**Data Structure Algorithm**

**Hitarth Patel  
150096724046  
Jensen Huang**

**Q1. What is a linear data structure? Give two examples.**

**A Linear Data Structure**

A linear data structure is a type of data structure in which the elements are arranged in a sequential manner, meaning that each element is connected to its previous and next element. This arrangement allows for efficient insertion, deletion, and traversal of elements.

**Examples**-Arrays, linked lists, stacks, and queues.

**Q2. What is the primary difference between an array and a linked list?**

|  |  |  |
| --- | --- | --- |
| Characteristics | Arrays | Linked Lists |
| Memory Allocation | Contiguous | Non-Contiguous |
| Size | Fixed | Dynamic |
| Access | Random Access | Sequential Access |
| Insertion/Deletion | Difficult and expensive | Easier and more efficient |
| Memory Usage | Fixed and predictable | Variable and unpredictable |
| Cache Performance | Better cache locality | Poorer cache locality |

**Q3. What type of data structure is best suited for representing hierarchical relationships? Explain?**

**Tree Data Structure**

A tree data structure is best suited for representing hierarchical relationships.

**Characteristics of Tree Data Structure**

* **Nodes**: Each node represents an entity or a piece of data.
* **Edges**: Edges connect nodes, representing relationships between them.
* **Root Node**: The topmost node, which is the starting point of the hierarchy.
* **Child Nodes**: Nodes that are directly connected to a parent node.
* **Leaf Nodes:** Nodes with no child nodes.

**Why Tree Data Structure is Suitable for Hierarchical Relationships**

* **Natural Representation**: Trees naturally represent hierarchical relationships, with each node having a parent-child relationship with its connected nodes.
* **Efficient Traversal:** Tree data structures allow for efficient traversal, making it easy to navigate through the hierarchy.
* **Flexible Structure:** Trees can be structured in various ways, such as binary trees, AVL trees, or B-trees, to suit specific use cases.

**Real-World Examples of Tree Data Structure in Hierarchical Relationships**

* **File System**: A file system is a classic example of a tree data structure, where directories and subdirectories are represented as nodes.
* **Organization Chart:** An organization chart is another example, where employees are represented as nodes, and their reporting relationships are represented as edges.
* **XML/HTML Document Structure**: XML and HTML documents use a tree-like structure to represent the relationships between elements.

**Q4. Name a data structure that operates using the "First In, First Out (FIFO)" principle.**

**FIFO (First-In-First-Out) Principle**

The FIFO principle states that the first element added to the queue will be the first one to be removed. This principle is what makes a queue a queue.

**Q5. Which data structure uses the "Last In, First Out (LIFO)" principle?**

**LIFO (Last-In-First-Out) Principle**

The LIFO principle states that the last element added to the stack will be the first one to be removed. This principle is what makes a stack a stack.

**Q6. Write a C++ program to traverse an array using function?**

#include<iostream>

using namespace std;

void traverseArray(int *arr*[], int *size*) {

for(int i = 0; i < *size*; i++) {

cout << *arr*[i] << " ";

}

}

int main() {

int size;

cout << "Enter the size of the array: ";

cin >> size;

int arr[size];

cout << "Enter elements of the array: ";

for(int i = 0; i < size; i++) {

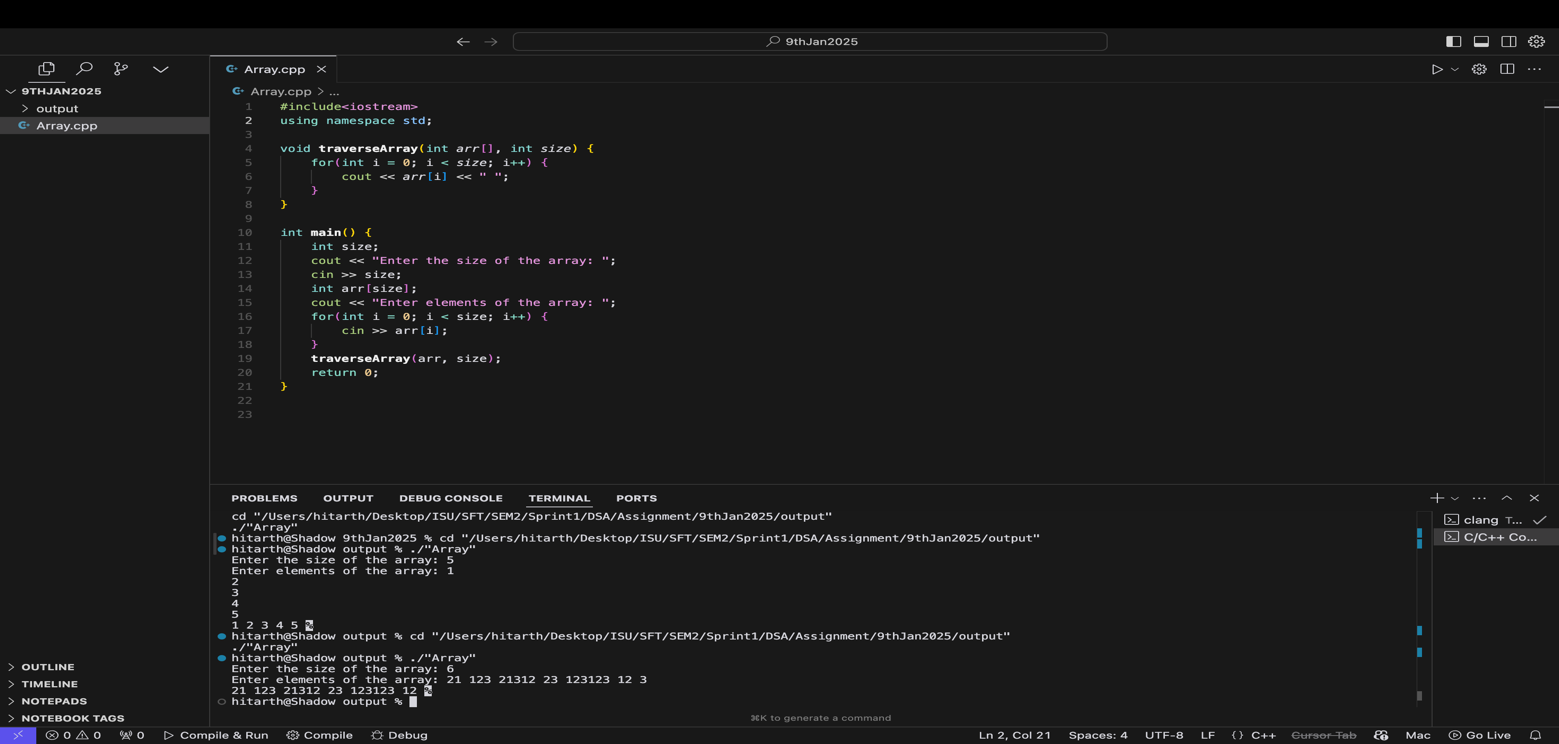
cin >> arr[i];

}

traverseArray(arr, size);

return 0;

}



**Q7. What is non linear data structure? Give an example?**

**Nonlinear Data Structure**

A nonlinear data structure is a type of data structure in which the elements are not arranged in a sequential manner. Instead, the elements are connected through multiple relationships, allowing for more complex and flexible data organization.

**Characteristics**

* **Non-Sequential Element Arrangement:** Elements are not stored in a contiguous or sequential manner.
* **Multiple Relationships Between Elements**: Elements are connected through multiple relationships, such as edges or pointers.
* **Complex Data Organization**: Nonlinear data structures allow for more complex and flexible data organization.

**Examples :**

* **Graphs:** A graph is a nonlinear data structure consisting of nodes (vertices) connected by edges.
* **Trees**: A tree is a nonlinear data structure in which nodes are connected through edges, with no cycles.
* **Hash Tables**: A hash table is a nonlinear data structure that maps keys to values using a hash function.

**Q8. What are time and space complexity?**

**Time Complexity**

Time complexity refers to the amount of time an algorithm takes to complete, usually measured in terms of the number of operations performed. It's typically expressed using Big O notation, which gives an upper bound on the number of operations.

**Space Complexity**

Space complexity refers to the amount of memory an algorithm uses, usually measured in terms of the maximum amount of memory allocated.