

Fall 2006.

CS 3345: Data Structures and Algs. class #5

Data structures: premised upon the ADT
Abstract Data Type

ADT: is the description of the data type at an abstract level.

For any ADT, we are typically interested in the operations allowed on it.

For almost all data types/structures we are interested in

- (1) Insertion
- (2) Deletion
- (3) Search

at the most basic level.

That said known ADT's offer certain adv. optimized for particular operations.

For eg.

- (1) In a student DB at a University,

for each student record, record insertion/deletion happen infrequently but updates need to happen more frequently.

- (2) In a retail store w/an employee DB,

insertion/deletion may be the most frequent operations.

It is not just at the level of operations, that requirements differ widely. They also may differ vastly in terms of the particular types of operations:

Function executions require LIFO structures.
(Stacks!)

however,

Operating systems & DES's require FIFO
(Queues!).

Depending upon how severe the requirements and how different from other structure available we design ADT's to cater to these.

Some known standard ADT's are:

Lists, Trees and Graphs.

Each is characterized by allowed set of operations and the efficiency with which these operations are performed by the ADT.

We start with the simplest known ADT:

(1) Lists: collections of similar entities ordered in a linear (sequential) fashion.

↓
only in the sense of the linear order.

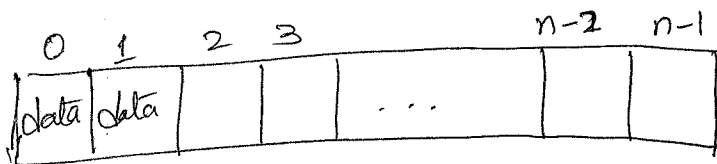
Allowed Operations:

- (a) Find or Search for an element
- (b) Insert an element
- (c) Delete an element.

can be "implemented" via two diff. mechms:

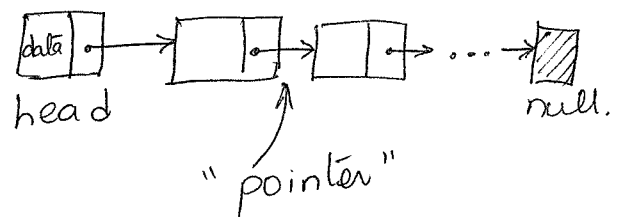
Arrays

Sequential, contiguous ordering of the list elements in memory.



Linked lists

Random ordering in memory, but each element does point to the "next".



The elements of "data" in the nodes of the structures above are "similar" or share common properties:

- (i) GPA of students
- (ii) scores of games
- (iii) airports of the country
- (iv) states in a country

etc.

