HANDS-ON TRAINING

DATA STRUCTURES

&

Object Oriented Programming

By Ali Imran

Data Structures (*topics*)

- Introduction
- Linked List
  - Doubly Linked List
- Stack
- Queue
  - Priority Queue
- Binary Tree
- Hash Table

Object Oriented Programming (*topics*)

Introduction
Object and Class
Inheritance
Polymorphism
Encapsulation
Interfaces
Abstraction

Next > DETAILS

Data Structures (*details*)

Introduction

In computer science, a data structure is a particular way of organizing data in a computer so that it can be used efficiently.

Linked List

In computer science, a linked list is a linear collection of data elements, called nodes, each pointing to the next node by means of a pointer. It is a data structure consisting of a group of nodes which together represent a sequence. Linked lists are among the simplest and most common data structures. Covered are: concepts, add, update, delete, traverse.

Stack

In programming, a special type of data structure in which items are removed in the reverse order from that in which they are added, so the most recently added item is the first one removed. This is also called last-in, first-out (LIFO). Adding an item to a stack is called pushing.

Queue

Queue is an abstract data structure, somewhat similar to Stack. In contrast to Queue, queue is opened at both end. One end is always used to insert data (enqueue) and the other is used to remove data (dequeue).

Priority Queue

In computer science, a priority queue is an abstract data type which is like a regular queue or stack data structure, but where additionally each element has a "priority" associated with it. In a priority queue, an element with high priority is served before an element with low priority.

Binary Tree

In computer science, a **binary tree** is a **tree** data **structure** in which each node has at most two children, which are referred to as the left child and the right child

Hash Table

In computing, a **hash table** (**hash** map) is a data **structure** used to implement an associative array, a **structure** that can map keys to values. A **hash table** uses a **hash** function to compute an index into an array of buckets or slots, from which the desired value can be found.

Next > OOP

OOP (*details*)

Introduction

Object Oriented Programming languages are defined by the following key words: abstraction, encapsulation, inheritance, and polymorphism. An object is a container of data and functions that affect the data. In an OOP, a "child" object can "extend" another object (making it more specific) by inheriting from a "parent" object. Thus the child gets all the parents data and functionality "for free".

Object and Class

Objects are at heart very simple. They are a way to represent information about a "real world" idea and they contain ways to manipulate that information. All of this "code" is encapsulated in an object recipe (Class file).

- They contain information about something in the program.
- They provide actions (also called functions or methods) which manipulate that information. By combining all the data and actions that can apply to an "object" into one piece of code, we make our programs easier to write and maintain

Inheritance

Inheritance is the idea that a "child class" gets all the functionality of its parent class. This works because child classes are more specific examples of the parent class

Polymorphism

Polymorphism is best summed up by the following:

- At compile time, a variable of the most general type is used to refer to an object that at run time may be any descendent type!
- At run time, "The object referenced by the variable behaves as it truly is, not as its variable is defined at compile time.

Encapsulation

Encapsulation, is the idea that the "data" associated with an object should (mostly) only be available via functions, and (possibly) that much of the data associated with an object will never be "visible" to the outside world.

For example, most of use do not care how our car engines work. What we do care about is that when we turn the key, the engine starts, when we press the gas, the car accelerates.

Interfaces

The idea of an "interface" is also key to OOP. An interface is the allowed interactions between the outside world (*e.g. other programs*) and the object itself.

Abstraction

In object-oriented programming, abstraction is one of three central principles (*along with encapsulation and inheritance*). Through the process of abstraction, a programmer hides all but the relevant data about an object in order to reduce complexity and increase efficiency.