# 1 : Introduction to Data Structures

IT3206 – Data Structures and Algorithms

Level II - Semester 3

# Overview

- This course module section provides fundamental knowledge in the application of different data structures and Algorithms.

# Intended Learning Outcomes

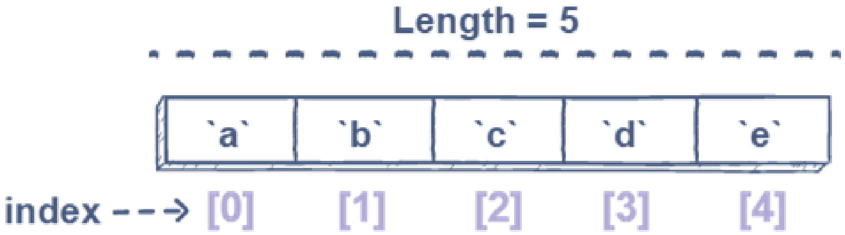
- At the end of this lesson, you will be able to;
  - Explain the use of data structures
  - Demonstrate different implementation techniques of data structures
  - Compare different applications of data structures

# List of subtopics

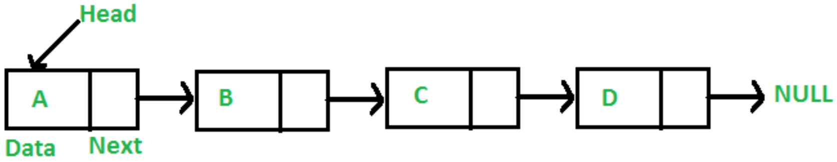
1. Introduction to data structures
    1. Introduction to data structures [Ref 3:pg.9, Ref 1:pg.230 ]
    2. Use of Data Structures [Ref 3:pg.10-11]
        1. Real- world data storage [Ref 3:pg.10-11]
        2. Programmer’s Tools [Ref 3:pg.11]
        3. Real-world Modeling  [Ref 3:pg.11]
    3. Overview of Data Structures [Ref 3:pg.11-12]
    4. Classification of Data Structures [Ref 4:pg.18]

# 1.1 Introduction to data structures

- A data structure is an arrangement of data in a computer's memory or disk.
- Algorithms manipulate the data in the data structure such as searching for a particular data item, sorting the data etc.
- Data structures include arrays, linked lists, queues, stacks, binary trees, and hash tables etc.
- Examples

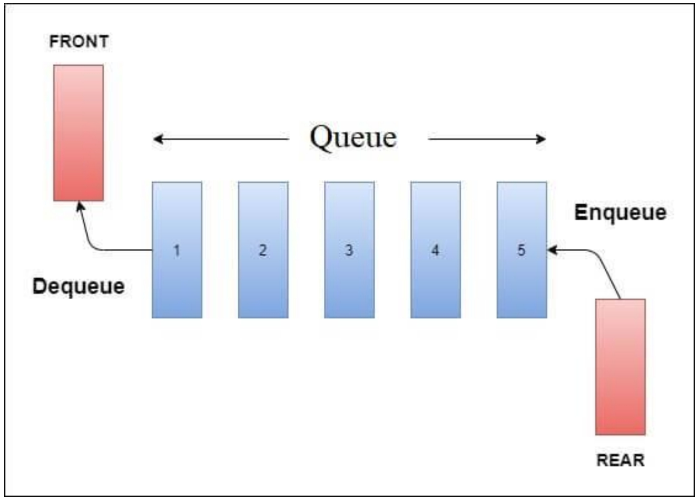


Array

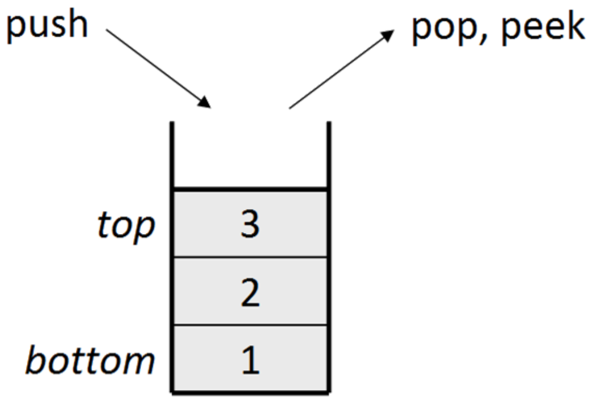


Linked list

# 1.1 Introduction to data structures cont.



Queue



Stack

# 1.2 Use of Data Structures

The use of data structures can be studied mainly in these three categories:

- Real-world data storage
- Programmer’s tools
- Real-world modeling

# 1.2.1 Real-world data storage

- Many of the structures and techniques considered here are about how to handle real-world data storage.
- Real-world data mean that the data that describes physical entities external to the computer.

**Examples**:

- A personnel record describes an actual human being.
- An inventory record describes an existing car part or grocery items.
- Financial transaction record describes an actual check written to pay the electric bill.

# 1.2.1 Real-world data storage cont.

- A stack of index cards can be considered as a non-computer related example of real-world data storage.
  - These cards can be used for a variety of purposes.
  - If each card holds a person's name, address, and phone number, the result is an address book.
  - If each card holds the name location, and value of a household possession, the result is a home inventory.

# 1.2.1 Real-world data storage cont.

- Most programs are more complex than index cards.
- Imagine the database the Department of Motor Vehicles uses to keep track of drivers’ licenses, or an airline reservations system that stores passenger and flight information.
- Such systems may include many data structures.

# 1.2.2 Programmer’s Tools

- Not all data storage structures are used to store real-world data.
- Typically, real-world data is accessed more or less directly by the users of some programs.
- Some data storage structures, however, are not meant to be accessed by the user, but by the program itself.
- A programmer uses such structures as tools to facilitate some other operation.

Examples: Stacks, queues, and priority queues are often used in this way

# 1.2.3 Real-world Modeling

- Some data structures directly model real-world situations.

Examples

- evaluation of mathematical expressions
- Queue of print jobs to the printer
- Queue of programs/process to be run
- Queue of network data packets to be send
- Graphs to represent airline routes between cities or connections in an electric circuit or tasks in a project.
- Queues to model customers waiting in line at a bank or cars waiting at a toll booth.

# 1.3 Overview of Data Structures

- Another way to look at data structures is to focus on their strengths and weaknesses.
- Following is a bird's-eye view of a landscape that we'll be covering later at ground level, so don't be alarmed if the terms used are not familiar.

| **Data Structure** | **Advantages** | **Disadvantages** |
|---|---|---|
| Array | Quick insertion, Fast access if index known | Slow search<br>Slow deletes<br>Fixed size |
| Ordered Array | Quicker search than unsorted array | Slow inserts<br>Slow deletes<br>Fixed size |
| Stack | Last-in, first-out acces | Slow access to other items |
| Queue | First-in, first-out access | Slow access to other items |
| Linked List | Quick insertion, quick deletion | Slow search |

# 1.3 Overview of Data Structures cont.

| Data Structure | Advantages | Disadvantages |
|---|---|---|
| Binary Tree | Quick search<br>Quick insertion<br>Quick deletion<br>(If the tree remains balanced) | Deletion algorithm is complex |
| Red-Black Tree | Quick search<br>Quick insertion<br>Quick deletion<br>(Tree always remains balanced) | Implementation is complex |
| 2-3-4 Tree | Quick search<br>Quick insertion<br>Quick deletion<br>(Tree always remains balanced)<br>(Similar trees good for disk storage) | Implementation is complex |

# 1.3 Overview of Data Structures cont.

| **Data Structure** | **Advantages** | **Disadvantages** |
|---|---|---|
| Hash Table | Very fast access if key is known<br>Quick inserts | Slow deletion<br>Access slow if key is not known<br>Inefficient memory usage |
| Heap | Quick insertion<br>Quick deletion<br>Quick access to largest item | Slow access to other items |
| Graph | Models real-world situations | Some algorithms are slow and complex |

# 1.4 Classification of Data Structures

- A data structure is a special format for organizing and storing data.
- Depending on the organization of the elements, data structures are classified into two types:

1. Linear data structures: Elements are accessed in a sequential order but it is not compulsory to store all elements sequentially.

   Examples:
   Arrays, Linked Lists, Stacks and Queues.

2. Non – linear data structures: Elements of this data structure are stored/accessed in a non-linear order.

   Examples: Trees and graphs.

# Summary

| | |
|---|---|
| 1.2.1 | Real-world data mean that the data that describes physical entities external to the computer. |
| 1.2.2 | A programmer uses data structures as tools to facilitate the required operations. |
| 1.2.3 | Some data structures directly model real-world situations. |