Data Structures and Algorithms

Lecture 01

Objectives

By the completion of this module, the student will be able to:

- Able to explain different types of data structures.
- Able to manipulate data structures for searching, sorting and other familiar function which occur frequently in computer programs.

- 2.5 Credits
- Lectures (2 hours)
- Lab (3 hours every other week)

Course Outline

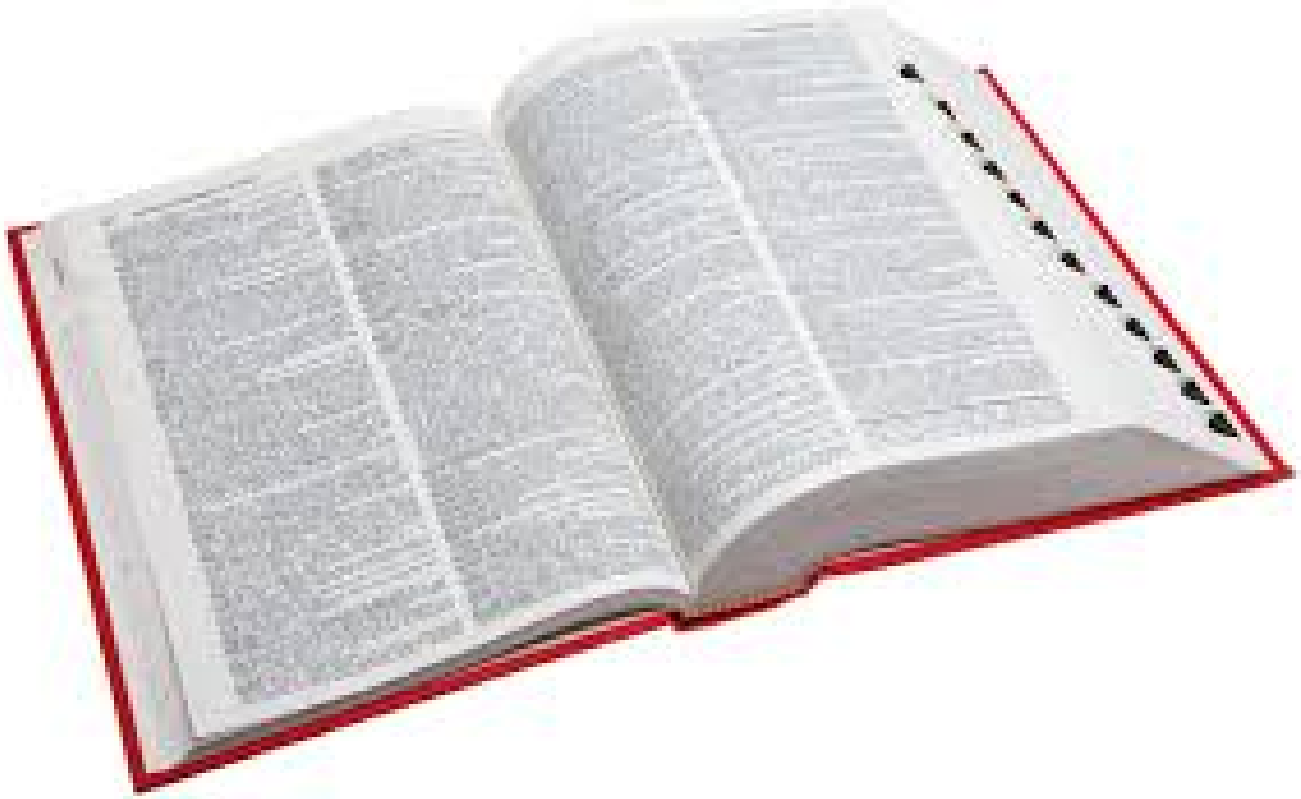
- Introduction to data structures and algorithms
- Stack, Queues and Linked data structures
- Trees and Graphs
- Sorting
- Searching
- Recursion
- Algorithm analysis techniques.

Recommended Text

- **Data Structures and Algorithms in Java, Adam Drozdek, Third Edition**
- **Data Structures & Algorithms in Java, Micheal Goodrich and Roberto Tamassia, Second Edition**

- Preliminaries:
  - Good programming skill
    - Any programming language
  - Good analytical skill with sufficient mathematical knowledge

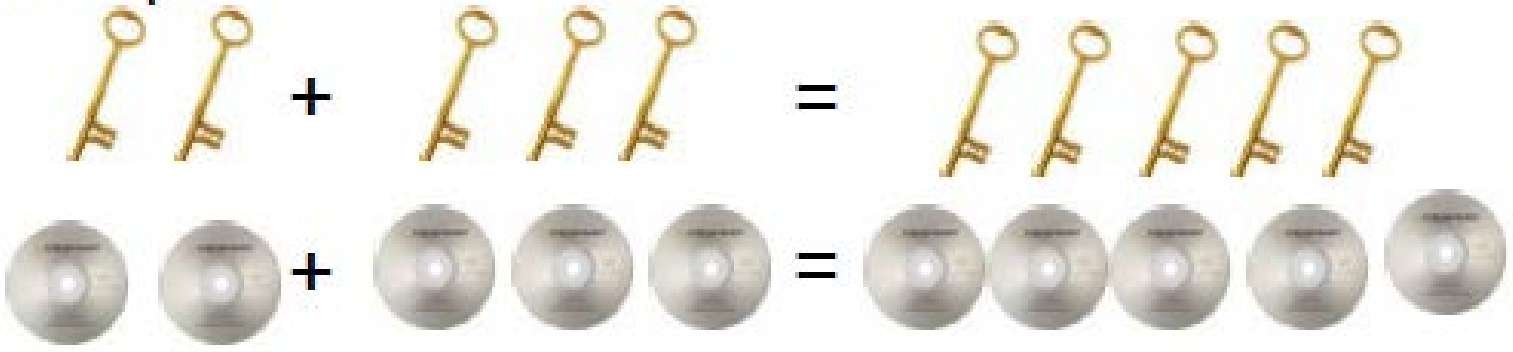
- Think about the process of finding a word in dictionary..



Abstraction

- Consider the general/logical idea that can be applied as different concrete instances

Example:



2 + 3 = 5

Abstract Data Types (ADT)

- Apply the concept of abstraction to the design of data structures

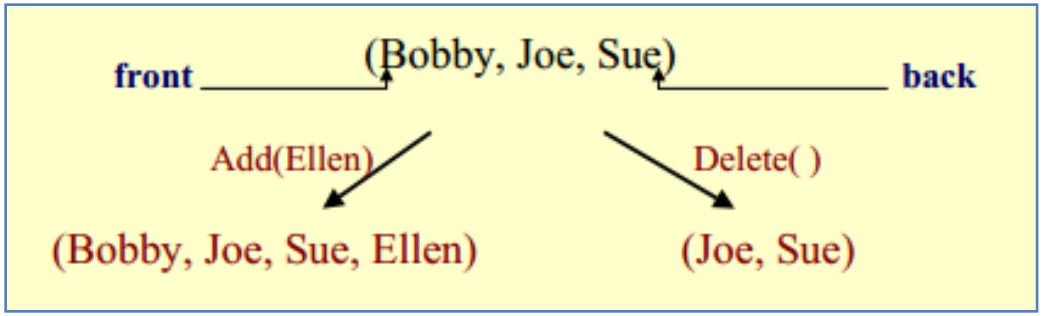
- A mathematical model of a data structure:
  - Collection of data items
  - Operations supported on them
  - Types of parameters of the operations

- Sound familiar… Classes/Interfaces in OOP

ADT explain

- Type defined in terms of its data items and associated operations
  - Not its implementation.
- List, Queue, Map, Set – ADTs
- Difference between list and set
  - List allows duplicity

- List (ADT) can be implemented by ArrayList and LinkedList (DS)



- Queue (ADT) – FIFO
  - Arbitrary number of items
  - Operations:
    - Add elements **only** to the end of the list
    - Remove elements **only** from the front of the list

    - Find the number of elements in the queue
    - Check whether the queue is empty or not

Data Structure

- Implementation of an ADT using specific structures.
  - Consider actual storage of data items in computer memory
  - Algorithms to perform specified operations
- Way to store and retrieve data
- Operations can be performed on these structures.
  - Insertion, Deletion, Searching, Sorting

Selection of a data structure

- What are the required operations?

- What is the efficiency of the each operation?
  - Sorting
  - Searching
  - Inserting
  - Traversing
  - Removing

Example: Bank Application

- Assume, basic operations are:
  - Open accounts (far less often than access)
  - Close accounts (far less often than access)
  - Access account to Add money
  - Access account to Withdraw money

- Teller and ATM transactions are expected to take little time.
- Opening or closing an account can take much longer (perhaps up to an hour).

- So, we are looking for a data structure (most appropriate one) which is
  - Highly efficient for search
  - Moderately efficient for insertion
  - Efficient or inefficient for deletion (Do not care)

- Answer: It is better to use (???)
  - Hash tables
  - Array List
  - Linked List

Why do we need data structures?

- To organize data to create more efficient computer programs.

- Handling more complex applications which are demanded for more calculations.

- The Choice of the data structure and algorithms can make the difference between a program running in a
  - Few Seconds OR
  - Many days

Efficiency

- A solution is said to be **efficient** if it solves the problem within its **resource constraints.**
  - **Space**
  - **Time**

- The **cost of a solution** is **the amount of resources** that the solution consumes.

Data Structure Philosophy

- Each data structure has costs and benefits.
- A data structure requires:
  - Space for storing each data item
  - Time to perform each operation
  - Programming effort
- Each problem has constraints on available space and time.

Data Structures Types

- Linear –
  - Elements form a sequence : array list, linked list

- Non-linear-
  - Data items are not arranged in a sequential structure.
    - Trees, graphs

Problems

- A task to be performed

- Mathematically,
  - A function which is matching between inputs (domain) and outputs (range)

Algorithms

- Step- by- step recipe for performing a task within a finite period of time.

- Algorithms often operate on a collection of data, which is stored in a structured way in computer memory (data structure)

- A problem can be solved by many algorithms.
  - Sorting problem : Insertion, Bubble, Selection, ShellSort, MergeSort

Characteristics of a good algorithm

- Finiteness:
  - Terminates after a finite number of steps and each step should be executable in finite amount of time
- No ambiguity:
  - Each steps of an algorithm must be precisely defined
- Input:
  - Algorithm should have a finite number of inputs

- Output:
  - An algorithm has one or more outputs (at least one output)
  - Can be proved to produce the correct output for a given input

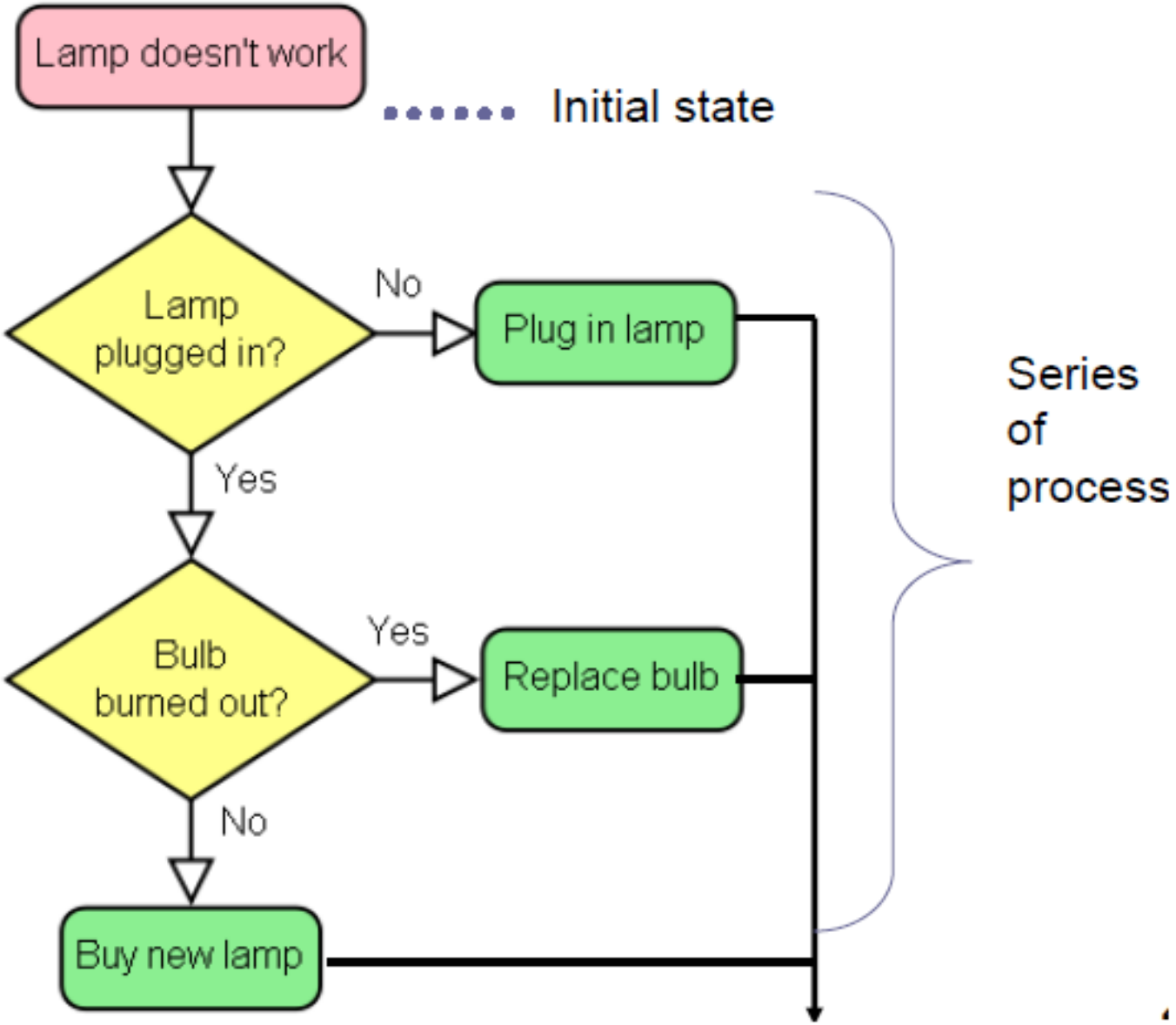
- Effectiveness:
  - Steps should be simple and basic

Algorithm creation techniques

- Flow Chart
- Pseudo code
- Programming language

Measurements

- Numerical factors for measuring the goodness and effectiveness of an algorithm
  - Running Time
  - Total memory usage


Lamp doesn't work
Initial state
Lamp plugged in?
No
Plug in lamp
Yes
Bulb burned out?
Yes
Replace bulb
No
Buy new lamp
Series of process

- Questions ???