Data Structures and Algorithm

(CSE 225)

Lecture 1

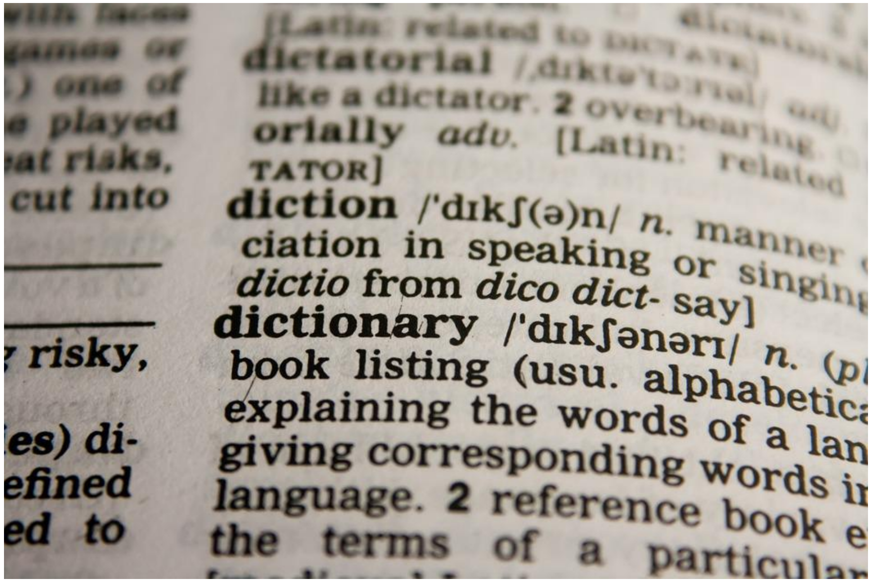
RECOMMENDED BOOKS

- C++ plus data Structures, Fifth Edition by Nell Dale
- Data Structures with C++ Schaum’s Outline Series

Introduction to Data Structures

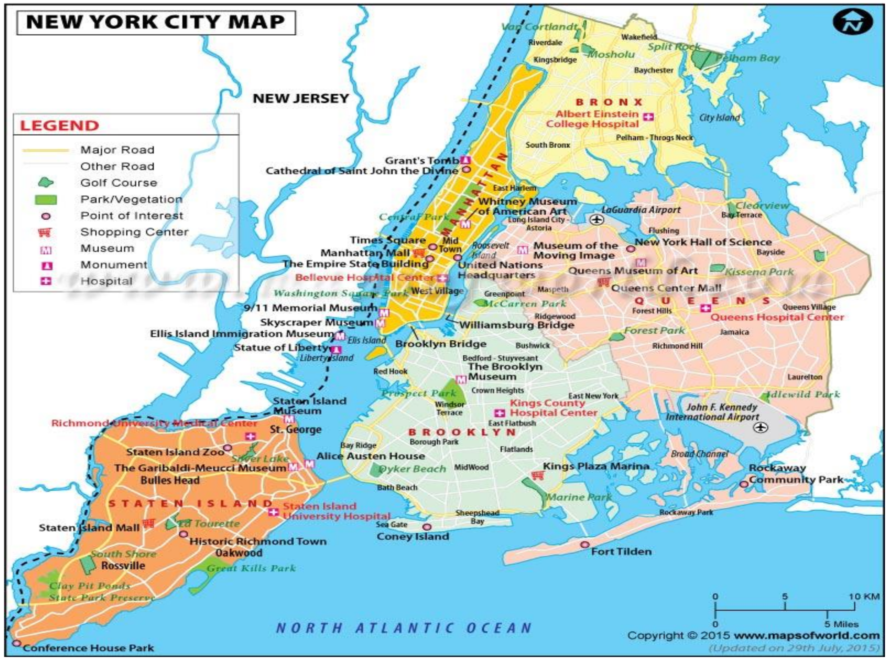
- One of the most **fundamental** courses in Computer Science
- Good knowledge of Data Structures is a must to design and develop efficient software system
- We deal with data all the time and how we **store**, **organize** and **group** our data together matters
- Let’s pick up some examples from our day-to-day life

Examples

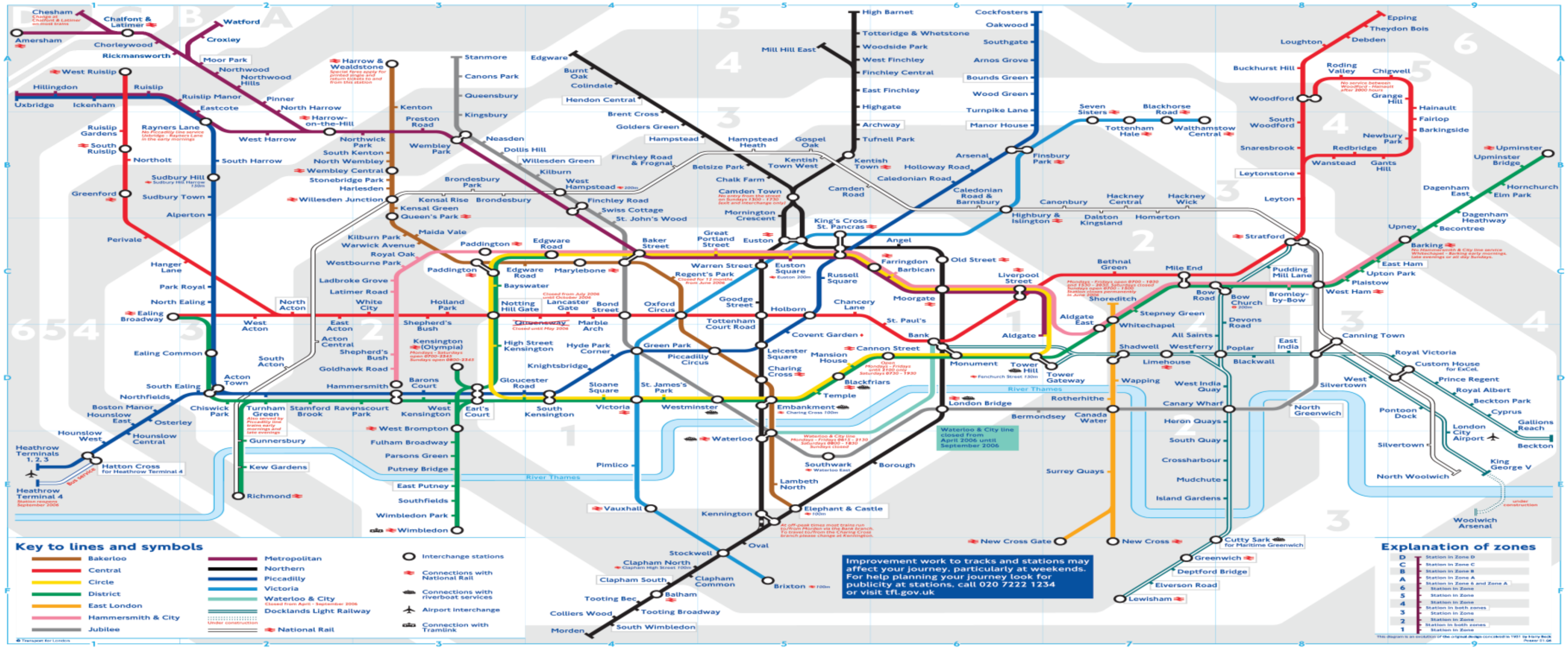


with faces
games or
) one of
e played
at risks,
cut into
y risky,
es) di-
efined
ed to

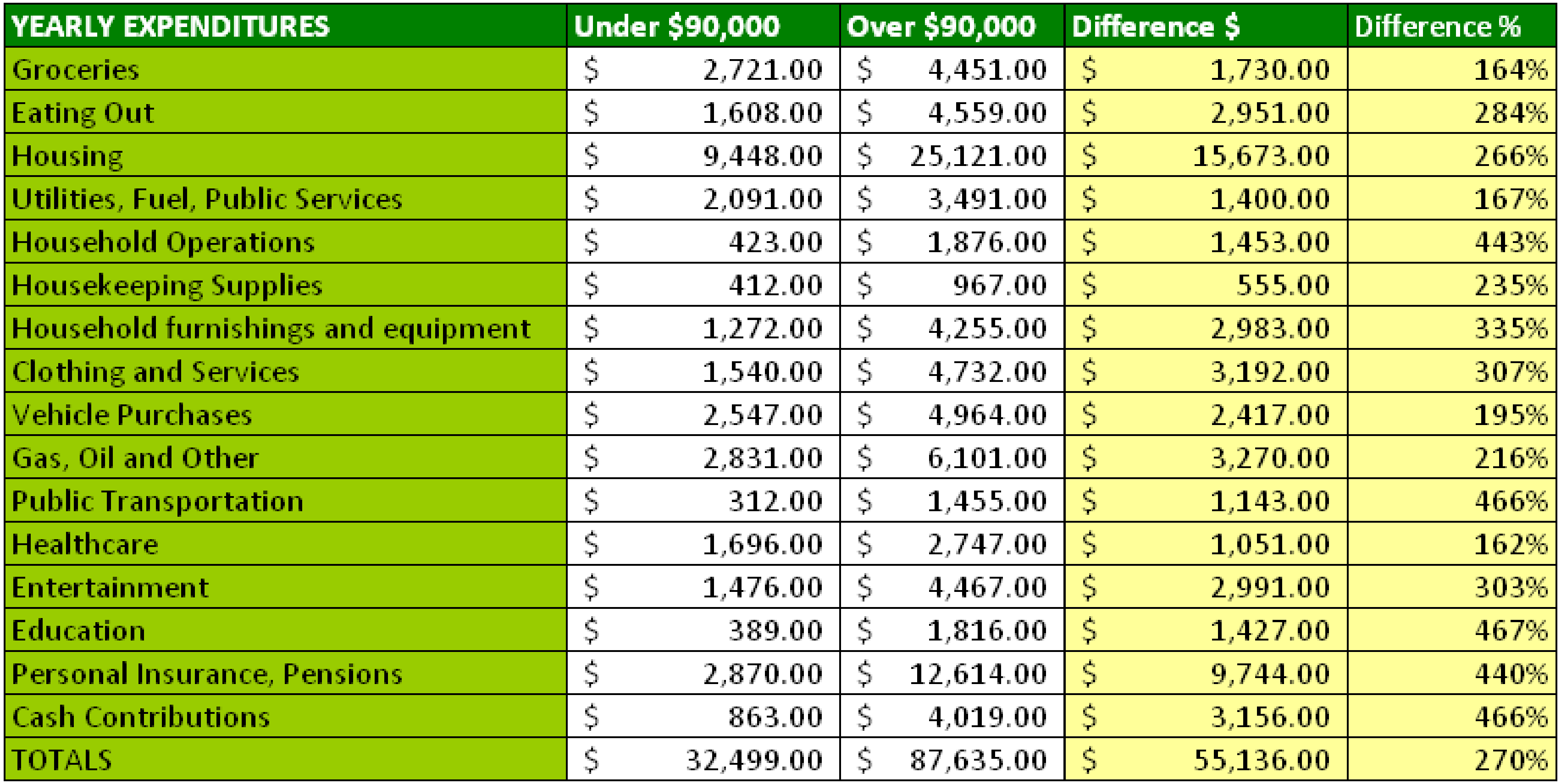
[Latin: related to DICTATE]
dictatorial /ˌdɪktəˈtɔːrɪəl/ adj.
like a dictator. 2 overbearing.
orially adv. [Latin: related
TATOR]
diction /ˈdɪkʃ(ə)n/ *n.* manner
ciation in speaking or singin
dictio from *dico dict-* say]
dictionary /ˈdɪkʃənərɪ/ *n.* (pl
book listing (usu. alphabetica
explaining the words of a lan
giving corresponding words in
language. **2** reference book e
the terms of a particular



Examples


Chesham
Chalfont & Latimer
Watford
Amersham
Croxley
Chorleywood
Rickmansworth
Moor Park
Northwood
Northwood Hills
West Ruislip
Pinner
Hillingdon
Ruislip
Ruislip Manor
Uxbridge
Ickenham
Eastcote
North Harrow
Harrow-on-the-Hill
Harrow & Wealdstone
Ruislip Gardens
Rayners Lane
West Harrow
South Ruislip
Northolt
South Harrow
Northwick Park
Preston Road
Kenton
Wembley Park
South Kenton
North Wembley
Wembley Central
Stonebridge Park
Harlesden
Willesden Junction
Sudbury Hill
Sudbury Town
Greenford
Alperton
Perivale
Hanger Lane
Park Royal
North Ealing
Ealing Broadway
North Acton
West Acton
East Acton
Ealing Common
Acton Central
South Acton
Acton Town
South Ealing
Northfields
Boston Manor
Hounslow East
Osterley
Hounslow West
Hounslow Central
Heathrow Terminals 1, 2, 3
Heathrow Terminal 4
Hatton Cross
Chiswick Park
Turnham Green
Stamford Brook
Ravenscourt Park
Gunnersbury
Kew Gardens
Richmond
Stanmore
Canons Park
Queensbury
Kingsbury
Edgware
Burnt Oak
Colindale
Hendon Central
Brent Cross
Golders Green
Hampstead
Neasden
Dollis Hill
Willesden Green
Kilburn
West Hampstead
Finchley Road
Swiss Cottage
St. John's Wood
Brondesbury Park
Kensal Rise
Brondesbury
Kensal Green
Queen's Park
Kilburn Park
Maida Vale
Warwick Avenue
Royal Oak
Westbourne Park
Paddington
Ladbroke Grove
Latimer Road
White City
Shepherd's Bush
Holland Park
Notting Hill Gate
Queensway
Lancaster Gate
Marble Arch
Bond Street
Edgware Road
Bayswater
Kensington (Olympia)
High Street Kensington
Hyde Park Corner
Knightsbridge
Goldhawk Road
Hammersmith
Barons Court
West Kensington
Earl's Court
Gloucester Road
South Kensington
Sloane Square
Victoria
West Brompton
Fulham Broadway
Parsons Green
Putney Bridge
East Putney
Southfields
Wimbledon Park
Wimbledon
Mill Hill East
High Barnet
Totteridge & Whetstone
Woodside Park
West Finchley
Finchley Central
East Finchley
Highgate
Archway
Tufnell Park
Hampstead Heath
Gospel Oak
Finchley Road & Frognal
Belsize Park
Chalk Farm
Kentish Town West
Kentish Town
Camden Town
Camden Road
Mornington Crescent
Baker Street
Marylebone
Great Portland Street
Euston
Regent's Park
Warren Street
Euston Square
Oxford Circus
Goodge Street
Tottenham Court Road
Russell Square
Holborn
Chancery Lane
Green Park
Piccadilly Circus
Leicester Square
Covent Garden
Charing Cross
St. James's Park
Westminster
Embankment
Temple
Blackfriars
Mansion House
Cannon Street
Waterloo
Pimlico
Vauxhall
Lambeth North
Southwark
Borough
Elephant & Castle
Kennington
Oval
Stockwell
Brixton
Clapham North
Clapham Common
Clapham South
Balham
Tooting Bec
Tooting Broadway
Colliers Wood
South Wimbledon
Morden
King's Cross St. Pancras
Angel
Farringdon
Barbican
Old Street
Moorgate
Liverpool Street
St. Paul's
Bank
Monument
Tower Hill
Tower Gateway
London Bridge
Aldgate
Aldgate East
Cockfosters
Oakwood
Southgate
Arnos Grove
Bounds Green
Wood Green
Turnpike Lane
Manor House
Finsbury Park
Arsenal
Holloway Road
Caledonian Road
Caledonian Road & Barnsbury
Highbury & Islington
Canonbury
Dalston Kingsland
Hackney Central
Hackney Wick
Homerton
Seven Sisters
Tottenham Hale
Blackhorse Road
Walthamstow Central
Bethnal Green
Mile End
Shoreditch
Whitechapel
Stepney Green
Bow Road
Bow Church
Devons Road
Bromley-by-Bow
Pudding Mill Lane
Stratford
Shadwell
Wapping
Rotherhithe
Canada Water
Surrey Quays
New Cross Gate
New Cross
Bermondsey
Limehouse
Westferry
All Saints
Poplar
Blackwall
East India
West India Quay
Canary Wharf
Heron Quays
South Quay
Crossharbour
Mudchute
Island Gardens
Cutty Sark
Greenwich
Deptford Bridge
Elverson Road
Lewisham
Epping
Theydon Bois
Debden
Loughton
Buckhurst Hill
Roding Valley
Chigwell
Grange Hill
Hainault
Fairlop
Barkingside
Woodford
South Woodford
Snaresbrook
Newbury Park
Redbridge
Gants Hill
Wanstead
Leytonstone
Leyton
Upminster
Upminster Bridge
Hornchurch
Elm Park
Dagenham East
Dagenham Heathway
Becontree
Upney
Barking
East Ham
Upton Park
Plaistow
West Ham
Canning Town
Royal Victoria
Custom House for ExCeL
Prince Regent
Royal Albert
Beckton Park
Cyprus
Gallions Reach
Beckton
West Silvertown
Pontoon Dock
London City Airport
King George V
Silvertown
North Woolwich
Woolwich Arsenal
North Greenwich
River Thames
Key to lines and symbols
Bakerloo
Central
Circle
District
East London
Hammersmith & City
Jubilee
Metropolitan
Northern
Piccadilly
Victoria
Waterloo & City
Docklands Light Railway
National Rail
Interchange stations
Connections with National Rail
Connections with riverboat services
Airport interchange
Connection with Tramlink
Explanation of zones
Improvement work to tracks and stations may affect your journey, particularly at weekends. For help planning your journey look for publicity at stations, call 020 7222 1234 or visit tfl.gov.uk

Examples

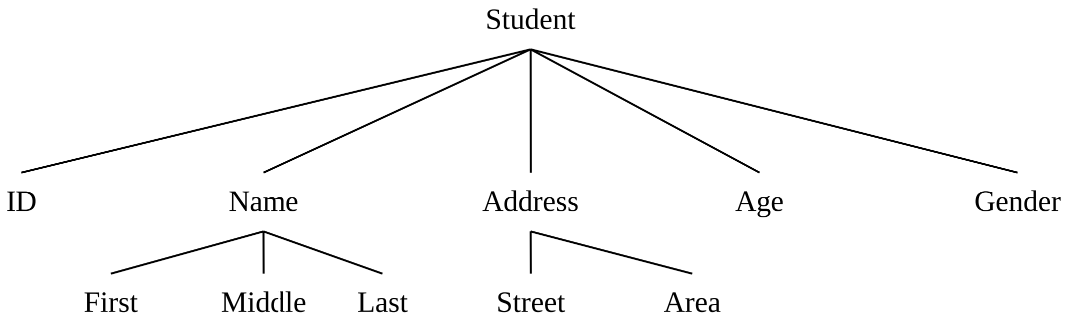


YEARLY EXPENDITURES	Under $90,000	Over $90,000	Difference $	Difference %
Groceries	$ 2,721.00	$ 4,451.00	$ 1,730.00	164%
Eating Out	$ 1,608.00	$ 4,559.00	$ 2,951.00	284%
Housing	$ 9,448.00	$ 25,121.00	$ 15,673.00	266%
Utilities, Fuel, Public Services	$ 2,091.00	$ 3,491.00	$ 1,400.00	167%
Household Operations	$ 423.00	$ 1,876.00	$ 1,453.00	443%
Housekeeping Supplies	$ 412.00	$ 967.00	$ 555.00	235%
Household furnishings and equipment	$ 1,272.00	$ 4,255.00	$ 2,983.00	335%
Clothing and Services	$ 1,540.00	$ 4,732.00	$ 3,192.00	307%
Vehicle Purchases	$ 2,547.00	$ 4,964.00	$ 2,417.00	195%
Gas, Oil and Other	$ 2,831.00	$ 6,101.00	$ 3,270.00	216%
Public Transportation	$ 312.00	$ 1,455.00	$ 1,143.00	466%
Healthcare	$ 1,696.00	$ 2,747.00	$ 1,051.00	162%
Entertainment	$ 1,476.00	$ 4,467.00	$ 2,991.00	303%
Education	$ 389.00	$ 1,816.00	$ 1,427.00	467%
Personal Insurance, Pensions	$ 2,870.00	$ 12,614.00	$ 9,744.00	440%
Cash Contributions	$ 863.00	$ 4,019.00	$ 3,156.00	466%
TOTALS	$ 32,499.00	$ 87,635.00	$ 55,136.00	270%

Data Structures

❑ **Data**

- Data are simply values or sets of values, raw materials used as inputs.
- A data item refers to a single unit of values.
- Data items that are divided into sub items are group items. Those that are not are called elementary items. For example, a student's name may be divided into three sub items – [first name, middle name and last name] but the ID of a student would normally be treated as a single item.



Student

ID Name Address Age Gender

First Middle Last Street Area

Data and Information

- **Data** can be defined as a representation of facts and concepts by values.
- **Data** is collection of **raw facts**.

Data is represented with the help of characters such as alphabets (A-Z, a-z), digits (0-9) or special characters (+,-,/,*,<,>,= etc.)

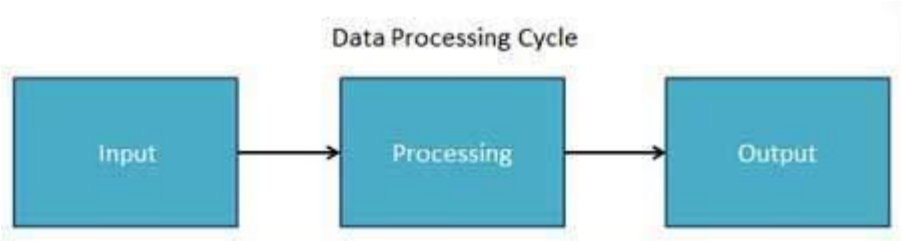
- **Data structure** is representation of the logical relationship existing between individual elements of data.

Information

- **Information** is organized or classified data, which has some meaningful values for the receiver. Information is the processed data on which decisions and actions are based.

- The processed data must qualify for the following characteristics
  - **Timely** – Information should be available when required.
  - **Accuracy** – Information should be accurate.
  - **Completeness** – Information should be complete.

Data Processing Cycle



- **Input:** the input data is prepared in some convenient form for processing
- **Processing:** the input data is changed to produce data in a more useful form
- **Output:** the result of the proceeding processing step is collected.

Data Structures

❑What is Data Structure?

- A data structure is defined as a particular way of **storing** and **organizing** data in our devices to use the data **efficiently** and **effectively**. The main idea behind using data structures is to minimize the **time** and **space complexities**. An efficient data structure takes minimum memory space and requires minimum time to execute the data.

How do we study data structures?

- When we study data structures, we study them in two ways:
  - Mathematical / logical models (we look at an abstract view of them or **ADT**)
  - Implementations of the ADT

ADT

Barbara Liskov is an American Computer Scientist and a Professor at MIT. She was the first woman to get a PhD in Computer Science at USA and is also a Turing Award winner for developing Liskov Substitution Principle.

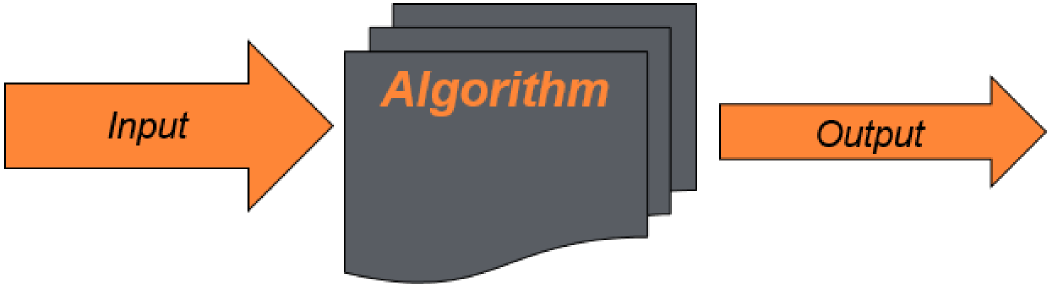
Barbara, along with Stephen N. Zilles first proposed the concepts of ADTs.

List (ADT)

- List (ADT)
  - Store a given number of elements of any data type
  - Read elements by position
  - Modify element at any position
- **Arrays** (Concrete Implementation)
- **Linked List** (Concrete Implementation)
- Abstract Data types are entities that are definition of data and operations but do not have implementations

What is an algorithm?

- A computational procedure that takes
  - ❑ some value, or set of values, as ***input***
  - ❑ produces some value, or set of values, as ***output***
  - ❑ may be specified:
    - o In English
    - o As a pseudocode
    - o As a computer program



- A sequence of computational steps that transform the input into the output.

What is an algorithm?

- **Algorithm** are the systematic ordered logical approach which is a well-defined, step-by-step procedure that allows a computer to solve a problem.

- **Algorithm** must
  - It must produce the correct result
  - It must finish in some finite time

- Algorithms are the ideas behind computer programs.

- An algorithm is the thing that stays the same whether the program is in Pascal running on a Windows or is in JAVA running on a Macintosh!

Algorithm

- ❑An Algorithm is a well defined list of steps to solve a problem.
- ❑Data structure is the logical or mathematical relationship of individual elements of data.

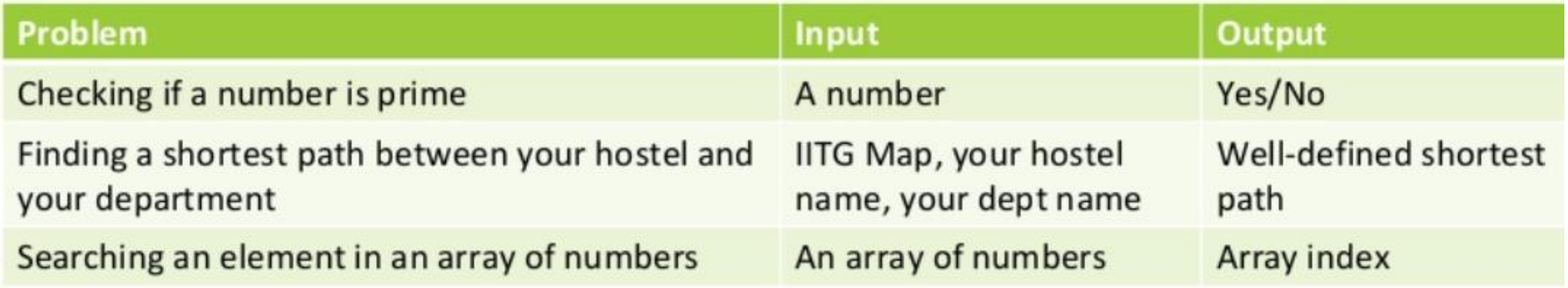
- ❑Algorithm + Data Structure= Program

Why Data Structures?

- They are essential ingredients in creating fast and powerful algorithms.
- They help to manage the organize data.
- They make code cleaner and easier to understand.

Algorithm

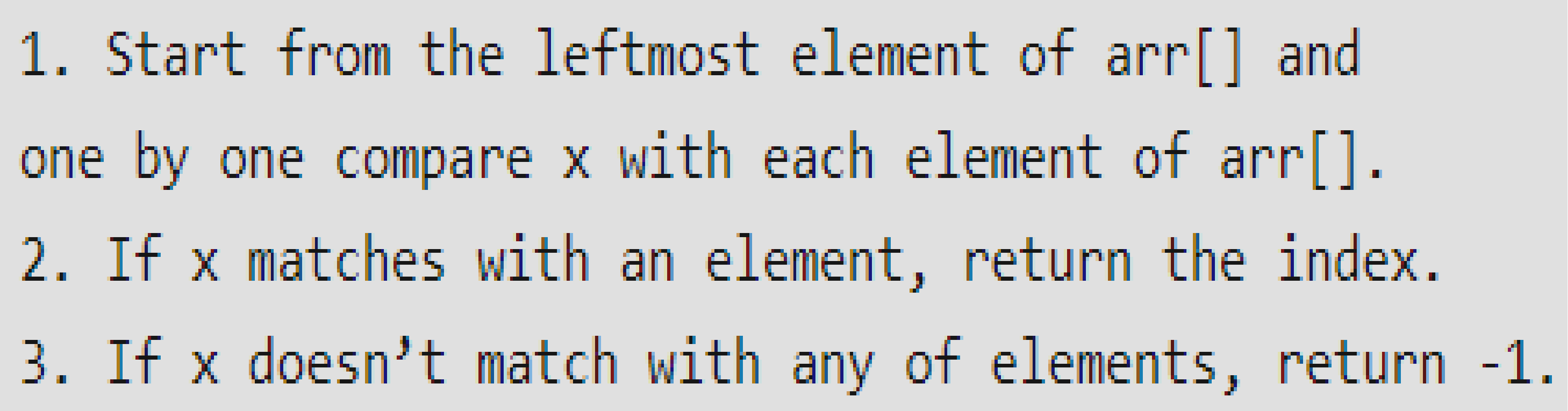
- Examples



Problem	Input	Output
Checking if a number is prime	A number	Yes/No
Finding a shortest path between your hostel and your department	IITG Map, your hostel name, your dept name	Well-defined shortest path
Searching an element in an array of numbers	An array of numbers	Array index

Algorithm

- **Linear search algorithm**



```
1. Start from the leftmost element of arr[] and
one by one compare x with each element of arr[].
2. If x matches with an element, return the index.
3. If x doesn’t match with any of elements, return -1.
```

Pseudocode

- High-level description of an algorithm
- More structured than English prose
- Less detailed than a program
- Preferred notation for describing algorithms
- Hides program design issues

Example: find max element of an array

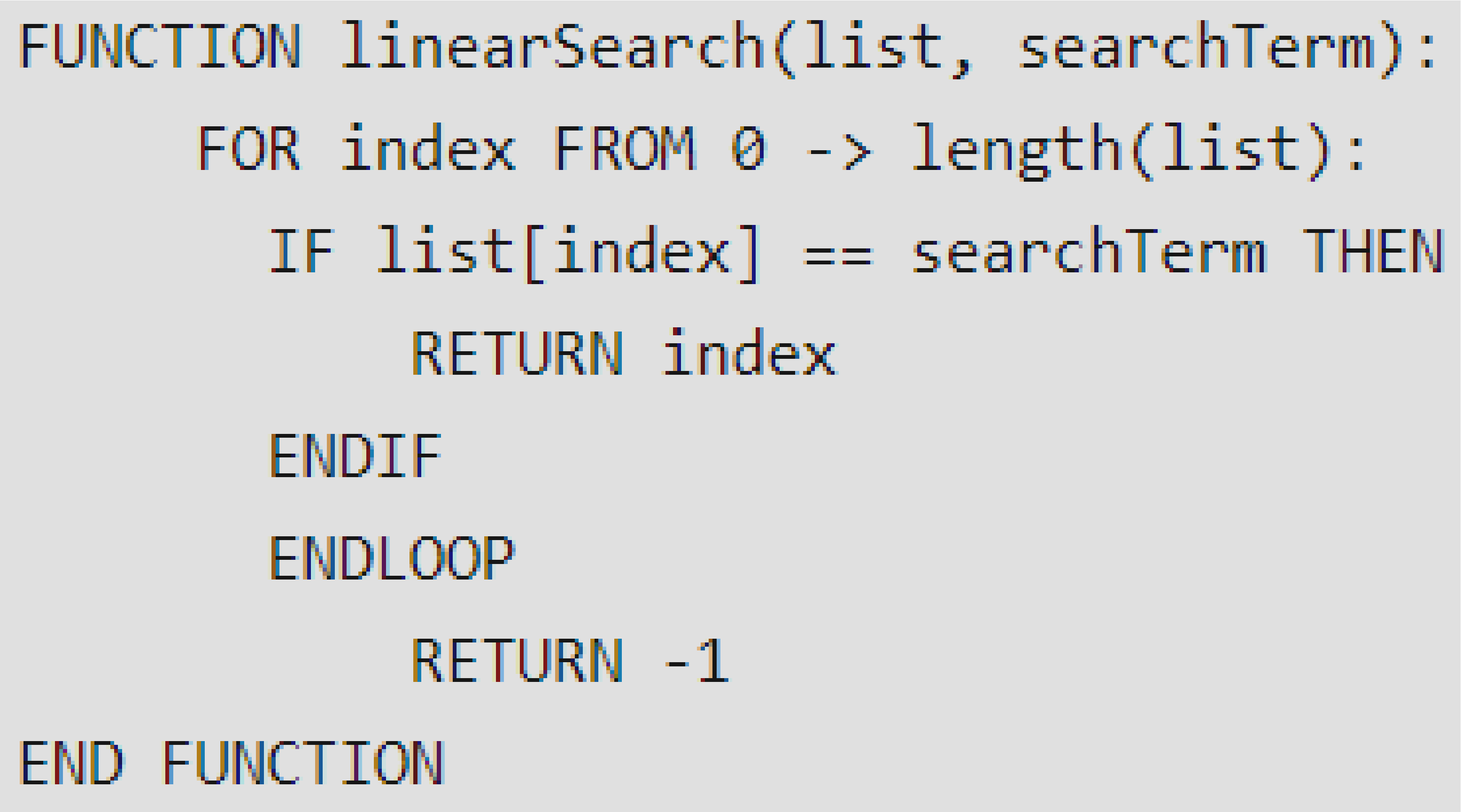
```
Algorithm $arrayMax(A, n)$
    Input array $A$ of $n$ integers
    Output maximum element of $A$

    $currentMax \leftarrow A[0]$
    for $i \leftarrow 1$ to $n - 1$ do
        if $A[i] > currentMax$ then
            $currentMax \leftarrow A[i]$
    return $currentMax$
```

Pseudocode

- It is one of the methods which can be used to represent an algorithm for a program.
- Many time algorithms are presented using pseudocode since they can be read and understood by programmers who are familiar with different programming languages.
- Pseudocode allows us to include several control structures such as **While, If-then-else, Repeat-until, for and case**, which is present in many high-level languages.

- **Note:** Pseudocode is **not** an actual programming language.

Pseudocode

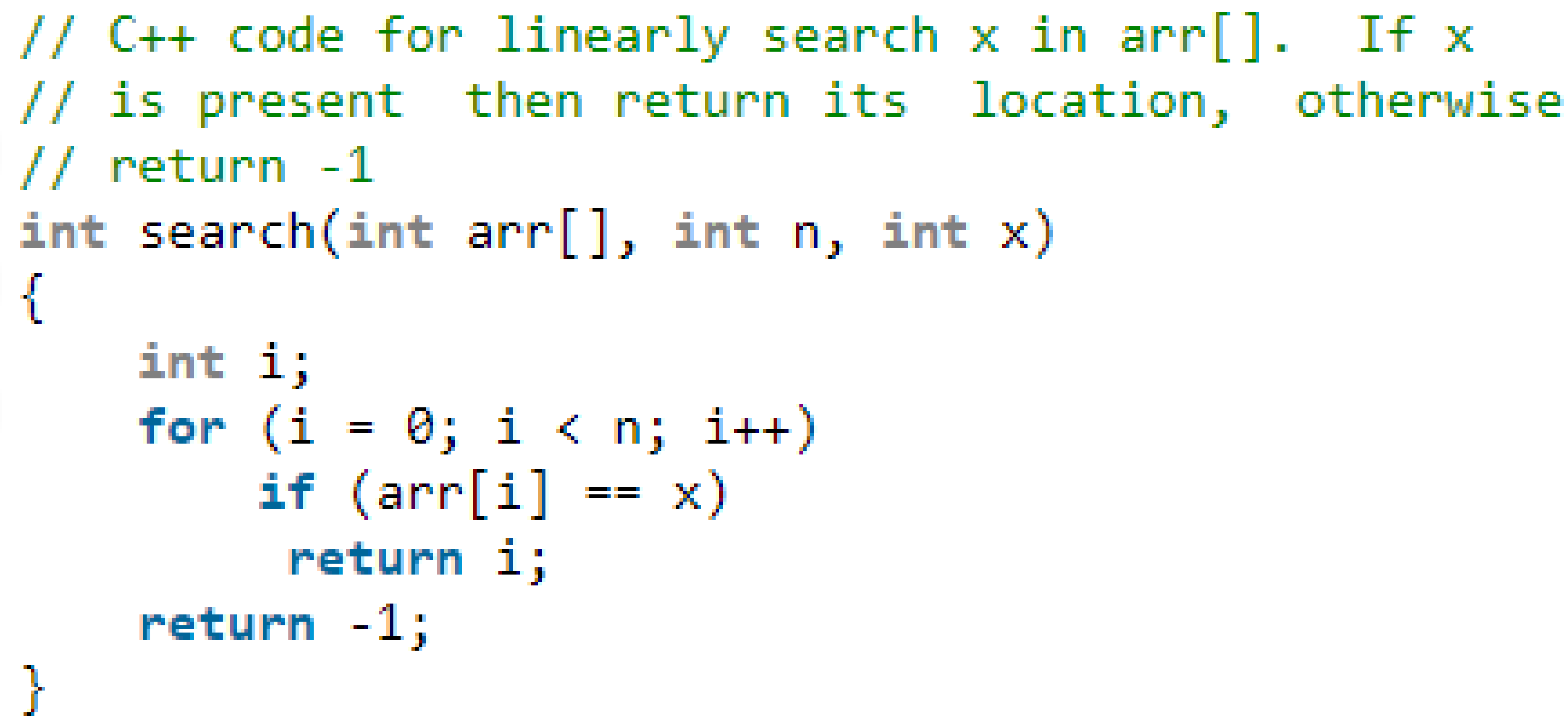


```
FUNCTION linearSearch(list, searchTerm):
     FOR index FROM 0 -> length(list):
       IF list[index] == searchTerm THEN
           RETURN index
       ENDIF
       ENDLOOP
           RETURN -1
END FUNCTION
```

Program

- A program is a set of ordered instructions for the computer to follow.
- The machine can’t read a program directly, because it only understands machine code. But we can write stuff in a computer language, and then a compiler or interpreter can make it understandable to the computer.

Program



```
// C++ code for linearly search x in arr[].  If x
// is present  then return its  location,  otherwise
// return -1
int search(int arr[], int n, int x)
{
    int i;
    for (i = 0; i < n; i++)
        if (arr[i] == x)
         return i;
    return -1;
}
```

Algorithm vs Pseudocode vs Program

Algorithm

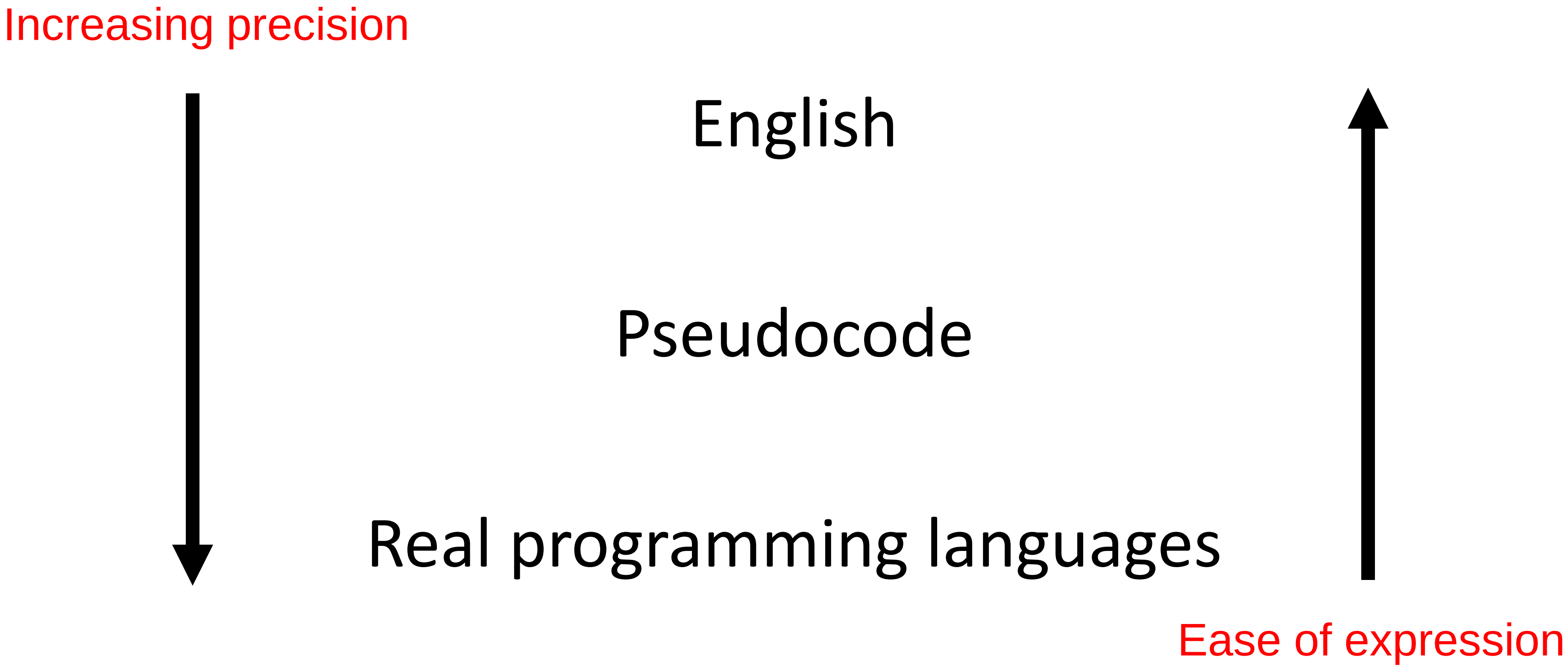
- An algorithm is defined as a well-defined sequence of steps that provides a solution for a given problem.
- Algorithms are generally written in a natural language or plain English language.

Pseudocode

- Pseudocode is one of the methods that can be used to represent an algorithm.
- Pseudocode is written in a format that is similar to the structure of a high-level programming language.

Program on the other hand allows us to write a code in a particular programming language.

How to express algorithms?



Increasing precision

English

Pseudocode

Real programming languages

Ease of expression

Classification of Data Structure

- Two broad categories of data structure are :
  - **Primitive** Data Structure
  - **Non-Primitive** Data Structure

Primitive Data Structure

- They are basic structures and directly operated upon by the machine instructions.

- Integer, Floating-point number, Character constants, string constants, pointers etc.

Non-Primitive Data Structure

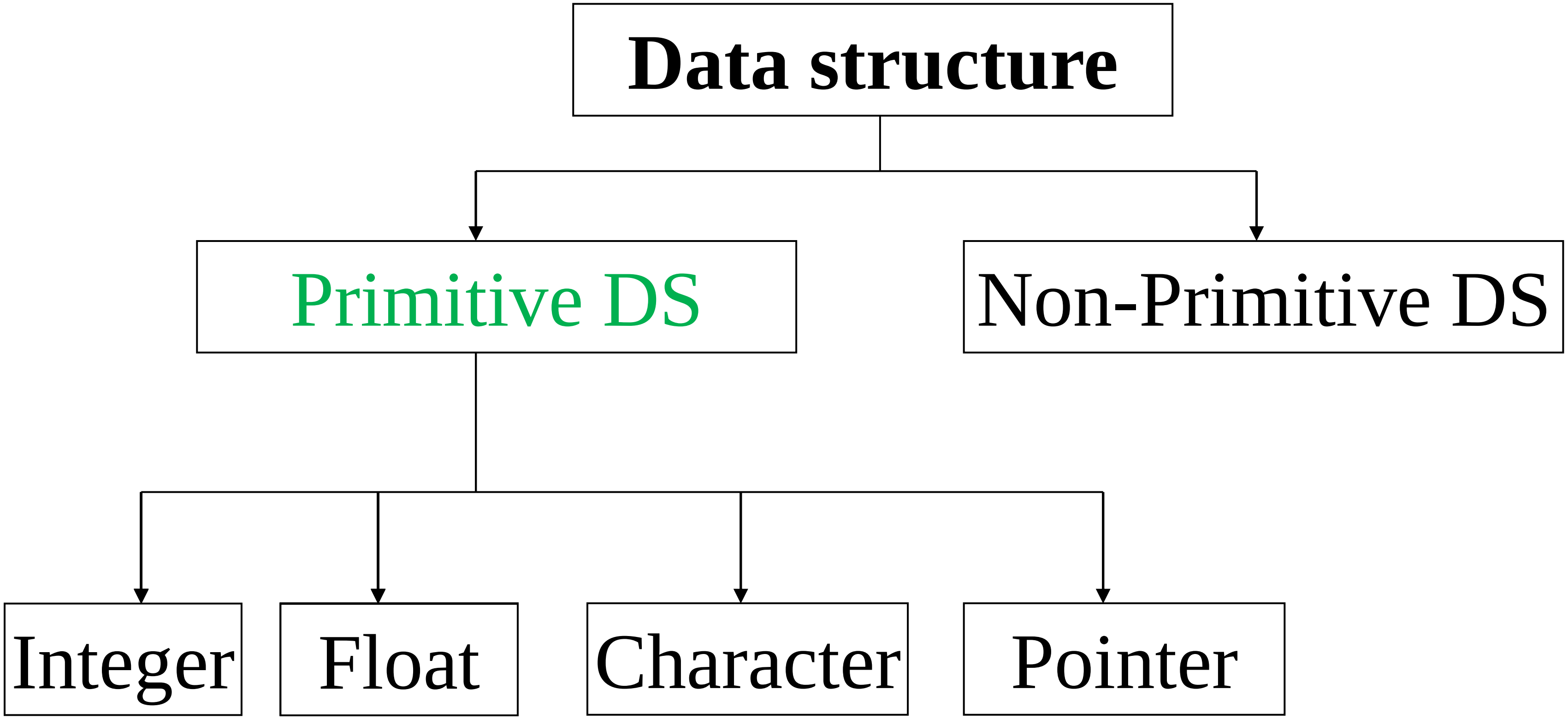
- These are derived from the primitive data structures.
- Example: Array, Lists, Stack, Queue, Tree, Graph

Difference between them

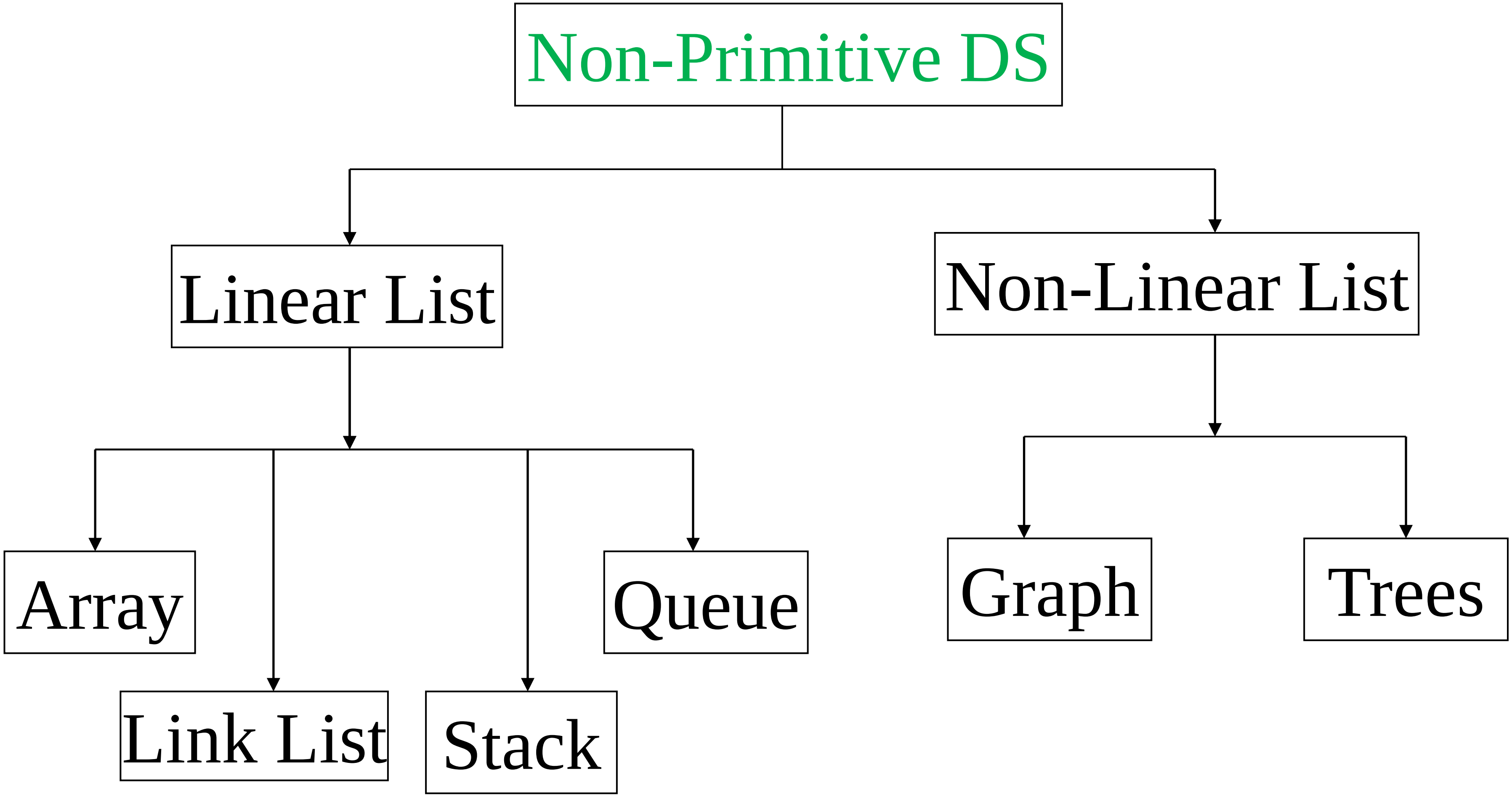
- **A primitive data structure** is generally a basic structure that is usually built into the language, such as an integer, a float.

- **A non-primitive data structure** is built out of primitive data structures linked together in meaningful ways, such as a linked-list, binary search tree, AVL Tree, graph etc.

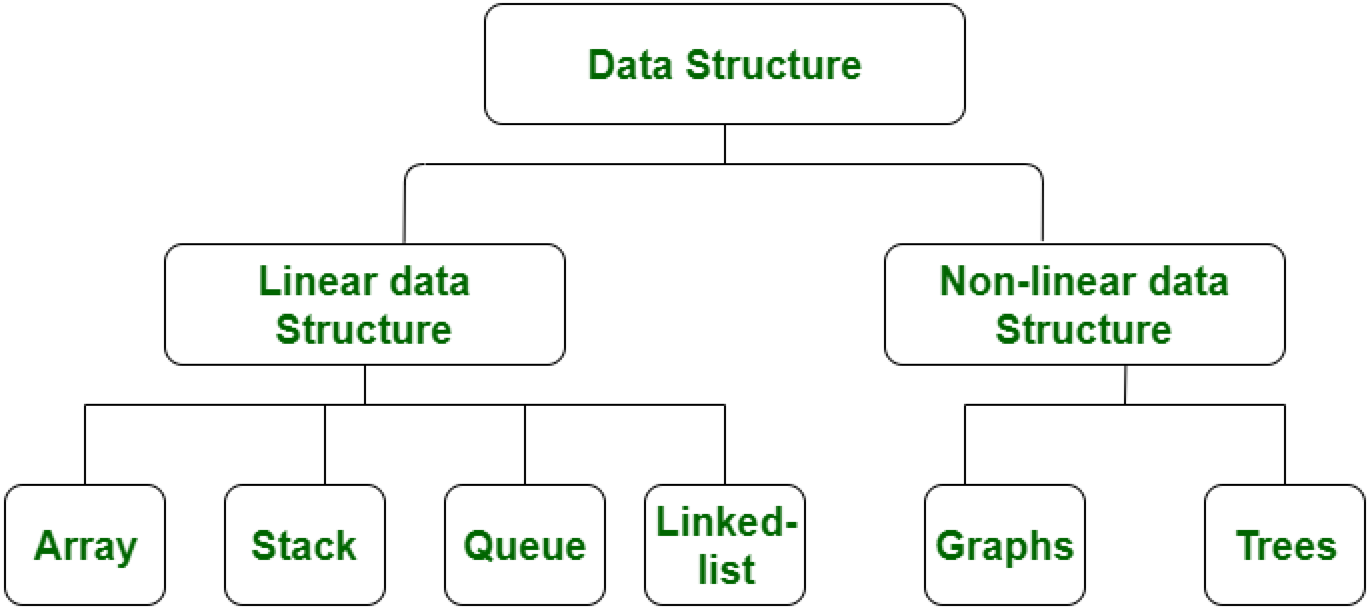
Classification of Data Structure



Classification of Data Structure



Classification of Data Structure



Data Structure

Linear data Structure

Non-linear data Structure

Array

Stack

Queue

Linked-list

Graphs

Trees

Classification of Data Structure

❑ Linear Data structure:

- Data structure in which data elements are arranged sequentially or linearly, where each element is attached to its previous and next adjacent elements, is called a linear data structure.

  Example: Array, Stack, Queue, Linked List, etc.

Classification of Data Structure

❑ Non-Linear Data Structure:

- Data structures where data elements are **NOT** placed sequentially or linearly are called non-linear data structures.
- This structure is mainly used to represent data containing a **hierarchical** relationship between elements. **Examples:** Trees and Graphs.

Types of Data

Characteristic	Description
Linear	In Linear data structures, the data items are arranged in a linear sequence. Example: Array
Non-Linear	In Non-Linear data structures, the data items are not in sequence. Example: Tree, Graph
Homogeneous	In homogeneous data structures, all the elements are of same type. Example: Array
Non-Homogeneous	In Non-Homogeneous data structure, the elements may or may not be of the same type. Example: Structures
Static	Static data structures are those whose sizes and structures associated memory locations are fixed, at compile time. Example: Array
Dynamic	Dynamic structures are those which expand or shrink depending upon the program need and its execution. Also, their associated memory locations changes. Example: Linked List created using pointers

Data Structure Operations

- The most commonly used operation on data structure are broadly categorized into following types:
  - Create
  - Selection
  - Updating
  - Searching
  - Sorting
  - Merging
  - Delete
  - Insert

Description of various Data Structures : Arrays

- An array is defined as a set of finite number of homogeneous elements or same data items.
- It means an array can contain one type of data only, either all integer, all float-point number or all character.

Arrays

- The elements of array will always be stored in the consecutive (continues) memory location.

- The number of elements that can be stored in an array, that is the size of array or its length is given by the following equation:

  (Upperbound-lowerbound)+1

Arrays

- Insertion of new element
- Deletion of required element
- Modification of an element
- Merging of arrays

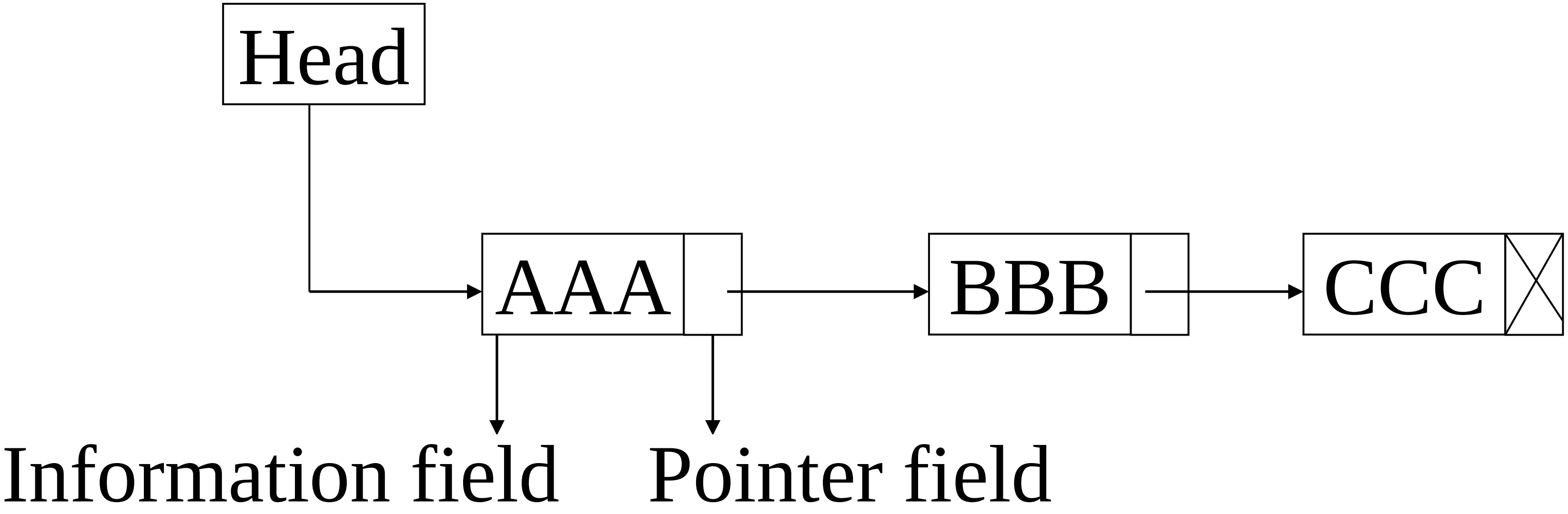
Lists

- A lists (Linear linked list) can be defined as a collection of variable number of data items.
- An element of list must contain at least two fields, one for storing data or information and other for storing address of next element.
- For storing address need a special data structure of list that is pointer type.

Lists

- Technically each such element is referred to as a node, therefore a list can be defined as a collection of nodes as show bellow:

[Linear Liked List]



Lists

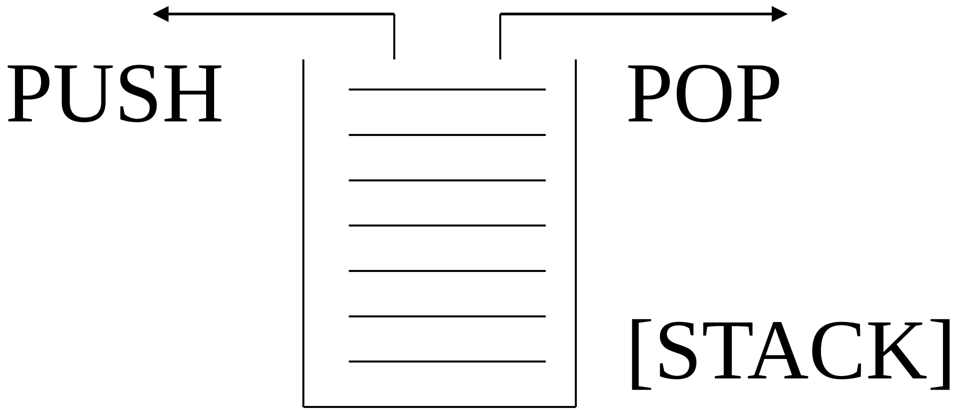
- Types of linked lists:
  - Single linked list
  - Doubly linked list
  - Single circular linked list
  - Doubly circular linked list

Stack

- A stack is also an ordered collection of elements like arrays, but it has a special feature that deletion and insertion of elements can be done only from one end called the top of the stack (TOP)
- Due to this property it is also called as last in first out type of data structure (LIFO).

Stack

- Insertion of element into stack is called PUSH and deletion of element from stack is called POP.
- The bellow show figure how the operations take place on a stack:



Stack

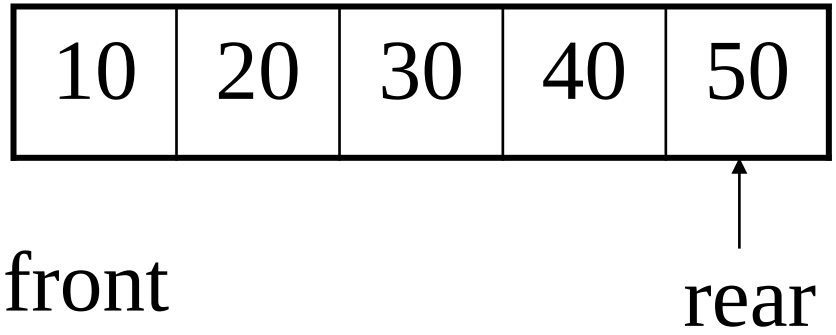
- The stack can be implemented into two ways:
  - Using arrays (Static implementation)
  - Using pointer (Dynamic implementation)

Queue

- Queue are first in first out type of data structure (i.e. FIFO)
- In a queue new elements are added to the queue from one end called REAR end and the element are always removed from other end called the FRONT end.
- The people standing in a railway reservation row are an example of queue.

Queue

- The bellow show figure how the operations take place on a queue:



10	20	30	40	50

front

rear

Queue

- The queue can be implemented into two ways:
  - Using arrays (Static implementation)
  - Using pointer (Dynamic implementation)

Trees

- Tree is non-linear type of data
- Tree represent the hierarchical relationship between various elements.

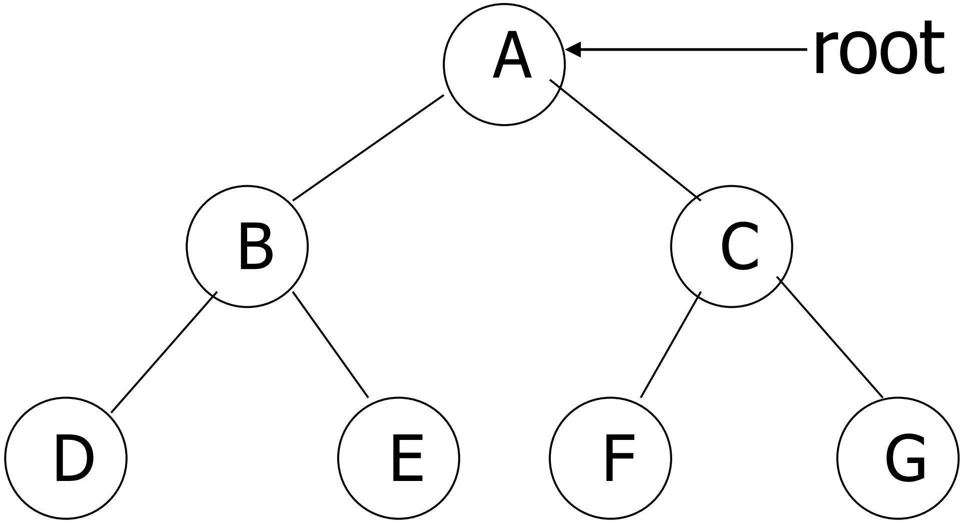
Trees

- There is a special data item at the top of hierarchy called the Root of the tree.

- The remaining data items are partitioned into number of mutually exclusive subset, each of which is itself, a tree which is called the sub tree.

Trees

- The tree structure organizes the data into branches, which related the information.



Graph

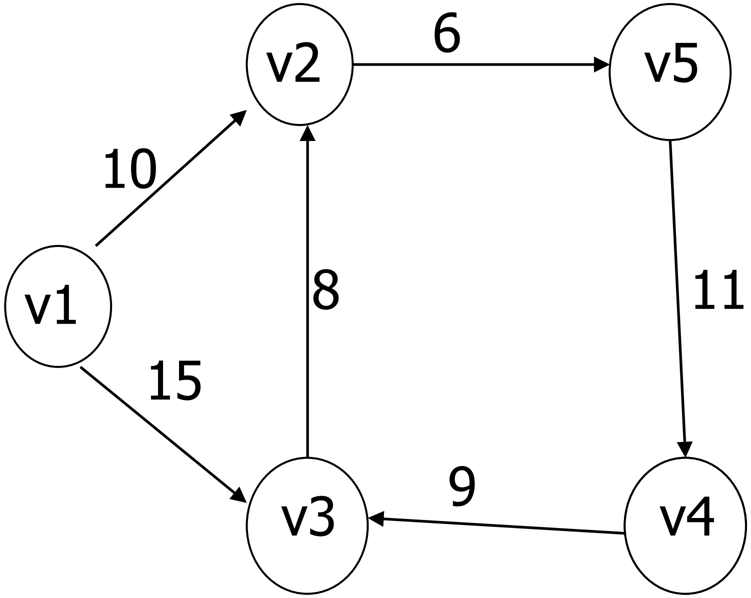
- Graph is a mathematical non-linear data structure capable of representing many kind of physical structures.
- Definition: A graph G(V,E) is a set of vertices V and a set of edges E.

Graph

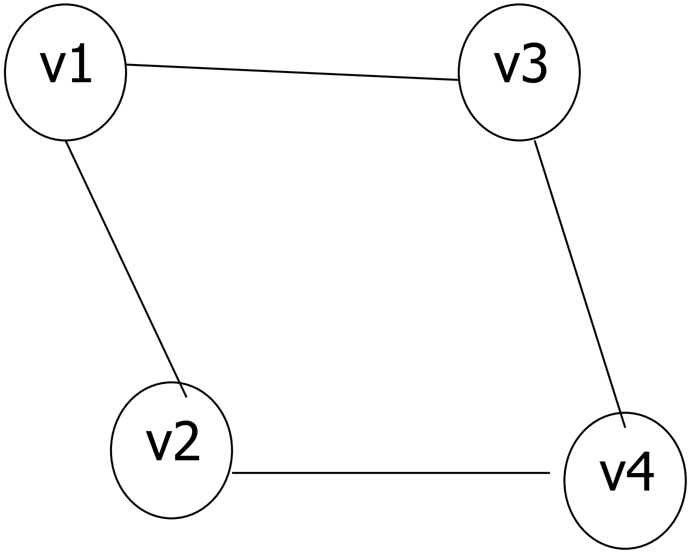
- An edge connects a pair of vertices and many have weight such as length, cost and another measuring instrument for according the graph.
- Vertices on the graph are shown as point or circles and edges are drawn as arcs or line segment.

Graph

- Example of graph:



[a] Directed & Weighted Graph



[b] Undirected Graph

Graph

- **Types of Graphs:**
  - Directed graph
  - Undirected graph
  - Simple graph
  - Weighted graph
  - Connected graph
  - Non-connected graph

OPERATIONS

- Data appearing in Data Structure are processed by means of certain operation

- Particular DS one chooses for a given situation depends largely on the frequency with which specific operations are performed

MAJOR OPERATIONs

- **Traversing**: Accessing each record exactly once so that certain items in the record may be processed [Also known as Visiting the record]
- **Searching**: Finding the location of the record with a given key value, or finding the locations of all record which satisfy one or more conditions
- **Inserting** : Adding a new record to the structure
- **Deleting** : Removing a record from the structure
- **Sorting**: Arranging a list in some logical order.
- **Merging**: Combing two list in a single list.

Abstract Data Type

- Abstract Data Types (ADT's) are a model used to understand the design of a data structure. Abstract mean an implementation-independent view of the data structure.

- ADTs specify the type of data stored and the operations that support the data

Abstract data type (ADT)

- Abstract data type (ADT) is a specification of a set of data and the set of operations that can be performed on the data. Each operation does a specific task.

Uses of ADT

- It helps to efficiently develop well designed program
- Facilitates the decomposition of the complex task of developing a software system into a number of simpler subtasks
- Helps to reduce the number of things the programmer has to keep in mind at any time
- Breaking down a complex task into a number of earlier subtasks also simplifies testing and debugging

List of ADT’s:

1. Insertion at first, middle, last
2. Deletion at first, middle, last
3. Searching
4. Reversing
5. Traversing
6. Modifying the list,
7. Merging the list