Uva Wellassa University

Data Structures and Analysis of Algorithms CST 225-3

About the Course

- Credits : 3
- Type of Credit : Compulsory
- Lecture Hours : 30 hours
- Practical Hours : 30 hours
- Evaluation Criteria :
  - Continuous Assessments 40%
  - End Semester Examinations 60%
- Schedule : Wednesday from 8.00 am to 10.00 am
  Thursday from 8.00 am to 10.00 pm
- Attendance: 80% attendance is compulsory

Objectives

- To provide the essential knowledge on different data structures and how to design and analyse the algorithms.

Learning Outcomes

At the end of the course, the students will be able to:

- explain the fundamental of data structures and algorithms its importance
- compare the performance of the algorithms and analyse those algorithms
- implement basic numerical algorithms
- understand simple data structures such as stack and queue, explain runtime and memory efficiency of them and implement related algorithms
- apply some algorithmic techniques to sort a given dataset

Learning Outcomes

At the end of the course, the students will be able to:

- describe implementation of various searching techniques
- explain different type of tree structures, various type of operations performs on tree and how the tree balancing affects to the efficiency
- solve problems using graph and greedy algorithms
- describe implementation of hash tables, with collision avoiding methods and resolution

Recommended References

- Mark Allen Weiss,2012,Data Structures and Algorithm Analysis in Java, 3rd Edition or Latest
- Michael T. Goodrich, Roberto Tamassia, David M. Mount, Data Structures and Algorithms in C++, 2nd Edition or Latest
- Cormen, Leiserson, Rivest, and Stein, Introduction to Algorithms, 3rd Edition or Latest
- Robert Lafore, Data Structures & Algorithms in Java, 2nd Edition or Latest
- Harsh Bhasin, 2015, Algorithms Design and Analysis, Oxford University Press
- Robert Sedgewick and Kevin Wayne ,2011, Algorithms, 4th Edition or Latest Addison Wesley, ISBN 0-321-57351-X

Introduction to Data Structures and Algorithms

Lecture 01

Content

- Introduction
- Types of Data Structures and its Importance
- Factors Affecting in Selecting an Algorithm
- Abstract Data Types (ADT)
- ADT Operations
- ADT Data Structures
- Introduction to Algorithms

Today’s Learning Outcome

At the end of this lecture you should be able to,

- Understand and define “data structure” and “algorithm”
- Understand the need of data structures
- Properties of data structures

What is a Computer Program?

- A set of instructions to a computer to perform some task or handle data.
- It is an implementation of an algorithm with a computer programming language.
- Eg: Program to add two numbers.
  - Input number 1 and number 2
  - Add two numbers
  - Print sum

What is Data?

- Programs are written to handle data.
- Refers to a single set of values or collection of values.
- Many forms such as text, number, image, etc.

What is a Data Structure?

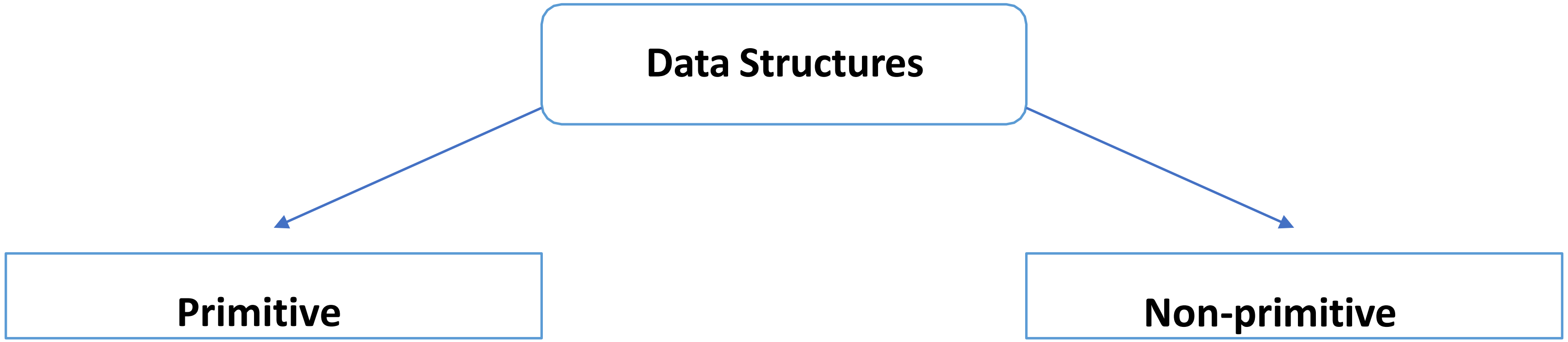
- Data Structure is a way to store and organize data efficiently.
- There are many ways of organizing the data in the memory.
- They provide both space efficiency and time efficiency in arranging the data.
- When learning Java; you have already used one of the data structures, Arrays.
- Array is a collection of data elements where data is stored sequentially (one after the other) in the memory.

Advantages of Data Structures

- Efficiency : The use of data structures make a program to work efficiently in term of time and space
- Reusability : A same data structure can be reused
- Abstraction : Internal logic of a data structure can be hidden from the end user

Types of Data Structures

- There are two types of data structures;
    1. Primitive data structure
    2. Non-primitive data structure



Data Structures

Primitive

Non-primitive

Primitive Data Structure

- Data structures which are supported at the machine level.
- Consist of primitive data types like int, char, float, double, and pointer.

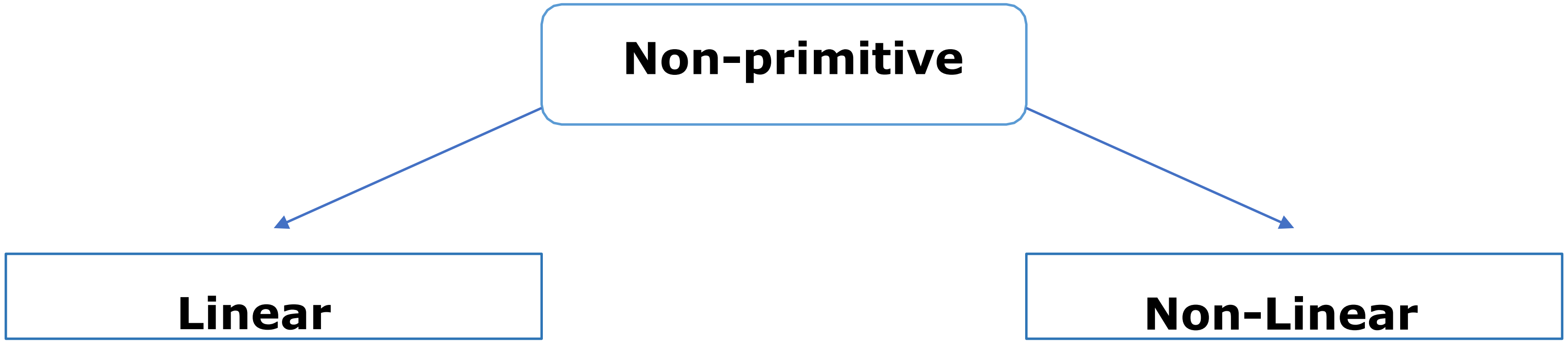
- They holds a single value.
  - int num = 4;
  - char ch =‘a’;
  - float f = 3.14;

Non-Primitive Data Structure

- They too provided by the languages, but cannot be formed using the primitive data structures.
- Used to store large and connected data.
- Eg:
  - Arrays
  - Lists
  - Queues

Non-Primitive Data Structure

- Non-primitive data structure can be divided into two types;
  1. Linear data structure
  2. Non-linear data structure



Non-primitive

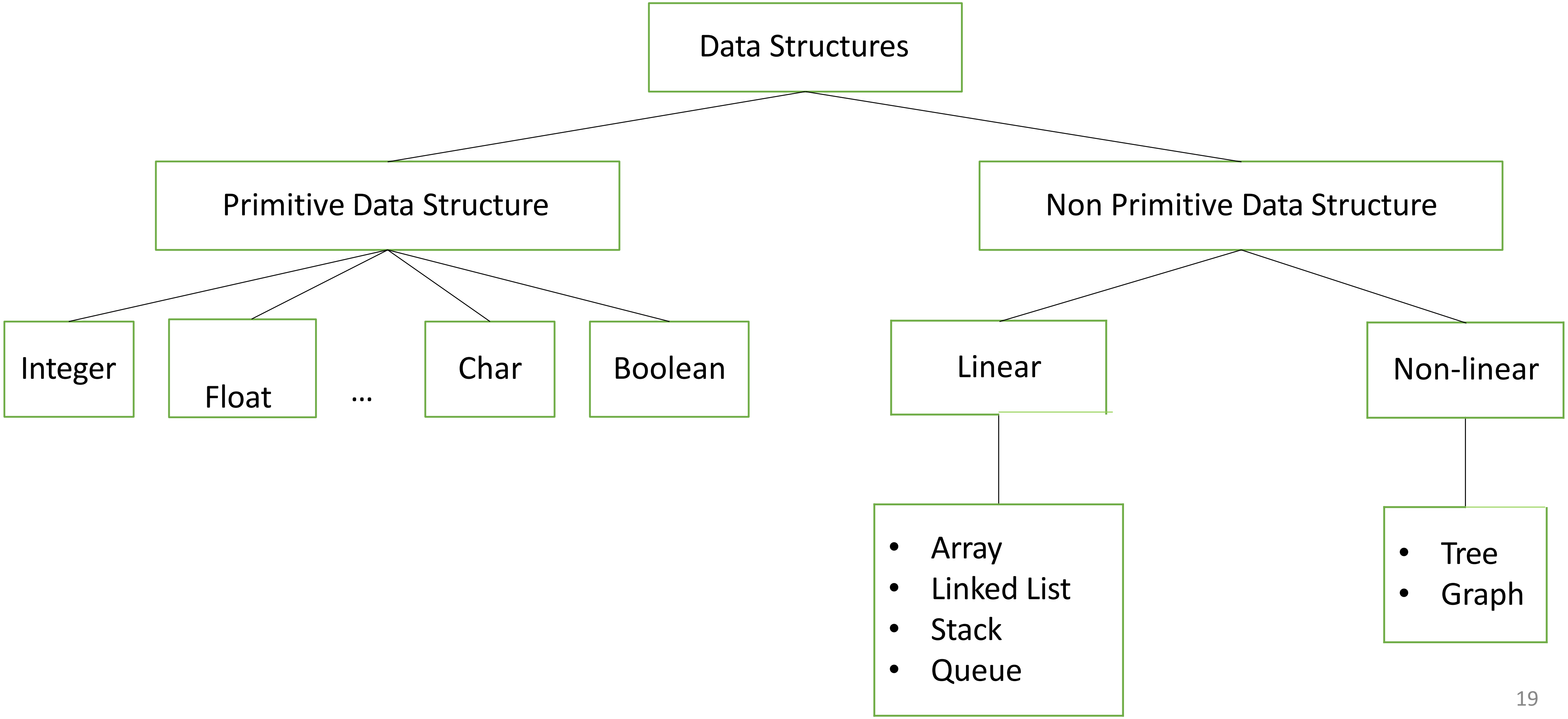
Linear

Non-Linear

Non-Primitive Data Structure

- In Linear data structure, the data is arranged in a sequential manner. For examples;
  - Arrays
  - Linked list
  - Stacks
  - Queues
- Here one element is connected to only one another element in a linear form.
- In non-linear data structure, one element is connected to 'n' number of elements. For examples;
  - Trees
  - Graphs

Data Structures Hierarchy

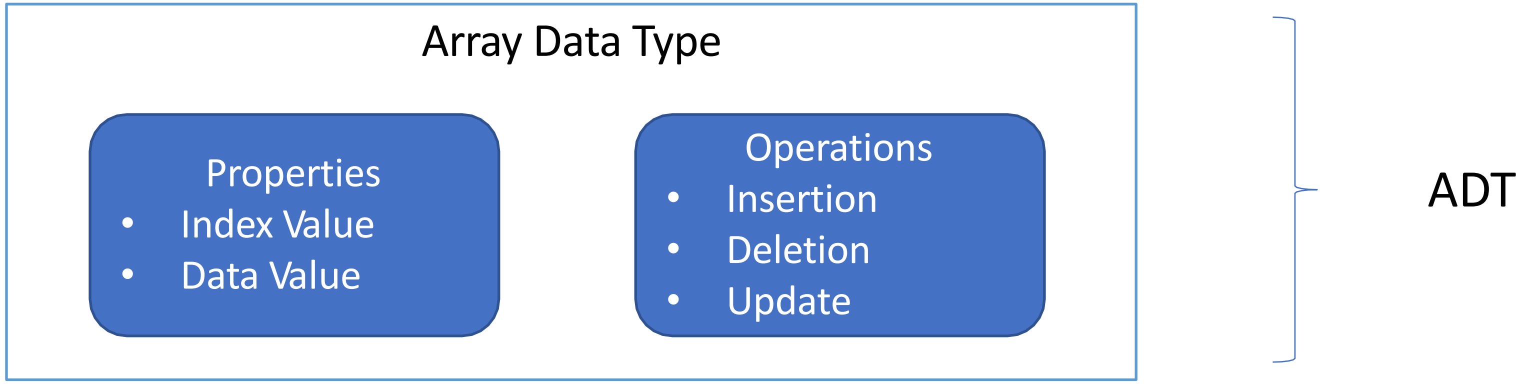


Abstract Data Types (ADT)

- An abstract data type (ADT) is a mathematical model for data structures.
- They define the behaviour of data structures by a set of values and a set of operations in an abstract way.
- They hide the implementation logic of a data structure while describing what operations are to be performed but not how these operations will be implemented.
- There can be different types of ADTs like;
    - List ADT
    - Stack ADT
    - Queue ADT

Array Data Structure

- An array is a collection of fixed number of components (data elements).
- In an array all the data elements have the same data type.
- There can be;
  - One-dimensional arrays : components are arranged in list form
  - Multi-dimensional arrays : components are arranged in tabular form



Array Data Type

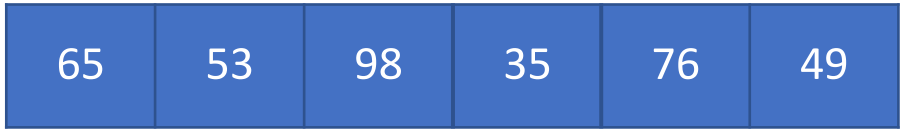
Properties
- Index Value
- Data Value

Operations
- Insertion
- Deletion
- Update

ADT

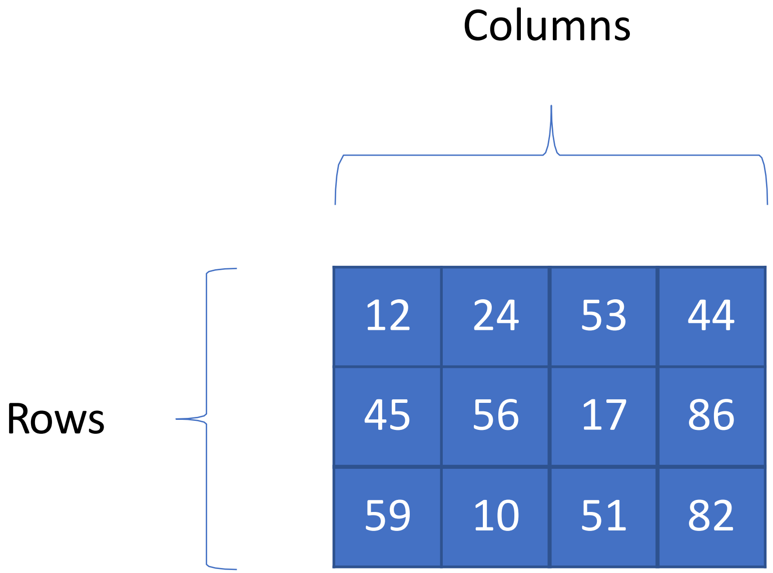
Array Data Structure

```
int marks[] = new int[6];
```



65	53	98	35	76	49

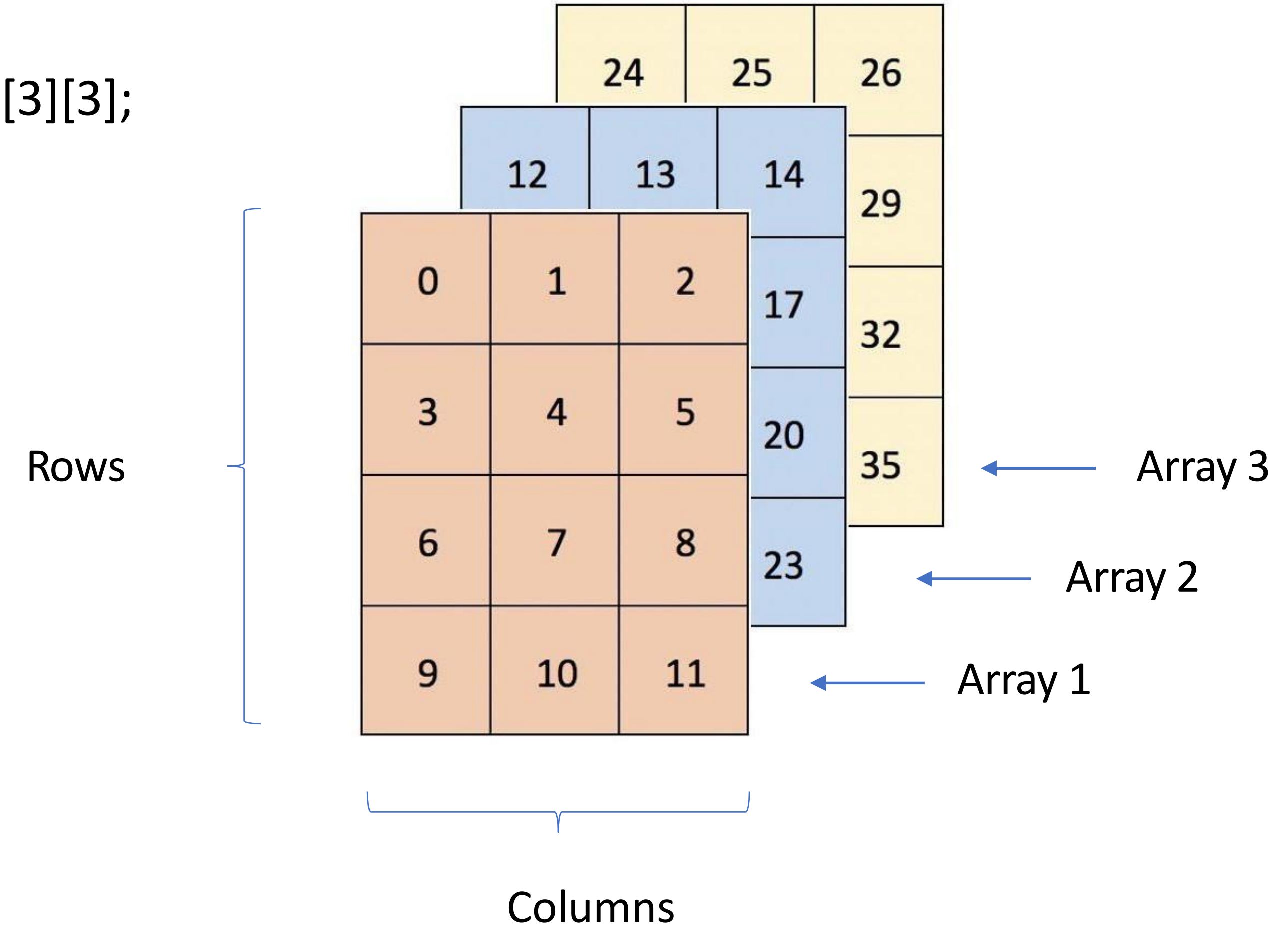
```
int lengths[][] = new int[3][4];
```



12	24	53	44
45	56	17	86
59	10	51	82

Array Data Structure

```
int data[][][] = new int[4][3][3];
```



Rows

Array 3

Array 2

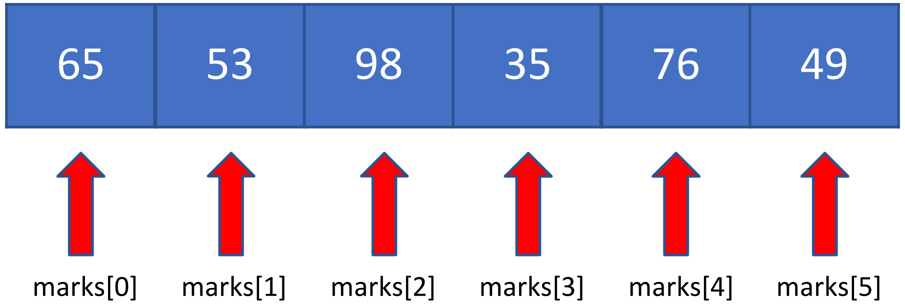
Array 1

Columns

Basics of Arrays

- An Array consists of a collection of consecutive memory locations that have the same type.
- The collection of data in an array is indexed and the index starts with 0.
- Index is called as the subscript as well.

```
int marks[] = new int[6];
```



65	53	98	35	76	49
marks[0]	marks[1]	marks[2]	marks[3]	marks[4]	marks[5]

What is an Algorithm?

- An algorithm is a finite set of instructions or logic, written in order, to accomplish a certain predefined task.
- Used to manipulate the data contained in these data structures such as searching and sorting.
- Can be expressed either as an informal high level description as **pseudocode** or using a **flowchart**.

Next..

- Stack

Questions?