Data Structures

Concept

A data structure organizes and stores data in a computer system. The structure used influences how efficiently data can be accessed, processed, and updated.

Types

1. Linear Data Structures

Elements are arranged sequentially and accessed by iterating through the structure.

a. Arrays

- Collection of elements of the same type, stored contiguously in memory.
- Example: `int[] arr = {1, 2, 3}`

b. Linked Lists

- Collection of nodes, each containing data and a reference to the next node.
- Example:

```
struct Node {
  int data;
  Node *next;
};
```

c. Stacks

- Last In First Out (LIFO) order (like a stack of plates).
- Elements are added and removed from the top.
- Example:

```
class Stack {
  vector<int> elements;

  void push(int element) {
    elements.push_back(element);
  }

  int pop() {
    if (!elements.empty()) {
      int top = elements.back();
      elements.pop_back();
      return top;
    }
    return -1; // Error
  }
};
```

d. Queues

- First In First Out (FIFO) order (like a line of people).
- Elements are added to the rear and removed from the front.
- Example:

```
class Queue {
  vector<int> elements;

  void enqueue(int element) {
    elements.push_back(element);
  }

  int dequeue() {
    if (!elements.empty()) {
      int front = elements.front();
      elements.erase(elements.begin());
      return front;
    }
    return -1; // Error
  }
};
```

2. Non-Linear Data Structures

Elements are arranged in a hierarchical or graph-like manner.

a. Trees

- Hierarchical data structure with multiple interconnected nodes.
- Example:

```
struct Node {
  int data;
  vector<Node*> children;
};
```

b. Graphs

- Collection of vertices connected by edges.
- Example:

```
struct Graph {
  map<int, vector<int>> adjacencyList;

  void addEdge(int u, int v) {
    adjacencyList[u].push_back(v);
    adjacencyList[v].push_back(u);
  }
```

};

```

**c. Sets**

- Collection of unique elements.
- Example: `set<int> mySet = {1, 2, 3}`

**d. Maps**

- Collection of key-value pairs.
- Example: `map<int, string> myMap = {{1, "One"}, {2, "Two"}, {3, "Three"}};`
```