Binary search tree rotations

- Constantly doing binary tree rotations can be
  - Costly
  - More error prone, just for the complexity of the rotations
  - Limiting in multi-threaded trees

- Rotations can lead to
  - A balanced tree, which makes searches faster
  - Balance robustness relative to the order in which data is added to the tree

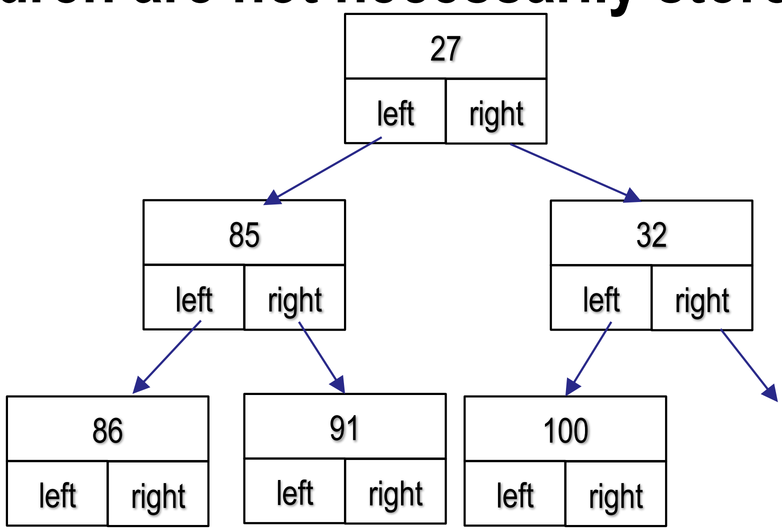
Min-Heap

- A heap is a data structure that allows you to remove the smallest element efficiently.
  - Not intended for you to search for elements
  - Not intended for you to remove arbitrary elements (although you can get the code to do it

- There is a variant called a Max-Heap to let you remove the largest element efficiently.

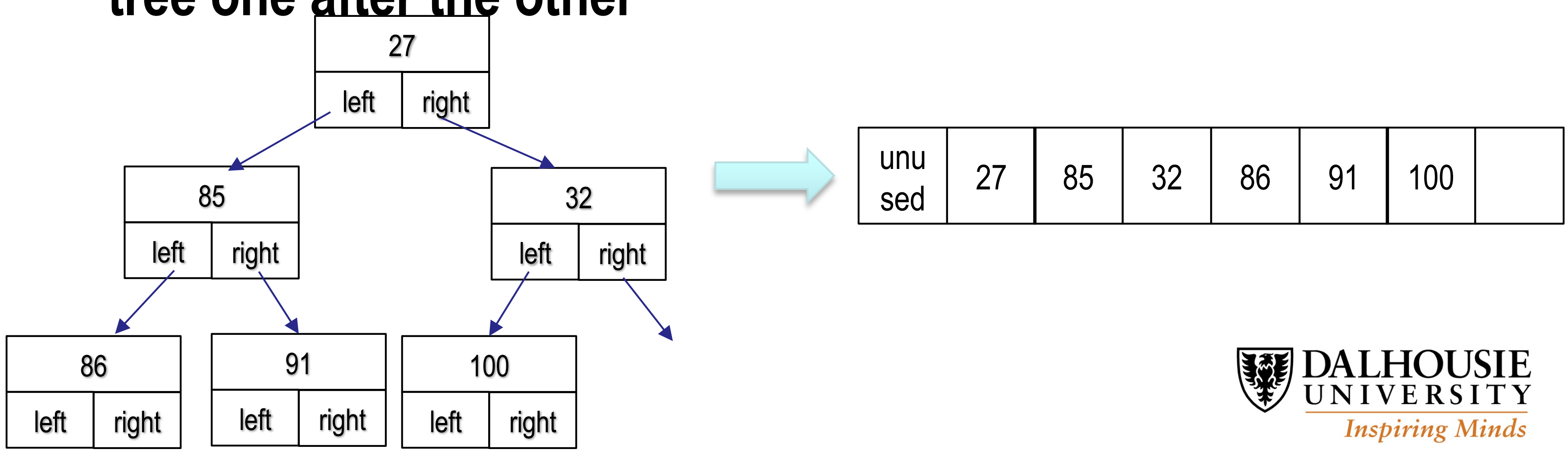
Min-Heap

- **Most often described / modeled as a binary tree:**
  - The parent is smaller than both children
  - The binary tree remains balanced
  - The children are not necessarily stored in any particular order



Min-Heap

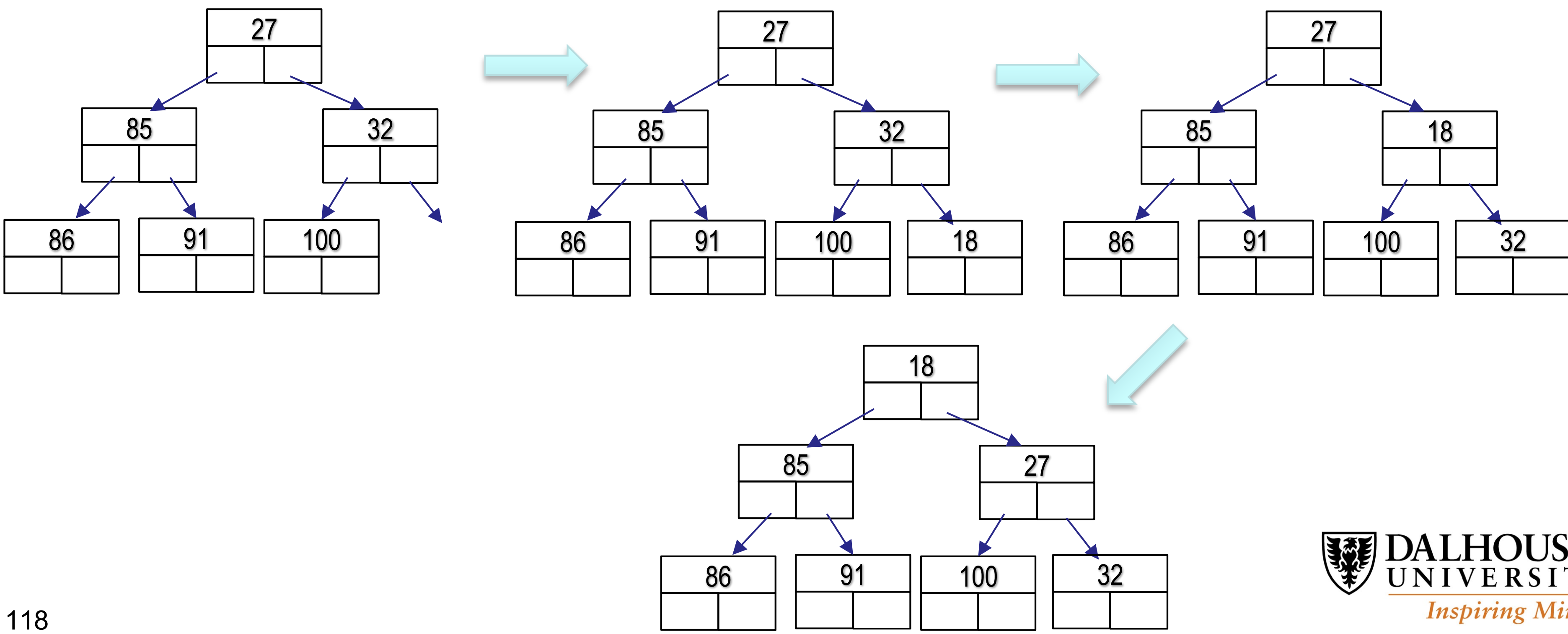
- The binary tree thought, as a complete binary tree, is often implemented through an array:
  - Store the root at array index 1
  - The children of node at index x are found at indices 2x and 2x+1
  - The array looks like you have stored the levels of the binary tree one after the other



DALHOUSIE UNIVERSITY *Inspiring Minds*

Min-Heap

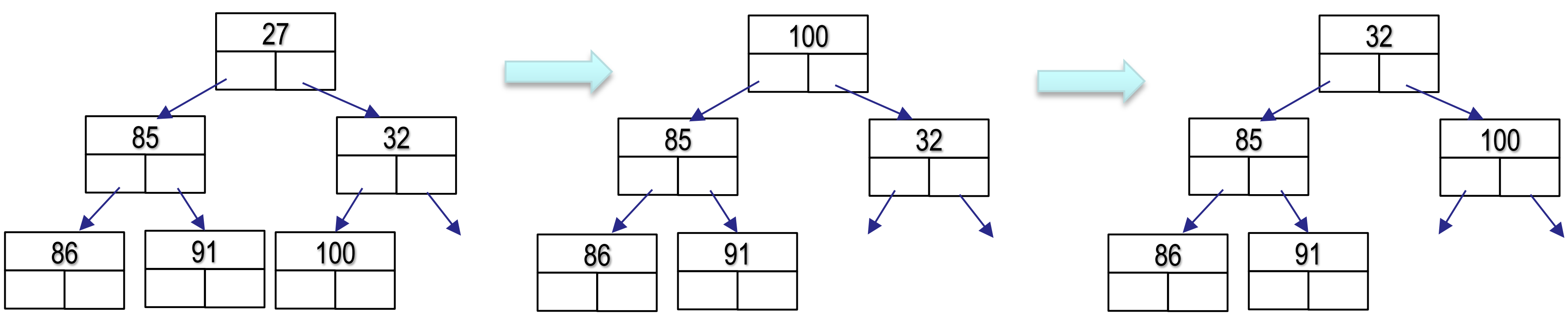
- To add an item:
  - Store it in the next spot in the bottom level
  - Continually swap it with its parent if it is smaller than the parent



DALHOUSIE UNIVERSITY *Inspiring Minds*

Min-Heap

- To remove an item:
  - Remove the top-most item.
  - Move the last item in the lowest level to the top
  - Continually compare this moved item to its two children and swap it with its smallest child



DALHOUSIE UNIVERSITY
Inspiring Minds

Min-Heap

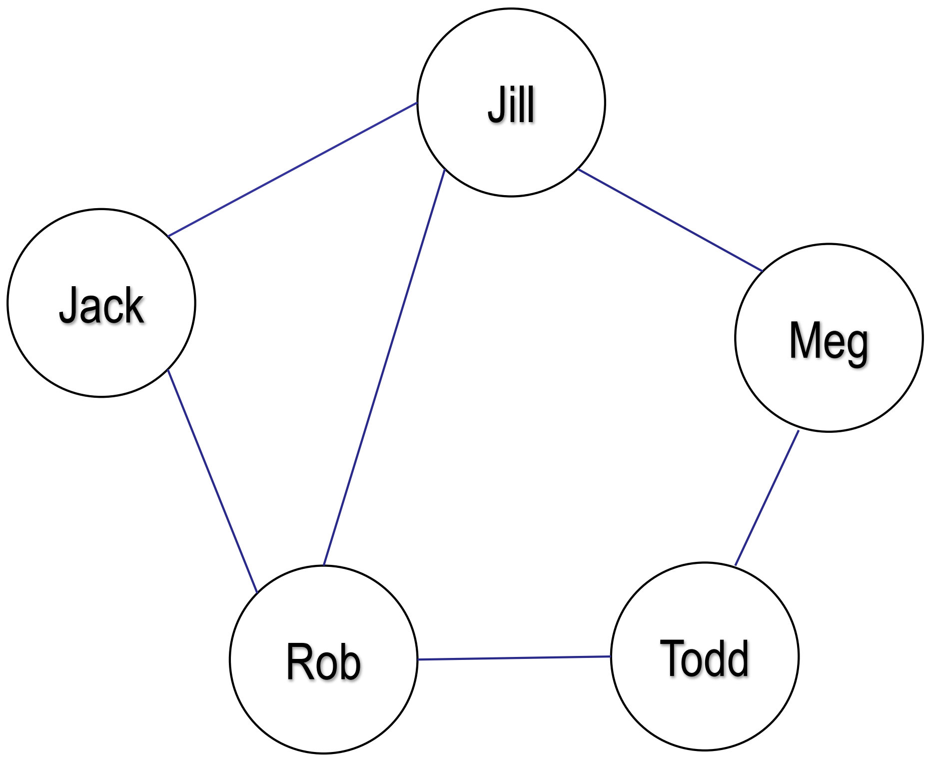
- **A min-heap is a common implementation of a priority queue.**

Graph

- An ADT for a relations between elements.
- Defined by a set of vertices V and a set of edges E where E captures the relations (subset of V x V)
  - Operators include add/remove vertex or edge, access adjacent vertex, test if an edge exists, and traverse
- Example:
  - Graph of people who know one another
    - V = set of people, E = edges between people who know each other

Graph

- Graph of people who know each other



Sample uses of graphs

- Finite state machine
- Transition model representation
- Process flow
- Computer network topology
- Neural network
- Language structure description
- …

DALHOUSIE UNIVERSITY
Inspiring Minds

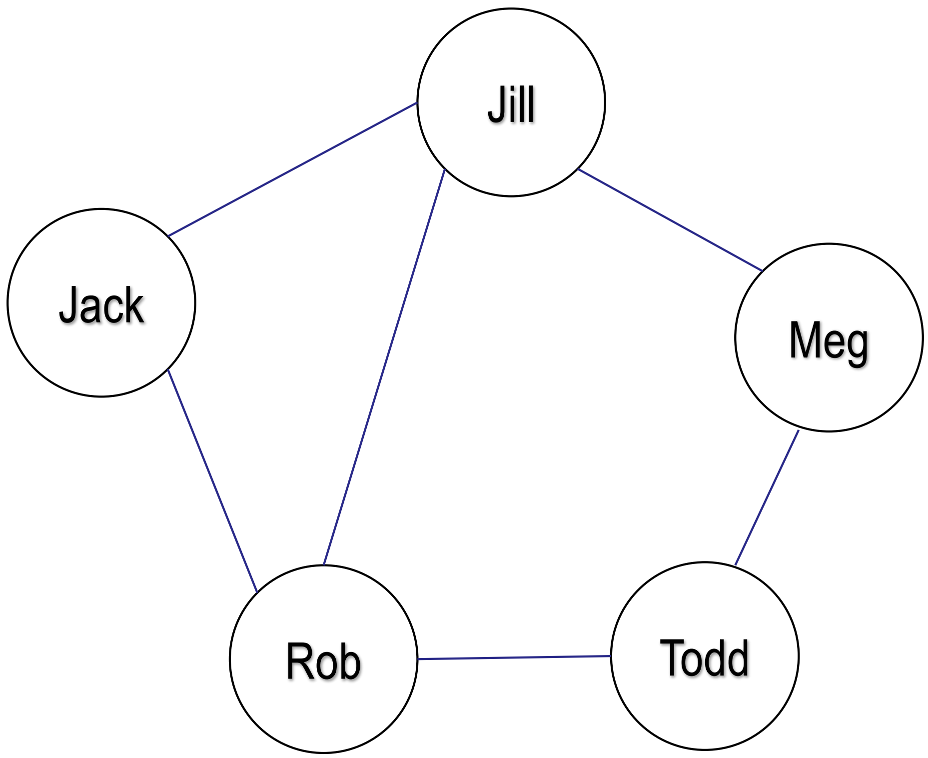
Graph representation

- What data structures let you store a graph?
  - Adjacency matrix
    - Have a |V| x |V| matrix, with each element of V represented in a row and a column
    - Put a value of 1 in the matrix where the element in the row and the column are related
  - Incidence matrix
    - Have a |V| x |E| matrix, with each element of V represented in a row and each edge represented by a column
    - Put a value of 1 in the matrix where the row is an endpoint of the edge that is given by the column
  - Adjacency list
    - For each vertex, store the list of vertices to which it is related
- Why choose one over another?

DALHOUSIE UNIVERSITY
Inspiring Minds

Graph

- Graph of people who know each other



Adjacency Matrix

	Jack	Jill	Meg	Todd	Rob
Jack	0	1	0	0	1
Jill	1	0	1	0	1
Meg	0	1	0	1	0
Todd	0	0	1	0	1
Rob	1	1	0	1	0

Incidence Matrix

	E1	E2	E3	E4	E5	E6
Jack	1	1				
Jill	1		1	1		
Meg				1	1	
Todd					1	1
Rob		1	1			1

DALHOUSIE UNIVERSITY
Inspiring Minds

Adjacency List

Vertex	Neighbours
Jack	{Jill, Rob}
Jill	{Jack, Meg, Rob}
Meg	{Jill, Todd}
Todd	{Meg, Rob}
Rob	{Jack, Jill, Todd}

Types of Graphs

- Edge influence
  - Undirected: you can traverse an edge in any direction
  - Directed: there is just one way by which you can follow an edge
- Connectivity
  - Connected: you can get from one vertex to any other vertex
  - Unconnected: there are some vertices that can't reach one another

Types of Graphs

- Edge multiplicity
  - Simple: there are no edges back to the same vertex (a loop) and at most one edge between pairs of vertices
  - Multi-edge: you can have loops and several edges between the same pair of vertices.
- Edge weight
  - Unweighted: no values associated with edges
  - Weighted: edges can have a “weight”, either as a cost to traverse, a number of times that you can use it, a capacity for the edge, …

Common graph problems

- Traversals (depth-first or breadth-first)
- Shortest path (Dijkstra's algorithm)
- Graph cycle detection (variant of depth-first search)
- Minimum spanning tree – get rid of cycles
  - Prim's algorithm, Kruskal's algorithm
- Connectivity
- Network flow
- Topological sorting