Objectives

- Be able to describe what is meant by a “balanced” binary search tree, and to discuss the impact of maintaining balance on the efficiency of a binary search tree
- Be able to define the terms graph, vertex, edge, and weight
- Be able to define the terms path and cycle
- Be able to describe the Graph abstract data type (ADT)
- Be able to describe the adjacency matrix representation of a graph
- Be able to describe the adjacency list representation of a graph, and to show its implementation in the Vertex class

__getitem__ and __setitem__

The “delete” operation

- Most difficult to code!

Search tree complexity analysis

Balanced Binary Search Trees

Graph Algorithms

- Terms vertex, edge, weight

- Example

Representing a graph

- Tuple (pair) (V, E), where V is a set of vertices and E is a set of edges

Paths and Cycles

The Graph ADT

- Graph()
- addVertex(v)
- addEdge(v1, v2, wt=None)
- getVertex(vKey)
- getVertices
- has(v)

The adjacency matrix representation

The adjacency list representation

Python Vertex class