DeepSeek - Into the Unknown

chat.deepseek.com/a/chat/s/81a317d3-9255-4298-8e74-4224a96021ff

1. Define data structure with example. Distinguish between linear and non-linear data structure?

Data Structure: A data structure is a way of organizing and storing data in a computer so that it can be accessed and modified efficiently. It defines the relationship between the data, the operations that can be performed on the data, and the rules for organizing the data.

Example: Arrays, Linked Lists, Stacks, Queues, Trees, Graphs, etc.

Linear vs. Non-linear Data Structures:

- **Linear Data Structure**: Elements are arranged in a sequential manner. Each element has a unique predecessor and successor except for the first and last elements. Examples include arrays, linked lists, stacks, and queues.
- **Non-linear Data Structure**: Elements are not arranged sequentially. Each element can have multiple predecessors and successors. Examples include trees and graphs.

---

2. What is complexity of an algorithm? Calculate the time complexity of searching for a specific element in an unsorted array using linear search.

Complexity of an Algorithm: Complexity refers to the amount of resources (time and space) required by an algorithm to solve a problem. It is usually expressed using Big O notation.

Time Complexity of Linear Search:

- In the worst case, the element is not present in the array, and the algorithm checks all n elements.
- Time Complexity: O(n).

---

3. Write down the advantages of using dynamic arrays.

- **Resizable**: Dynamic arrays can grow or shrink in size as needed.
- **Efficient Memory Usage**: Memory is allocated dynamically, so it uses only as much memory as required.

- **Random Access**: Elements can be accessed in constant time O(1).
- **Ease of Implementation**: Many programming languages provide built-in support for dynamic arrays (e.g., `ArrayList` in Java, `vector` in C++).

---

4. Define data structure. List out the areas in which data structures are applied extensively.

Data Structure: A data structure is a way of organizing and storing data in a computer so that it can be accessed and modified efficiently.

Areas of Application:

- Database Management Systems
- Operating Systems
- Artificial Intelligence
- Compiler Design
- Network Data Handling
- Graphics and Multimedia
- Blockchain Technology

---

5. Consider the linear arrays AAA (5:50), BBB (-50:10), and CCC (18). Find the number of elements in each array.

- **AAA (5:50)**: Number of elements = $50-5+1=46$.
- **BBB (-50:10)**: Number of elements = $10-(-50)+1=61$.
- **CCC (18)**: Number of elements = 18.

---

6. Suppose Base (AAA)=300 and w=4 words per memory cell for AAA. Find the address of AAA[15], AAA[35], and AAA[55].

- **Formula**: Address=Base+(i−Lower Bound)×w
- **AAA[15]**: $300+(15-5)\times4=300+40=340$.
- **AAA[35]**: $300+(35-5)\times4=300+120=420$.
- **AAA[55]**: Invalid, as 55 is out of bounds (5:50).

---

7. Distinguish between linear and binary search methods.

- **Linear Search**:
  - Works on both sorted and unsorted arrays.
  - Time Complexity: O(n).
  - Checks each element sequentially.
- **Binary Search**:
  - Works only on sorted arrays.
  - Time Complexity: O(logn).
  - Divides the array into two halves and eliminates one half based on the comparison.

---

8. How does a pointer array save memory when storing a variable-sized group of data? Discuss with necessary figures.

- **Pointer Array**: Instead of storing the actual data, a pointer array stores the addresses of the data. This allows for dynamic memory allocation and efficient storage of variable-sized data.
- **Memory Saving**: Only the addresses are stored in the array, which are typically smaller in size compared to the actual data.

---

9. Differentiate between data type and data structure. Why should you have good knowledge of data structures as a CSE student?

- **Data Type**: Defines the type of data that a variable can hold (e.g., `int`, `float`, `char`).
- **Data Structure**: Defines how data is organized and stored in memory (e.g., arrays, linked lists).

Importance for CSE Students:

- Efficient problem-solving.
- Optimizing resource usage (time and space).
- Essential for software development and algorithm design.

---

10. Write down the algorithm for bubble sort and analyze the algorithm to find best case and worst case.

Bubble Sort Algorithm:

python

Copy

```
for i from 0 to n-1:
    for j from 0 to n-i-1:
        if arr[j] > arr[j+1]:
            swap(arr[j], arr[j+1])
```

Best Case: O(n) (when the array is already sorted).
Worst Case: O(n2) (when the array is sorted in reverse order).

11. An array CSE[-4 … 6, -2 … 12], stores elements in Row Major Wise, with the address AR[2][3] as 4142. If each element requires 2 bytes of storage, find the Base address.

- **Formula**: Address=Base+(i−Lower Bound1)×Columns×w+(j−Lower Bound2)×w
- Given: Address[2][3]=4142, w=2, Columns=12−(−2)+1=15.
- Solve for Base: 4142=Base+(2−(−4))×15×2+(3−(−2))×2.
- 4142=Base+6×30+5×2.
- 4142=Base+180+10.
- Base=4142−190=3952.

12. Define data structure. Name different types of data structure. List operations that can be performed on these data structures.

Data Structure: A way of organizing and storing data in a computer.

Types:

- Linear: Arrays, Linked Lists, Stacks, Queues.
- Non-linear: Trees, Graphs.

Operations:

- Insertion
- Deletion
- Traversal

- Searching
- Sorting

---

13. Write down the algorithm for binary search. Validate the algorithm with a suitable data set.

Binary Search Algorithm:

python

Copy

```
def binary_search(arr, target):
    low, high = 0, len(arr) - 1
    while low <= high:
        mid = (low + high) // 2
        if arr[mid] == target:
            return mid
        elif arr[mid] < target:
            low = mid + 1
        else:
            high = mid - 1
    return -1
```

Validation:

- Data Set: [11, 22, 33, 44, 55, 59, 66, 70, 77, 80, 85, 88, 91, 95, 99].
- Search for 55: Found at index 4.
- Search for 75: Not found.

---

14. An array X [-15…….10, 15……40] requires one byte of storage. If beginning location is 1500 determine the location of X [15] [20] using the following:

i) Column Major Wise Calculation.
ii) Row Major Wise Calculation.

- **Column Major**: Address=Base+(j−Lower Bound2)×Rows×w+(i−Lower Bound1)×w.
- **Row Major**: Address=Base+(i−Lower Bound1)×Columns×w+(j−Lower Bound2)×w.

---

15. Define time complexity, space complexity with example.

- **Time Complexity**: Measures the amount of time an algorithm takes to run as a function of the input size. Example: O(n) for linear search.

- **Space Complexity**: Measures the amount of memory an algorithm uses as a function of the input size. Example: O(1) for iterative algorithms.

---

16. Define data structures. Write down and briefly explain types of the data structures that used to store data in computer memory.

Data Structures: A way of organizing and storing data in a computer.

Types:

- **Linear**: Arrays, Linked Lists, Stacks, Queues.
- **Non-linear**: Trees, Graphs.

---

17. What is algorithm? Why Big O notation is used in algorithm? Give example.

Algorithm: A step-by-step procedure to solve a problem.

Big O Notation: Used to describe the upper bound of an algorithm's complexity, providing a worst-case scenario.

Example: O(n2) for bubble sort.

---

18. With a suitable example, briefly describe the commonly performed operations in data structures.

Operations:

- **Insertion**: Adding an element to a data structure.
- **Deletion**: Removing an element from a data structure.
- **Traversal**: Accessing each element of a data structure.
- **Searching**: Finding an element in a data structure.
- **Sorting**: Arranging elements in a specific order.

Example: Inserting an element into an array, deleting an element from a linked list, etc.

---

19. List the operations of the data structure.

- Insertion
- Deletion

- Traversal
- Searching
- Sorting

---

20. List the drawbacks of Bubble Sort algorithms.

- **Time Complexity**: O(n2) in the worst case.
- **Inefficient for Large Data Sets**: Not suitable for large arrays.
- **Lack of Adaptivity**: Does not adapt to the existing order of elements.

---

21. Suppose the following 15 elements are stored in an array A: 11, 22, 33, 44, 55, 59, 66, 70,77, 80, 85, 88, 91, 95, 99. Using the Binary Search algorithm, find ITEM= 75 and ITEM=55.

- **ITEM=55**: Found at index 4.
- **ITEM=75**: Not found.

---

22. Consider the Linear arrays A(4:10, 2:16, 3:23), B(-3:15, 4:9) and C(20)

i) Find the number of elements in each array.
ii) Suppose Base (A) = 300 and w=5 words per memory cell for A. Find the address of A (5, 10,15).

- **i)**
  - A: $(10-4+1)\times(16-2+1)\times(23-3+1)=7\times15\times21=2205$.
  - B: $(15-(-3)+1)\times(9-4+1)=19\times6=114$.
  - C: 20.
- **ii)**
  - Address=$300+(5-4)\times15\times21\times5+(10-2)\times21\times5+(15-3)\times5$.
  - Address=$300+1\times1575+8\times105+12\times5$.
  - Address=$300+1575+840+60=2775$.

---

23. Define Multidimensional Array.

A multidimensional array is an array that contains one or more arrays. It can be thought of as an array of arrays. For example, a 2D array is an array of arrays, and a 3D array is an array of 2D arrays.

---

24. Consider the Linear arrays A(4:10, 2:16, 3:23), B(-3:15, 4:9) and C(20)

i) Find the number of elements in each array.
ii) Suppose Base (A) = 300 and w=5 words per memory cell for A. Find the address of A (5, 10,15).

- **i)**
  - A: $(10-4+1)\times(16-2+1)\times(23-3+1)=7\times15\times21=2205$.
  - B: $(15-(-3)+1)\times(9-4+1)=19\times6=114$.
  - C: 20.
- **ii)**
  - Address=$300+(5-4)\times15\times21\times5+(10-2)\times21\times5+(15-3)\times5$.
  - Address=$300+1\times1575+8\times105+12\times5$.
  - Address=$300+1575+840+60=2775$.
