Introduction to Data Structures and Algorithms

As part of Units 1 and 2

CC4 Data Structures and Algorithms

Christine Gonzales

Table of Contents

- Concepts on algorithms and data structures
- Data types and structures
- Data structures
- Array address calculations

Concepts on Algorithms and Data Structures

Basic terminologies | Parts of a programming language | Concepts in algorithms and data structures

Basic Terminologies

- Algorithm
- Data structure
- Programming language

Basic Terminologies – Algorithm

- Finite structure of well-defined instructions
- Used to solve a problem
- Typically independent of the programming language
- Can be expressed in:
  - Formal language (English, Filipino, etc.)
  - Flowcharting
  - Pseudocode
  - Programming language

Basic Terminologies – Data Structure

- Process of organizing data in a computer for more efficient use
- Looks into:
  - Collection of data values
  - Relationships amongst data values,
  - Functions and operations applied to the data
- Expressed as an algorithm
  - All data structures are algorithms, but not all algorithms are data structures

Basic Terminologies – Programming Language

- Set of commands used to create a software program
- Used to properly illustrate the concepts in an algorithm and data structure

*In this course, **Java** shall be used as the programming language to express the algorithms .and data structures.*

Parts of a Programming Language

- Data types and objects (int, float, boolean, String)
- Expressions (assignment, printing)
- Operations (arithmetic, conditional, logical)
- Decision control structures (if, else if, else, switch)
- Iterative control structures (while, for, do-while)
- Arrays (one-dimensional, multi-dimensional)
- Methods (user-defined, parameters, return type)

Parts of a Programming Language

- Other parts:
  - Input
  - Classes and objects

Concepts in Algorithms and Data Structures

- Data
- Data type
- Basic operations

Concepts in Algorithms and Data Structures

Data

A data must have the following characteristics:

- Atomic – Define a single concept
- Traceable – Be able to be mapped to some data element
- Accurate – Should be unambiguous
- Clear and Concise – Should be understandable

Concepts in Algorithms and Data Structures

Data Type

- Classifies various types of data which help:
  - Determine the values that can be used
  - Type of operations that can be performed

Concepts in Algorithms and Data Structures

Basic Operations

- Data in data structures are processed by operations
- Largely depends on the frequency of the operation that needs to be performed
- Examples:
  - Search
  - Insertion
  - Deletion
  - Sorting
  - Merging

Data Types and Structures

What is a Data Type? | What is a Data Structure? | Forms of Data Structures

What is a Data Type?

- Attribute of data that tells the compiler / interpreter how the data is intended to be used
- Looks into what kind of data can be placed inside of the variable
- Types of data types:
  - Built-in
  - Derived
  - Data object – represents an object having a data (i.e. String)

What is a Data Type?

Built-in Data Type

- Programming language has built-in support
- Examples:
  - Integers
  - Boolean (true, false)
  - Floating (Decimal numbers)
  - Character and Strings

What is a Data Type?

Derived Data Type

- Implementation independent
- Normally built by the combination of primary or built-in data types and associated operations on them
- Examples (based on the one-dimensional array):
  - List
  - Array
  - Stack
  - Queue

What is a Data Structure?

- Collection of data type values
- Process of organizing data in a computer for more efficient use
- Looks into:
  - Collection of data values
  - Relationships amongst data values,
  - Functions and operations applied to the data
  - Expressed as an algorithm

Forms of Data Structures

- Linear
- Tree
- Hash
- Graphs

Forms of Data Structures

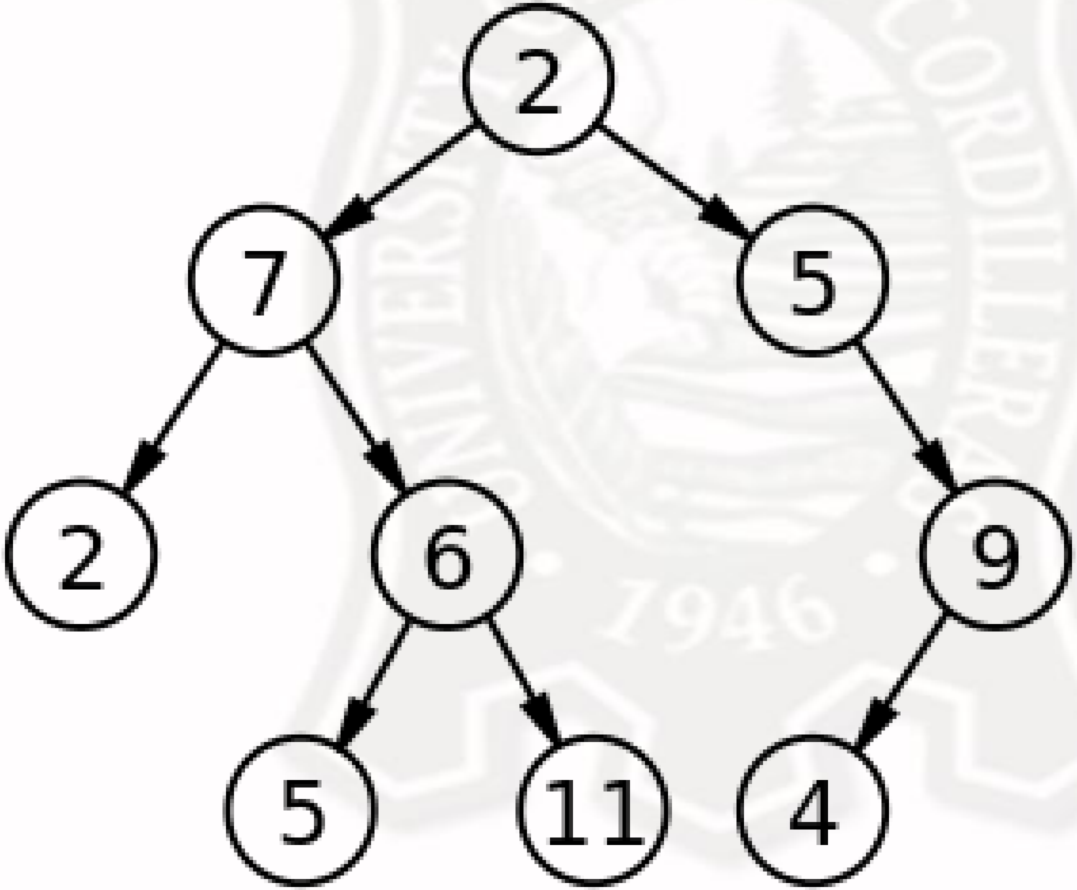
Linear

- Structure where the elements are stored sequentially
- Elements are connected to the previous and the next element
- As the elements are stored sequentially, so it can be traversed or accessed in a single run
- Examples:
  - Array
  - List

Forms of Data Structures

Tree

- Represent a hierarchical tree structure
- Contains a root value and subtrees of children (with a parent node)
- Represented as a set of linked nodes



Forms of Data Structures

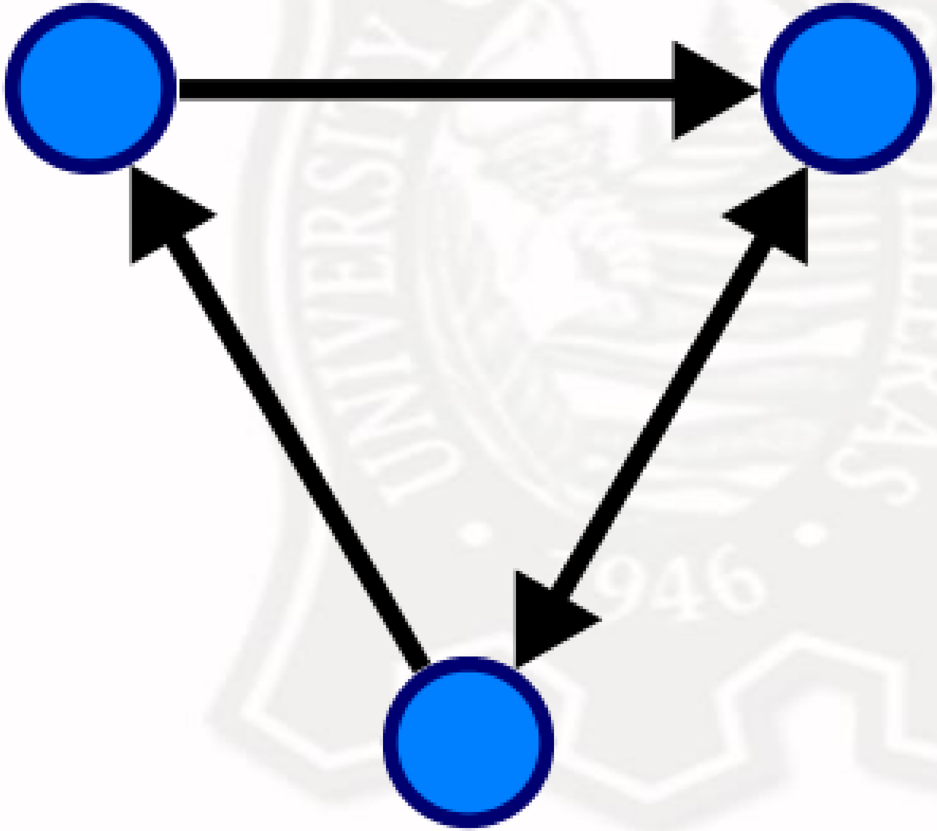
Hash Table

- Data structure capable of mapping keys to values
  - Key – labels the pair; used to pertain to the pair
  - Value – data stored as the pair to the key
- Typically abstracted and enhanced with additional behaviors
- Example:
  - Dictionary (Python)

Forms of Data Structures

Graph

- Abstract data type that follows the principles of graph theory
- Structure is non-linear
- Consists of:
  - Nodes / Vertices – points on the graph
  - Edges – lines connecting each node

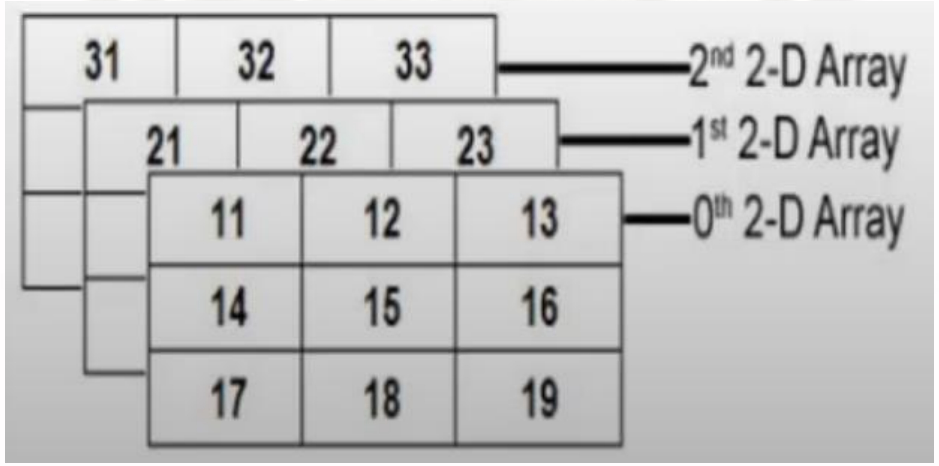


Array Address Calculations

Address calculations | Row major system | Column major system

Review: Multidimensional Array

- **One-Dimensional Array**
  - Also known as a list or just array
  - A set of elements stored sequentially
  - Uses the index as a pointer
- **Two-Dimensional Array**
  - Also known as a matrix
  - Stored sequentially in two dimensions
  - Data is stored “row and column wise”
- **Three-Dimensional Array**
  - Think as a collection of 2D arrays



Address Calculation

Two Ways:

- Row major system
  - All elements of the row are stored consecutively
- Column major system
  - All elements of the column are stored consecutively

	0	1	2
0	5	6	2
1	7	0	-3
2	4	1	3

Row	5	6	2	7	0	-3	4	1	3
Col	5	7	4	6	0	1	2	-3	3

Row Major System

- All elements of the same rows are stored consecutively
- Formula:

Address of A[i][j] = baseAddress + w * (i * c + j)

- which means:
  - baseAddress = assigned address to A[0][0]
  - w = storage size of one element stored in the array
  - i = row index
  - c = number of columns
  - j = column index

Row Major System – Example

Address of A[i][j] = baseAddress + w * (i * c + j)

	0	1	2	3	4
0	50	55	60	65	70
1	75	80	85	90	95
2	100	105	110	★ 115	120
3	125	130	135	140	145

B = 50
W = 5
i = 2
j = 3

50 + 5 (2 * 5 + 3)
50 + 5 (10 + 3)
50 + 5 (13)
50 + 65
Address of [2][3] = 115

Column Major System

- All elements of the same columns are stored consecutively
- Formula:

$$\textit{Address of } A[i][j] = \textit{baseAddress} + w * (i + r * j)$$

- where:
  - baseAddress = assigned address of A[0][0]
  - w = storage size of one element stored in the array
  - i = row index
  - r = number of rows
  - j = column index

Column Major System – Example

$$\textit{Address of } A[i][j] = baseAddress + w * (i + r * j)$$

	0	1	2	3	4
0	50	70	90	110	130
1	55	75	95	115	135
2	60	80	100	★ 120	140
3	65	85	105	125	145

B = 50
W = 5
i = 2
j = 3

50 + 5 (2 + 4 * 3)
50 + 5 (2 + 12)
50 + 5 (14)
50 + 70
Address of [2][3] = 120