Data Structures (CS- 213)

Mohammad Asad Abbasi

Lecture 1: Introduction

Course Structure

- Lectures / Lab/ Class participation
- Assignments
- Quizzes
- Midterm examination
- Final examination

Grading

Assignments	5%
Quizzes	5%
Lab	20%
Midterm Exam	20%
Final Exam	50%
Total	100

Readings

- Readings from the required text will be assigned for each lecture, read them in advance.
- Course book :**Data Structures by Seymour Lipschutz.**
  (International Edition- Schaum's Outline Series.)
- The book contains self-test exercises in each section
- Work these exercises
- Use material from other books and research papers, the ultimate source should be lectures.

WEEKLY LECTURE PLAN	
S.No	**Topics**
Week1	**Data Structures and their Operations:** Introduction, Overview of Data Structures, Data Structure's operations, Types of Data Structures, Abstract data types
Week 2	**Algorithm Analysis:** Introduction, Pseudo coding, Control Structures, Analysis of Algorithms, Efficiency of Algorithms, Complexity of Algorithms, Asymptotic Notation/ Big O Notations
Week 3	**Pointers:** Introduction, Pointer vs. Arrays, Arrays of Pointer, Pointer to Pointer, Null pointer, Void Pointer, Invalid Pointer, Dangling Pointer Reference Variable, Dynamic Array (malloc, new, free, delete operators). Constructor in class, types of Constructor (Null, Default, Parametric, Overloading, Copy Constructor)
Week 4 & 5	**Sorting and Searching Techniques:** Linear Search, Binary Search, Efficiency of Algorithms, Bubble Sort, Selection Sort, Insertion Sort, Quick Sort, Shell Sort, Radix Sort, Merge Sort, Heap Sort
Week 6	**Linked Lists:** Introduction, Representation of Linked lists in memory, Traversing a linked list, Searching a linked list, Memory allocation, Insertion and Deletion, Efficiency of Linked lists, Header Linked lists, Two way lists, Two way Circular Header linked list, Operations on Two way CHL

Week 7	**Stacks:** Introduction, Stack Model, Array & Link List representation of Stacks, Representation of Stack through Class ,Recursion, Parenthesis Checking through stack, Efficiency of Stack. Polish Notations, Prefix, Infix, Postfix
Week 8	**Queues:** Introduction, Queue Model, Array & Link List implementation of Queues, Deques, Priority Queues, Linked list & Array implementation of Priority Queues
Week 9	**Mid Term Exam**
Week 10 & 11	**Trees:** Basic Terminologies, Binary Trees, Array/Link List representation of binary trees in memory, Traversing Binary Trees, Pre, In, and Post-order traversal, Binary Search Trees, Searching, Inserting, and Deleting in Binary Search Trees, Efficiency of BST, Binary Heaps, Heap creation and operations, Heap sort, AVL Tree, Creating AVL Trees, Balancing AVL Tree by single or double rotations, Red-Black Trees, Decision Trees,CPU burst (Exponential averaging)
Week 12	**Minimum Spanning Trees**: Applications of Shortest path algorithms (Prim's Algorithm, Dijkstra's Algorithm)

Week 13 & 14	**Graphs:** Basic Terminologies, Graph Theory, Directed, Undirected Graphs, Sequential representation of graphs, Adjacency Matrix, Path Matrix, Graph algorithms, Link List representation of graphs, Operations on Graphs, Graph Traversals (Depth-First Traversal, Breadth First Traversal)
Week 15	**Data Compression:** Encoding /Decoding, Huffman's Algorithm, Revision
Week 16	**Final Exam**

Smart devices

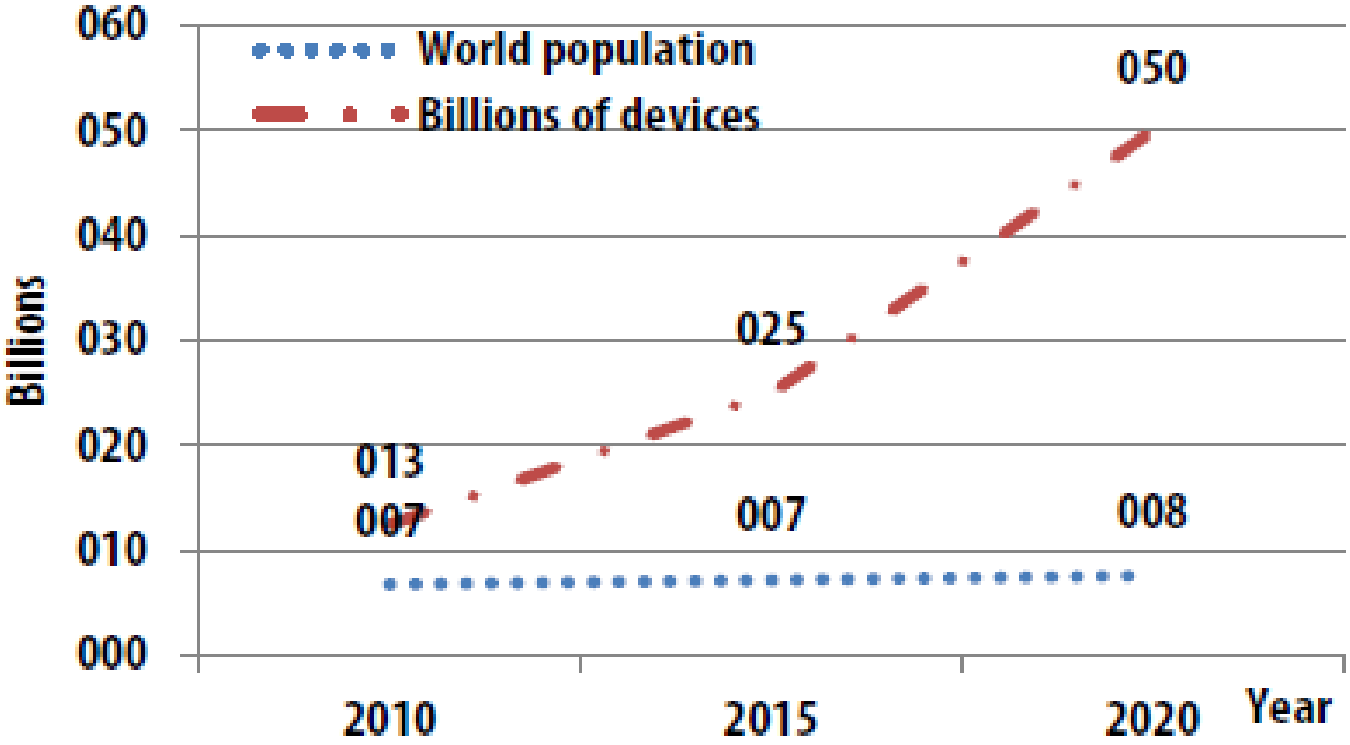
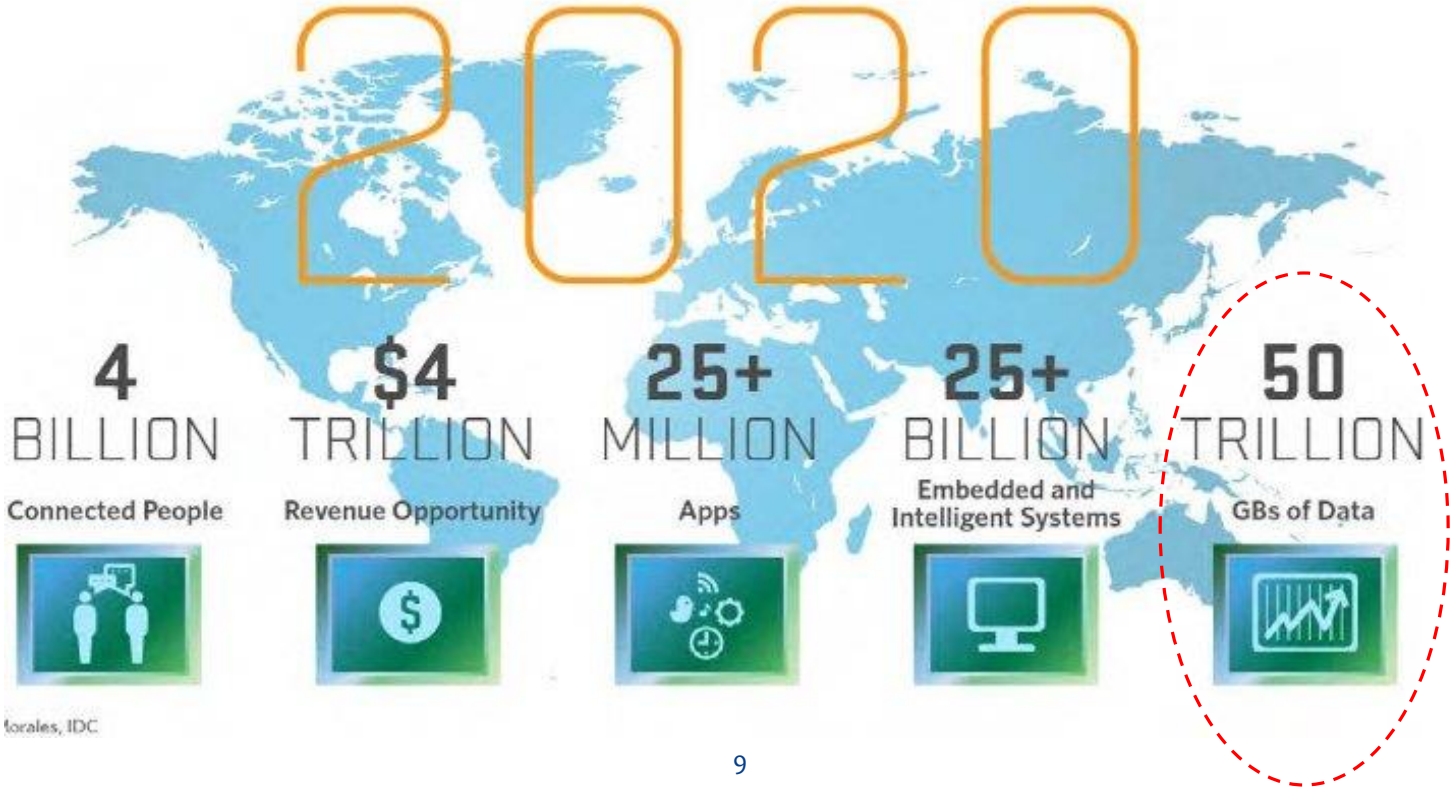


Figure 1. IoT devices and the future evolution

Smart devices



Introduction

- ***Data***
  - Values or a set of values
  - Data item refers to single unit of values
- **Data item**
  - Group item :
    - Data item that can be subdivided into sub item.
    - Ex Name : First Name, Middle initial and Last Name
  - Elementary item:
    - Data item that can not be sub divided into sub item
    - Ex : card number / Bank Pass Book Number is treated as single item
- Collection of data are frequently organized into a hierarchy of **fields, records and files**

Introduction

- Entity
  - Something that has certain attributes or properties which may be assigned values
  - Values may be numeric or non-numeric
- Ex The employee of an organization
  - **Attributes** Name Age Gender Employee Code
  - **Values** John 33 M 3472

Introduction

- **Field ,Record and File**
- **Field**
  - a single elementary unit of information representing an attribute of an entity
- **Record**
  - the collection of field values of a given entity
- **File**
  - the collection of records of the entities in a given entity set

Name	Age	Gender	Roll Number	Branch
A	17	M	109cs0132	CSE
B	18	M	109ee1234	EE
C	19	F	109ce0012	CE
D	20	F	108mm0132	MM

Introduction

- **Entity Set**

- Entity with similar attributes (e. g all employees of an organization) form an **entity set**

- Each attribute of an entity set has a **range of values** [the set of possible values that could be assigned to the particular attribute]

- **Information**: Data with given attribute or processed data

Introduction

- **Record**
- Record may be of fix and variable length
- Fixed Length Record
  - All records contain the same amount of data items with the same amount of space assigned to each data item
- Variable Length Record
  - File records may contain different lengths.
  - e.g student record usually have variable lengths .since different students take different number of courses
  - Usually variable length records have a minimum and a maximum length

Introduction to Data Structures

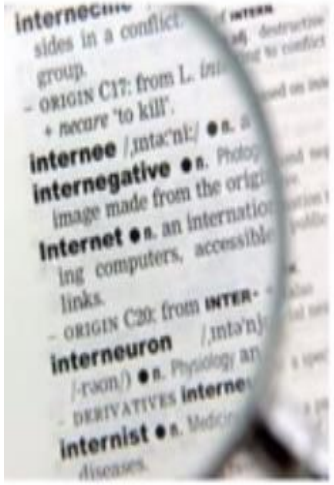
- How do we organize information so that we can find, update, add and delete portions of it efficiently?

- "The data structures deal with the study of how the data is **organized** in the memory, how efficiently it can be **retrieved** and **manipulated** and the possible ways in which different data items are logically related".

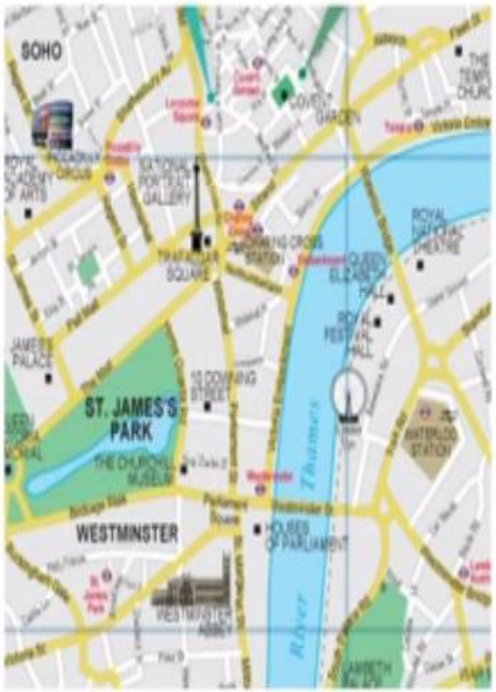
Introduction to Data Structures

It’s an agreement about:

- How to store a collection of objects in memory
- What operations we can perform on that data
- The algorithms for those operations
- How time and space efficient those algorithms are



English Dictionary



City map

ABC Hardware

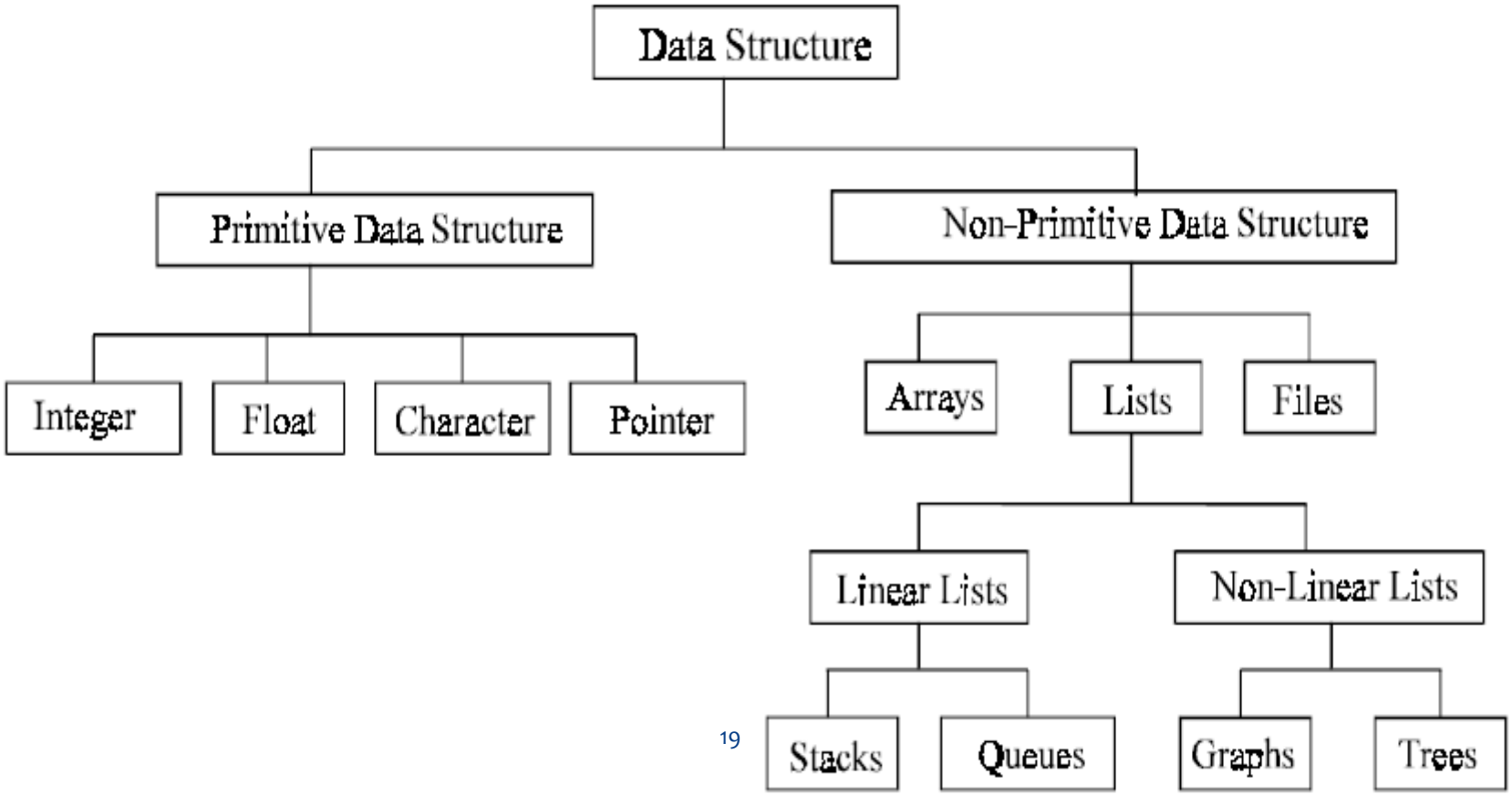
Cash Book - 03/01/2013 to 03/31/2013

S. no.	Date	Particulars	Debit	Credit
1	03/01/2013	Opening balance		50000
2	03/02/2013	Transport bill	2000	
3	03/07/2013	Goods sales		1500
4	03/08/2013	Bank Loan		5000
5	03/15/2013	Goods sales		1000
6	03/17/2013	Electiricty bill	1200	
7	03/21/2013	Good sales		1200
8	03/25/2013	Hardware purchase	500	
9	03/29/2013	Employee salary	20000	
10	03/31/2013	Closing Balance	35000	
		Total	58,700	58,700

Data Structures

- The way in which the data is organized affects the performance of a program for different tasks.
- Computer programmers decide which data structures to use based on the **nature of the data** and the **processes** that need to be performed on that data.
- They can be classified as:
  - Primitive data structures
  - Non primitive data structure.

Classifications



Classifications

- **Primitive data structure**
  - Basic data types that are available in most of the programming languages. The primitive data types are used to represent single values.
  - These are data structures that can be manipulated directly by machine instructions.
  - Primitive types are also known as built-in types or basic types.
  - In C language, the different primitive data structures are int, float, char, double.

- **Non primitive data structures**
  - The data types that are derived from primary data types are known as non-Primitive data types. These data types are used to store group of values.
  - These are data structures that can not be manipulated directly by machine instructions. Arrays, linked lists, files etc., are some of non-primitive data structures and are classified into **linear data structures** and **non-linear data structures**.

Linear and non- linear data structures

- The data structures that show the relationship of logical adjacency between the elements are called linear data structures.
- Otherwise, they are called non-linear data structures.
- Different linear data structures are stacks, queues, linear linked lists such as singly linked list, doubly linked linear lists etc.
- Trees, graphs and files are non-linear data structures.

Common Data Structures

- Array
- Stack
- Queue
- Linked List
- Tree
- Heap
- Hash Table
- Priority Queue

Operations

- Add
  - Index
  - Key
  - Position
  - Priority
- Get
- Change
- Delete

Examples

1. How does Google quickly find web pages that contain a search term?

2. What's the fastest way to broadcast a message to a network of computers?

3. How can a subsequence of DNA be quickly found within the genome?

4. How does your operating system track which memory (disk or RAM) is free?

5. In the game Half-Life, how can the computer determine which parts of the scene are visible?

Suppose You're Google Maps...

- You want to store data about cities (location, elevation, population)...

- What kind of operations should your data structure(s) support?

Operations to support the given scenarios...

- Finding addresses on map?
  - Lookup city by name...
- Mobile iPhone user?
  - Find nearest point to me...
- Car GPS system?
  - Calculate shortest-path between cities...
  - Show cities within a given window...
- Political revolution?
  - Insert, delete, rename cities

Data Organizing Principles

- **Ordering**
  - Put keys into some order so that we know something about where each key is, relative to the other keys.
  - Phone books are easier to search because they are alphabetized.

- **Linking**
  - Add pointers to each record so that we can find related records quickly.
  - E.g. The index in the back of book provides links from words to the pages on which they appear.

- **Partitioning**
  - Divide the records into 2 or more groups, each group sharing a particular property.
  - E.g. Multi-volume encyclopedias (Aa-Be, W-Z)
  - E.g. Folders on your hard drive

Ordering

Pheasant,	10
Grouse,	89
Quail,	55
Pelican,	3
Partridge,	32
Duck,	18
Woodpecker,	50
Robin,	89
Cardinal,	102
Eagle,	43
Chicken,	7
Pigeon,	201
Swan,	57
Loon,	213
Turkey,	99
Albatross,	0
Ptarmigan,	22
Finch,	38
Bluejay,	24
Heron,	70
Egret,	88
Goose,	67

Sequential Search – O(n)

Albatross,	0
Bluejay,	24
Cardinal,	102
Chicken,	7
Duck,	18
Eagle,	43
Egret,	88
Finch,	38
Goose,	67
Grouse,	89
Heron,	70
Loon,	213
Partridge,	32
Pelican,	3
Pheasant,	10
Pigeon,	201
Ptarmigan,	22
Quail,	55
Robin,	89
Swan,	57
Turkey,	99
Woodpecker,	50

(2) (3) (4) (1)

Search for "Goose"

Binary Search O(log n)

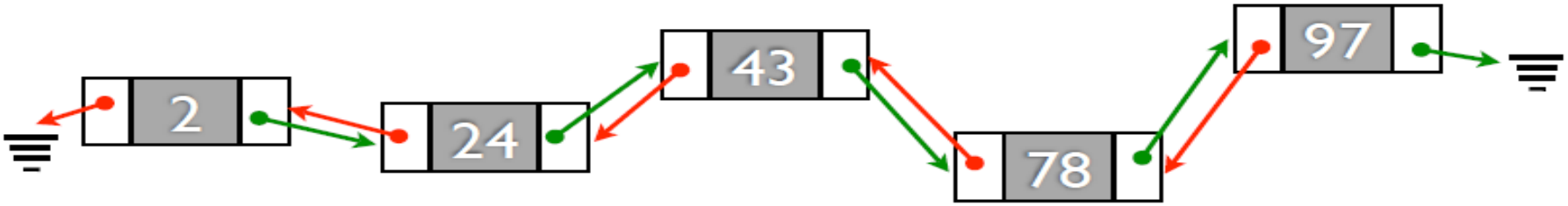
Every step discards half the remaining entries:

$$n / 2^k = 1$$

$$2^k = n$$

$$k = \log n$$

Linking

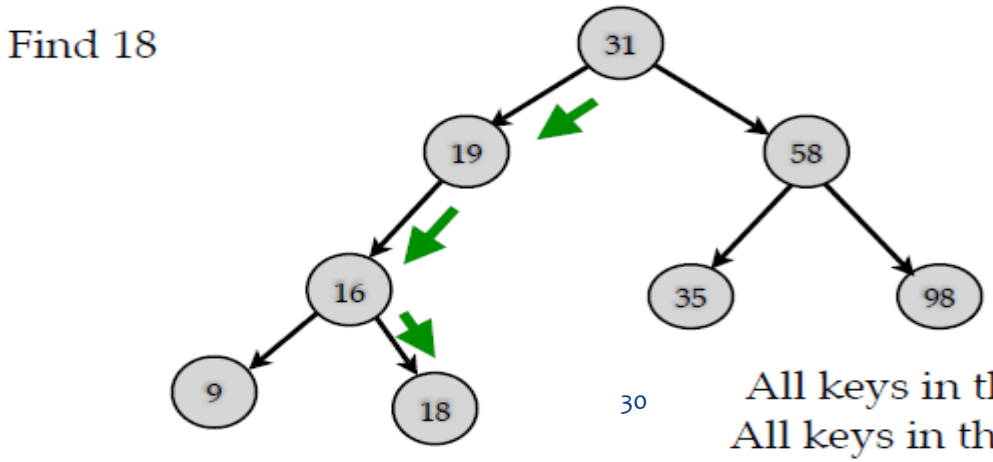


- Records located any where in memory
- Green pointers give "next" element
- Red pointers give "previous" element
- Insertion & deletion easy if you have a pointer to the middle of the list

- Don't have to know size of data at start
- Pointers let us express relationships between pieces of information.

Partitioning

- Ordering implicitly gives a partitioning based on the "<" relation.
- Partitioning usually combined with linking to point to the two halves.
- Prototypical example is the Binary Search Tree:



All keys in the left subtree are < the root
All keys in the right subtree are ≥ the root