Graph: consists of vertices and edges.

Edge (arcs): each edge is a pair (v, w), v, w ∈ V.

If a pair is ordered, the graph is **directed** (digraphs).

Path: sequences of vertices.

Simple path: all vertices are distinct (no dupes).

Cycle: from one node, you can find a path back to that node. Begins and ends at one vertex.

Weights/costs: assign weights/costs to an edge (distance between vertices).

Weighted path: each edge of the graph is assigned a cost.

A tree is a graph with no cycles.

Minimum spanning tree

Exam 2

Ch4

AVLTree, Huffman, expression trees, B-Tree, 2-4 tree

Ch5

Hashing (linear probing, quadratic probing, separate chaining), those are open addressing techniques

Priority Queue, heap implementation, heapsort, mergesort, quicksort, radixsort, analysis of mergesort and quicksort

linear probing: say x%17, add 19, 34, 49, 16, 32.

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 |
| 34 | 15 | 19 |  |  |  |  |  |  |  |  |  |  |  |  | 49 | 16 |

19 % 17 = 2.

34 % 17 = 0.

49 % 17 = 15.

16 % 17 = 16.

32 % 17 = 15. 15 is already occupied, then add 1 to the hash function and hash again.

15 + 1 = 16, 16 % 17 = 16. 16 is already occupied, add 2 to 15.

15 + 2 = 17, 17 % 17 = 0. 0 is occupied, add 3 to 15.

15 + 3 = 18, 18 % 17 = 1.

If table is full, rehash and enlarge the array.

Quadratic probing adds 12, 22, 32, etc to the element and rehash.