41

A **Hash Table** is a data structure that stores values which have keys associated with each of them. Furthermore, it supports lookup efficiently if we know the key associated with the value. Hence it is very efficient in inserting and searching, irrespective of the size of the data.

**Direct Addressing** uses the one-to-one mapping between the values and keys when storing in a table. However, there is a problem with this approach when there is a large number of key-value pairs. The table will be huge with so many records and may be impractical or even impossible to be stored, given the memory available on a typical computer. To avoid this issue we use **hash tables**.

### Hash Function

A special function named as the **hash function** (**h**) is used to overcome the aforementioned problem in direct addressing.

In direct accessing, a value with key **k** is stored in the slot **k**. Using the hash function, we calculate the index of the table (slot) to which each value goes. The value calculated using the hash function for a given key is called the **hash value** which indicates the index of the table to which the value is mapped.

$$h(k) = k \% m$$

- **h:** Hash function
- **k:** Key of which the hash value should be determined
- **m:** Size of the hash table (number of slots available). A prime value that is not close to an exact power of 2 is a good choice for **m**.

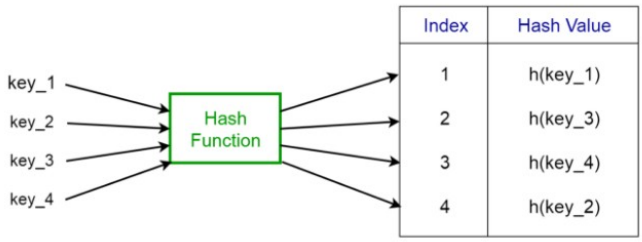


Fig 5. Representation of a Hash Function (Image by author)

### Applications of hash tables

- Used to implement database indexes.
- Used to implement associative arrays.
- Used to implement the "set" data structure.

42

A **tree** is a hierarchical structure where data is organized hierarchically and are linked together. This structure is different from a linked list whereas, in a linked list, items are linked in a linear order.

Various types of trees have been developed throughout the past decades, in order to suit certain applications and meet certain constraints. Some examples are binary search tree, B tree, treap, red-black tree, splay tree, AVL tree and n-ary tree.

### Binary Search Trees

A **binary search tree (BST)**, as the name suggests, is a binary tree where data is organized in a hierarchical structure. This data structure stores values in sorted order.

Every node in a binary search tree comprises the following attributes.

1. **key**: The value stored in the node.
2. **left**: The pointer to the left child.
3. **right**: The pointer to the right child.
4. **p**: The pointer to the parent node.

A binary search tree exhibits a unique property that distinguishes it from other trees. This property is known as the **binary-search-tree property**.

Let **x** be a node in a binary search tree.

- If **y** is a node in the **left** subtree of x, then **y.key ≤ x.key**
- If **y** is a node in the **right** subtree of x, then **y.key ≥ x.key**

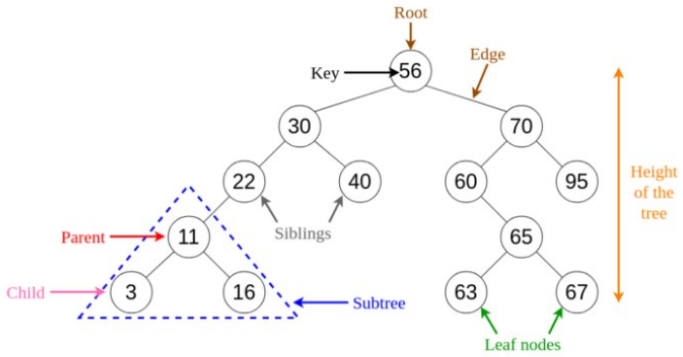


Fig 6. Visualization of Basic Terminology of Trees (Image by author)

### Applications of trees

- **Binary Trees**: Used to implement expression parsers and expression solvers.
- **Binary Search Tree**: used in many search applications where data are constantly entering and leaving.
- **Heaps**: used by JVM (Java Virtual Machine) to store Java objects.
- **Treaps**: used in wireless networking.

Excerpts from Mallawaarachchi, V. (2020). *8 Common Data Structures Every Programmer Must Know*. towardsdatascience.com

43

A **Heap** is a special case of a binary tree where the parent nodes are compared to their children with their values and are arranged accordingly.

Let us see how we can represent heaps. Heaps can be represented using trees as well as arrays. Figures 7 and 8 show how we can represent a binary heap using a binary tree and an array.

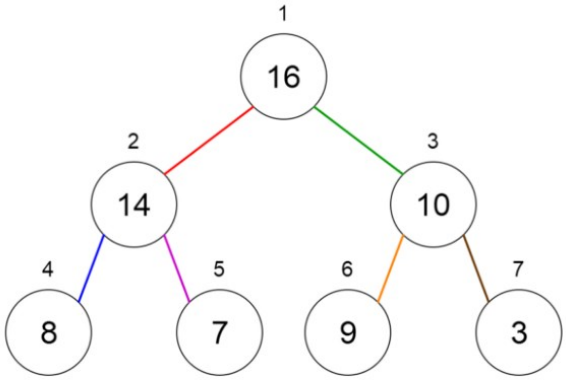


Fig 7. Binary Tree Representation of a Heap (Image by author)

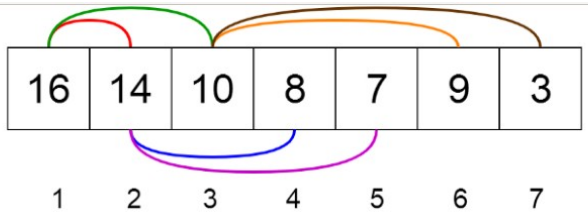


Fig 8. Array Representation of a Heap (Image by author)

Heaps can be of 2 types.

1. **Min Heap** — the key of the parent is less than or equal to those of its children. This is called the **min-heap property**. The root will contain the minimum value of the heap.
2. **Max Heap** — the key of the parent is greater than or equal to those of its children. This is called the **max-heap property**. The root will contain the maximum value of the heap.

### Applications of heaps

- Used in **heapsort algorithm**.
- Used to implement priority queues as the priority values can be ordered according to the heap property where the heap can be implemented using an array.
- Queue functions can be implemented using heaps within **O(log n)** time.
- Used to find the $k^{th}$ smallest (or largest) value in a given array.

44

A **graph** consists of a finite set of **vertices** or nodes and a set of **edges** connecting these vertices.

The **order** of a graph is the number of vertices in the graph. The **size** of a graph is the number of edges in the graph.

Two nodes are said to be **adjacent** if they are connected to each other by the same edge.

### Directed Graphs

A graph **G** is said to be a **directed graph** if all its edges have a direction indicating what is the start vertex and what is the end vertex.

We say that **(u, v)** is **incident from** or **leaves** vertex **u** and is **incident to** or **enters** vertex **v**.

**Self-loops**: Edges from a vertex to itself.

### Undirected Graphs

A graph **G** is said to be an **undirected graph** if all its edges have no direction. It can go in both ways between the two vertices.

If a vertex is not connected to any other node in the graph, it is said to be **isolated**.

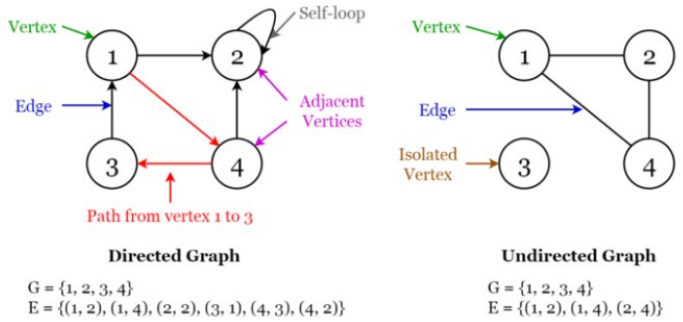


Fig 9. Visualization of Terminology of Graphs (Image by author)

### Applications of graphs

- Used to represent social media networks. Each user is a vertex, and when users connect they create an edge.
- Used to represent web pages and links by search engines. Web pages on the internet are linked to each other by hyperlinks. Each page is a vertex and the hyperlink between two pages is an edge. Used for Page Ranking in Google.
- Used to represent locations and routes in GPS. Locations are vertices and the routes connecting locations are edges. Used to calculate the shortest route between two locations.

Excerpts from Mallawaarachchi, V. (2020). *8 Common Data Structures Every Programmer Must Know*. towardsdatascience.com