, Selection Sort are the various sorting algorithms.

5. What is searching? Write its types.

- A Search algorithm is the step-by-step procedure used to locate specific data among a collection of data.
- **Example:** Linear Search, Binary Search

Section-C

Answer the following questions **(3 Marks)**

1. List the characteristics of an algorithm.

- Input
- Output
- Finiteness
- Definiteness
- Effectiveness
- Correctness
- Simplicity
- Unambiguous
- Feasibility
- Portable
- Independent

2. Discuss about Algorithmic complexity and its types.

ALGORITHMIC COMPLEXITY:

- The complexity of an algorithm f(n) gives the running time and/or the storage space required by the algorithm in terms of n as the size of input data.

TYPES OF COMPLEXITY:

1. Time Complexity

- The Time complexity of an algorithm is given by the number of steps taken by the algorithm to complete the process.

2. Space Complexity

- **Space complexity** of an algorithm is the amount of memory required to run to its completion.
- The space required by an algorithm is equal to the sum of **fixed part and variable part.**

3. What are the factors that influence time and space complexity.

The two main factors, which decide the efficiency of an algorithm are,

- **Time Factor** -Time is measured by counting the number of key operations like comparisons in the sorting algorithm.
- **Space Factor** - Space is measured by the maximum memory space required by the algorithm.

4. Write a note on Asymptotic notation.

- **Asymptotic Notations** are languages that use meaningful statements about time and space complexity.
- The following three asymptotic notations are mostly used to represent time complexity of algorithms:

(i) Big O

- Big O is often used to describe the worst-case of an algorithm.

(ii) Big Ω

- Big Omega is the reverse Big O.
- **Example:** If **Big O** is used to describe the upper bound (worst - case) then, **Big Ω** is used to describe the lower bound (best-case).

(iii) Big Θ

- When an algorithm has a complexity with **lower bound = upper bound**, that algorithm has a complexity O (n log n) and Ω (n log n), it's actually has the complexity Θ (n log n).
- Time complexity is **n log n** in both best-case and worst-case.

5. What do you understand by Dynamic programming?

- Dynamic programming is used when the solution to a problem can be viewed as the result of a sequence of decisions.
- Dynamic programming approach is similar to divide and conquer (i.e) the problem can be divided into smaller sub-problems.
- Results of the sub-problems can be re-used to complete the process.
- Dynamic programming approaches are used to find the solution in optimized way.

Section - D

Answer the following questions: **(5 Marks)**

1. Explain the characteristics of an algorithm.

Characteristics	Meaning
Input	Zero or more quantities to be supplied.
Output	At least one quantity is produced.
Finiteness	Algorithms must terminate after finite number of steps.
Definiteness	All operations should be well defined.
Effectiveness	Every instruction must be carried out effectively.
Correctness	The algorithms should be error free.
Simplicity	Easy to implement.
Unambiguous	Algorithm should be clear and unambiguous. Each of its steps should be clear and must lead to only one meaning.
Feasibility	Should be feasible with the available resources.
Portable	An algorithm should be generic, independent and able to handle all range of inputs.
Independent	An algorithm should have step-by-step directions, which should be independent of any programming code.

2. Discuss about Linear search algorithm.

LINEAR SEARCH:

- Linear search also called sequential search is a sequential method for finding a particular value in a list.
- This method checks the search element with each element in sequence until the desired element is found or the list is exhausted.
- In this searching algorithm, list need not be ordered.

Pseudo code:

1. Traverse the array using for loop
2. In every iteration, compare the target search key value with the current value of the list.
   - If the values match, display the current index and value of the array
   - If the values do not match, move on to the next array element. If no match is found, display the search element not found.
3. If no match is found, display the search element not found.

Example:

- To search the number 25 in the array given below, linear search will go step by step in a sequential order starting from the first element in the given array.
- if the search element is found that index is returned otherwise the search is continued till the last index of the array.
- In this example number 25 is found at index number 3.

index	0	1	2	3	4
values	10	12	20	25	30

Snippet:

Input: values[]={10,12,20,25,30}
Target=25

Output:

3

3. What is Binary search? Discuss with example.

BINARY SEARCH:

- Binary search also called half-interval search algorithm.
- It finds the position of a search element within a sorted array.
- The binary search algorithm can be done as divide-and-conquer search algorithm and executes in logarithmic time.

Pseudo code for Binary search:

1. Start with the middle element:

a) If the search element is equal to the middle element of the array, then return the index of the middle element.

b) If not, then compare the middle element with the search value,

c) If **(Search element > number in the middle index)**, then select the elements to the right side of the middle index, and go to Step-1.

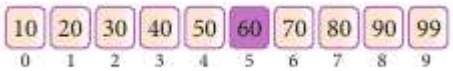
d) If **(Search element < number in the middle index)**, then select the elements to the left side of the middle index, and start with Step-1.

2. When a **match is found**, **display success message** with the index of the element matched.

3. If **no match is found** for all comparisons, then **display unsuccessful message**.

<u>Binary Search Working principles with example</u>:

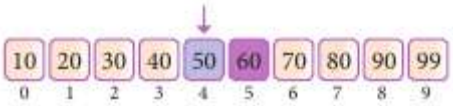
- List of elements in an array must be sorted first for Binary search.
- The array is being sorted in the given example and it is suitable to do the binary search algorithm.
- Let us assume that the **search element is 60** and we need to search the location or index of search element 60 using binary search.



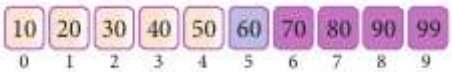
- First, we find index of middle element of the array by using this formula :

mid = low + (high - low) / 2

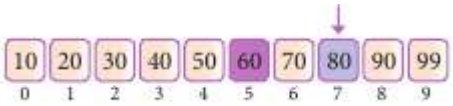
- Here it is, 0 + (9 - 0) / 2 = 4. So, 4 is the mid value of the array.



- Compare the value stored at index 4 with target value, which is not match with search element. As the search value 60 > 50.

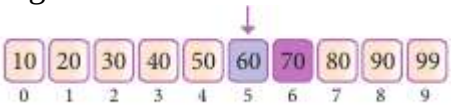


- Now we change our search range **low to mid + 1** and find the new mid value as index 7.
- We compare the value stored at index 7 with our target value.

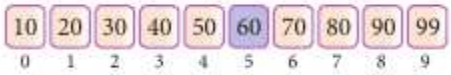


- Element not found because the value in index 7 is greater than search value . (80 > 60)
- So, the search element must be in the lower part from the current mid value location

- Now we change our search range **low to mid - 1** and find the new mid value as index 5



- Now we compare the value stored at location 5 with our search element.
- We found that it is a match.



- We can conclude that the search element 60 is found at lcoation or index 5.

4. Explain the Bubble sort algorithm with example.

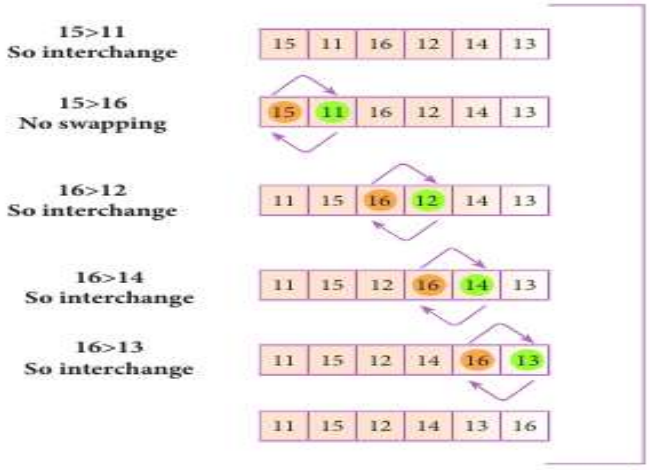
- Bubble sort is a simple sorting algorithm, it starts at the beginning of the list of values stored in an array.
- It compares each pair of adjacent elements and swaps them if they are in the unsorted order.
- This comparison and passed to be continued until no swaps are needed, which shows the values in an array is sorted.
- It is named so becase, the smaller elements "bubble" to the top of the list.
- It is too slow and less efficient when compared to other sorting methods.

Pseudo code

1. Start with the first element i.e., index = 0, compare the current element with the next element of the array.
2. If the current element is greater than the next element of the array, swap them.
3. If the current element is less than the next or right side of the element, move to the next element.
4. Go to Step 1 and repeat until end of the index is reached.

Example:

- Consider an array with values {15, 11, 16, 12, 14, 13}
- Below, we have a pictorial representation of how bubble sort.



- The above pictorial example is for iteration-1.
- Similarly, remaining iteration can be done.
- The final iteration will give the sorted array.
- At the end of all the iterations we will get the sorted values in an array as given below:

5. Explain the concept of Dynamic programming with suitable example.

- Dynamic programming is used when the solution to a problem can be viewed as the result of a sequence of decisions.
- Dynamic programming approach is similar to divide and conquer (i.e) the problem can be divided into smaller sub-problems.
- Results of the sub-problems can be re-used to complete the process.
- Dynamic programming approaches are used to find the solution in optimized way.

Steps to do Dynamic programming

- The given problem will be divided into smaller overlapping sub-problems.
- An optimum solution for the given problem can be achieved by using result of smaller sub-problem.
- Dynamic algorithms uses Memoization.

Fibonacci Iterative Algorithm with Dynamic Programming Approach

- The following example shows a simple Dynamic programming approach for the generation of Fibonacci series.
- Initialize f0=0, f1 =1
- step-1: Print the initial values of Fibonacci f0 and f1
- step-2: Calculate fibanocci fib ← f0 + f1
- step-3: Assign f0← f1, f1← fib
- step-4: Print the next consecutive value of fibanocci fib
- step-5: Goto step-2 and repeat until the specified number of terms generated
- For example if we generate fibonacci series upto 10 digits, the algorithm will generate the series as shown below:
- The Fibonacci series is : 0 1 1 2 3 5 8 13 21 34 55

PREPARED BY

J. BASKARAN M.Sc., B.Ed. (C.S)
jbaskaran89@gmail.com

J. ILAKKIA M.Sc., M.Phil., B.Ed. (C.S)
jilakkia@gmail.com