DATA STRUCTURES

UNIT - I Linear Structures

Introduction :-

The method of manipulating large amount of data in a better way Data structure is a systematic way to organize data in order to use it efficiently. Following terms are foundation terms of a data structure.

* Interface
* Implementation

Interface :-

Each data structure has an interface. It represents set of operations, type of parameters they can accept and return type of these operations.

Implementation:-

It provides the internal representation of data structure. Implementation also provides the definition of algorithms used in the operations of data structure.

Characteristics of a Data Structure

* Correctness - Data structure Implementation should implement its interface correctly.
* Time Complexity - Running time or execution time of operations of data structure must be as small as possible.
* Space Complexity - Memory usage of a data structure operation should be as little as possible.

Need for Data structure

* Data search
* processor speed
* Multiple requests

To improve data search, processor speed & respond users multiple requests we need data structure.

Basic Terminolgies

Data - collection of facts, concepts, figures, abbreviations. occurences (or) instructions in a formalized manner.

Information — It is the processed data

Record - Collection of related fields.

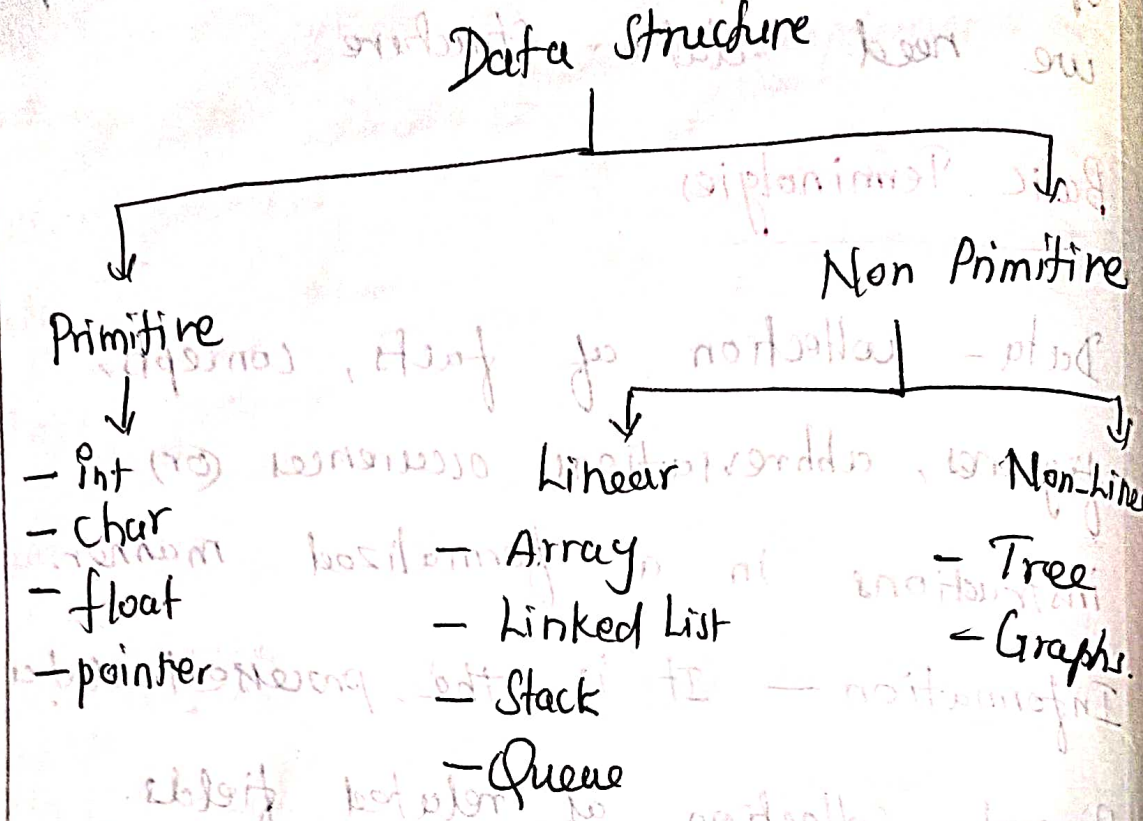
Definition of Data Structures

A way of organizing, Storing and retrieving data's and their relationship with each other.

Applications of Data Structure:-

* operating systems
* Compiler Design
* statics and numerical Analysis
* Network Analysis
* Expert systems
* Database Management systems.

Types of Data Structures



Primitive Data Structures

It is a basic data structures which can be directly operated by machine instruction.

Non Primitive Data Structures

Data structure emphasis on structuring of group of heterogeneous or homogeneous data items.

Linear Data structures

Representing data in order or arranging element in order.

Non Linear DS

Hierarchical representation of data.

Abstract Data Type (ADT)

→ Extension of modular design

→ Break the program down into modules.

→ A modules are logical unit, small in size and does a specific job.

Advantage of modular design

→ Make changes easier

→ people to work on modules simultaneously.

→ Easier to debug.

Definition:

ADT is a set of operations and mathematical abstraction but not detail about how the set of operations is implemented. This can be viewed as extension of modular design.

Language supporting ADT

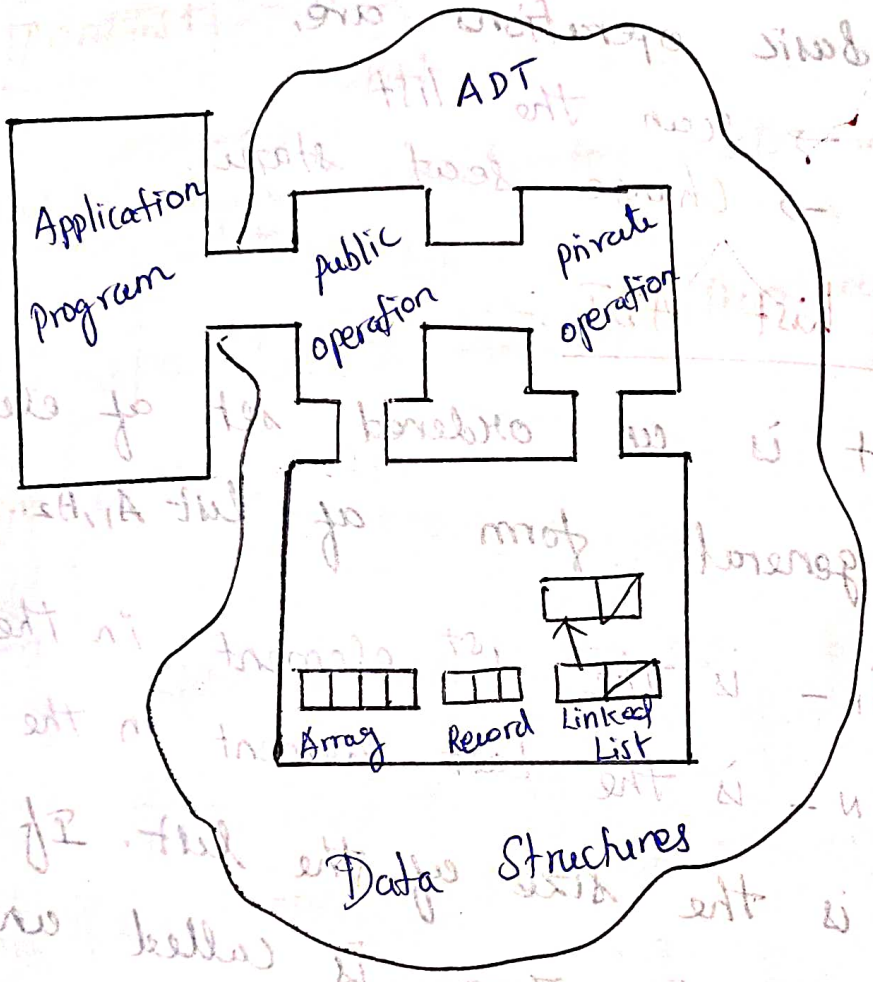
- C
- C++
- Java
- python
- C#

Function of ADT

→ What can be done with the data not how it is done.

An abstract data type is a data declaration packaged together with the operations that are meaningful on the data type.

In other words we encapsulate the data and the operations on data and we hide them from the user.



Implementation of ADT

* Declaration of Data
* Declaration of operations

→ Storage to store data items.
→ Algorithm for basic operations.

Eg Airplane flight 10 seats to be assigned.

→ collection of data items is the list of seats

→ Basic operations are,

→ scan the list

→ change seat status.

The list ADT :-

List is an ordered set of elements.

The general form of list $A_1, A_2 \ldots A_N$

A_1 - is the 1st element in the list

A_N - is the last element in the list

N - is the size of the list. If size of N is zero, is called an empty list

→ Element with the operation 'i' is A_i

→ Successor of $A_i \rightarrow A_{i+1}$

→ Predecessor of $A_i \rightarrow A_{i-1}$