DATA STRUCTURE UNIT-1 NOTES

1. Introduction to Data Structure

What is Data?

Data means raw facts and figures.

Examples:

- Student marks
- Employee salary
- Roll number
- Name

Example:

```
85, Rahul, 1200
```

These are raw values and have no proper meaning alone.

What is Information?

Processed and organized data is called information.

Example:

```
Rahul scored 85 marks in Data Structure.
```

Now the data has proper meaning.

Difference Between Data and Information

Data	Information
Raw facts	Processed data
No meaning alone	Meaningful
Unorganized	Organized

---

2. What is Data Structure?

A Data Structure is a way of organizing and storing data efficiently in computer memory.

It helps in:

- Fast access
- Easy modification
- Efficient processing

Definition

> Data Structure is a systematic way of organizing data in computer memory.

Real Life Example

Bookshelf Example

- Books arranged subject-wise
- Easy searching
- Easy insertion/removal

Same concept used in data structures.

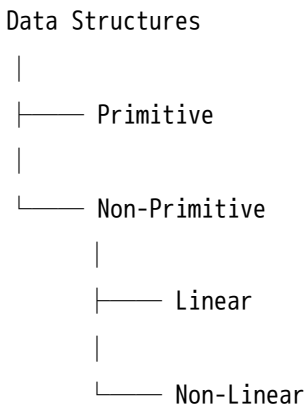
Applications of Data Structures

- Database management
- Operating systems
- Web browsers
- Artificial Intelligence
- Compiler design
- Social media applications

---

3. Classification of Data Structures

Data Structures are mainly divided into:



(A) Primitive Data Structures

Basic data types provided by programming language.

Examples:

- int
- char
- float
- double

(B) Non-Primitive Data Structures

More complex data structures.

Examples:

- Array
- Linked List
- Stack
- Queue
- Tree
- Graph

Linear Data Structure

Data arranged sequentially.

Examples:

- Array
- Linked List
- Stack
- Queue

Non-Linear Data Structure

Data arranged hierarchically.

Examples:

- Tree
- Graph

---

4. Abstract Data Type (ADT)

Definition

ADT is a logical description of data and operations without implementation details.

It tells:

- What operations can be performed
- Not how operations are implemented

Example of Stack ADT

Operations:

- Push
- Pop
- Peek

Implementation may use:

- Array
- Linked List

Advantages of ADT

- Reduces complexity

- Improves code reusability
- Provides abstraction

---

5. Memory Representation

Data structures are stored in computer memory.

Memory is divided into:

- RAM
- Cache
- Registers

Types of Memory Allocation

1. Static Memory Allocation

Memory allocated during compile time.

Example:

```
int arr[5];
```

Fixed size.

---

2. Dynamic Memory Allocation

Memory allocated during runtime.

Functions used:

- malloc()
- calloc()
- realloc()
- free()

Example:

```
ptr = (int*)malloc(5*sizeof(int));
```

---

6. Operations on Data Structures

---

1. Traversing

Accessing all elements one by one.

Example:

Printing array elements.

---

2. Insertion

Adding new element.

---

3. Deletion

Removing element.

---

4. Searching

Finding element.

---

5. Sorting

Arranging data.

Example:

Ascending order.

---

6. Merging

Combining two data structures.

---

7. Cost Estimation of Operations

Cost means time taken for operations.

Measured using:

- Time Complexity
- Space Complexity

Time Complexity

Amount of time required.

Examples:

- O(1)
- O(n)
- O(log n)

Space Complexity

Amount of memory used.

Example

Linear Search:

O(n)O(n)O(n)

Binary Search:

O(logn)O(\log n)O(logn)

---

8. Introduction to Linear Data Structures

Linear data structure stores data sequentially.

Examples:

- Array
- Linked List
- Stack
- Queue

Advantages

- Easy implementation
- Sequential access
- Simple operations

---

9. Arrays

Definition

Array is a collection of similar data items stored at contiguous memory locations.

Syntax in C

```
int arr[5];
```

Example

```
int arr[5]={10,20,30,40,50};
```

Memory Representation of Array

```
Address      Value

1000         10
```

```
1004        20
1008        30
1012        40
1016        50
```

Elements stored continuously.

Advantages of Array

- Fast access
- Easy traversal

Disadvantages of Array

- Fixed size
- Insertion/deletion difficult

Types of Arrays

1. One Dimensional Array

```
int arr[5];
```

---

2. Two Dimensional Array

```
int arr[3][3];
```

Used for matrix.

---

10. Linked List

Definition

Linked List is a linear data structure where elements are connected using pointers.

Each node contains:

1. Data
2. Address of next node

Structure of Node

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

Representation of Linked List

```
10 → 20 → 30 → NULL
```

Advantages of Linked List

- Dynamic size
- Easy insertion/deletion

Disadvantages

- Extra memory for pointer
- Sequential access only

---

11. Representation of Linked List in Memory

Nodes are not stored continuously.

Example:

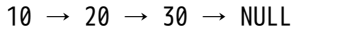
```
Address      Data      Next

1000          10       2000
2000          20       3000
3000          30       NULL
```

---

12. Types of Linked List

(A) Singly Linked List

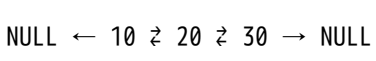
Each node points to next node only.



(B) Doubly Linked List

Each node has:

- Previous pointer
- Next pointer

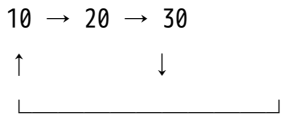


Advantages

- Traversal possible in both directions.

(C) Circular Linked List

Last node points to first node.



Advantages

- Efficient memory usage
- Used in CPU scheduling

---

13. Applications of Linked List

---

1. Polynomial Manipulation

Polynomial represented using nodes.

Example:

5x2+3x+25x^2 + 3x + 25x2+3x+2

Each term stored in separate node.

---

2. Dynamic Memory Allocation

Used in memory management.

---

3. Music Playlist

Songs connected dynamically.

---

4. Browser History

Back and forward operations.

---

14. Polynomial Manipulation Using Linked List

Polynomial terms stored as nodes.

Each node contains:

- Coefficient
- Power
- Pointer

Example

5x2+3x+25x^2 + 3x + 25x2+3x+2

Representation:

```
[5,2] → [3,1] → [2,0]
```

Advantages

- Dynamic storage
- Easy addition/subtraction

---

15. Difference Between Array and Linked List

Array	Linked List
Fixed size	Dynamic size
Contiguous memory	Non-contiguous memory
Fast access	Slow access
Difficult insertion	Easy insertion

Important Viva Questions

1. What is data structure?
2. Difference between data and information.
3. What is ADT?
4. Difference between array and linked list.
5. What is dynamic memory allocation?
6. What is linked list?
7. Types of linked list.
8. Advantages of linked list.

Important 7-Mark Questions

1. Explain classification of data structures.
2. Explain ADT with example.
3. Explain operations on data structures.
4. Explain linked list with memory representation.
5. Difference between array and linked list.

Important 14-Mark Questions

1. Explain arrays and linked lists in detail.
2. Explain types of linked list with diagrams.
3. Explain data structure classification with examples.

4. Explain memory representation and operations on data structures.
5. Explain polynomial manipulation using linked list.

Quick Revision

Array

- Fixed size
- Continuous memory
- Fast access

Linked List

- Dynamic size
- Pointer based
- Easy insertion/deletion

ADT

Defines operations only.

Time Complexity

Measures execution time.

Conclusion

Data Structures help in efficient data storage and processing.

Unit-1 forms the foundation of:

- Algorithms
- Operating Systems
- Databases
- Artificial Intelligence
- Software Development

Understanding Arrays and Linked Lists is very important for coding interviews and programming.