Data Structure: Review

Fatemeh Panahi
Department of Computer Science
University of Toronto
CSC263-Fall 2017

Complexity Review

Any problem has some input.
Consider a set of all possible values for different input that makes the sample space:

- Best Case: Minimum complexity
- Average Case: Expected value over the sample space by considering the probability distribution over inputs. $E(X) = \sum x_i \, P(x_i)$
- Worst case: Maximum complexity

For randomized algorithm the algorithm makes random variable itself.

- Expected running time: expected value over the random variable generated by the algorithm

- Amortized Analysis: A sequence of operations
    - Aggregate, Accounting method

Asymptotic notations

$f(n) = O(g(n)) \rightarrow \exists n_0, c_0$ such that for $n > n_0$ $f(n) \leq c_0\, g(n)$

$f(n) = \Omega(g(n)) \rightarrow \exists n_0, c_0$ such that for $n > n_0$ $f(n) \geq c_0\, g(n)$

$f(n) = \Theta(g(n)) \rightarrow \exists n_0, c_1, c_2$ such that for $n > n_0$ $c_1 g(n) \leq f(n) \leq c_2\, g(n)$

Order of growth of some common functions

$$O(1) < O(\log^* n) < O(\log n) < O(\sqrt{n}) < O(n) < O(n \log n) < O(n^2) < O(n^3) < O(2^n)$$

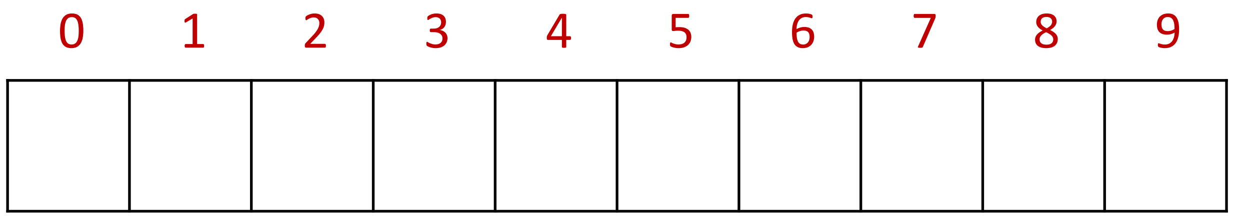
Examples:

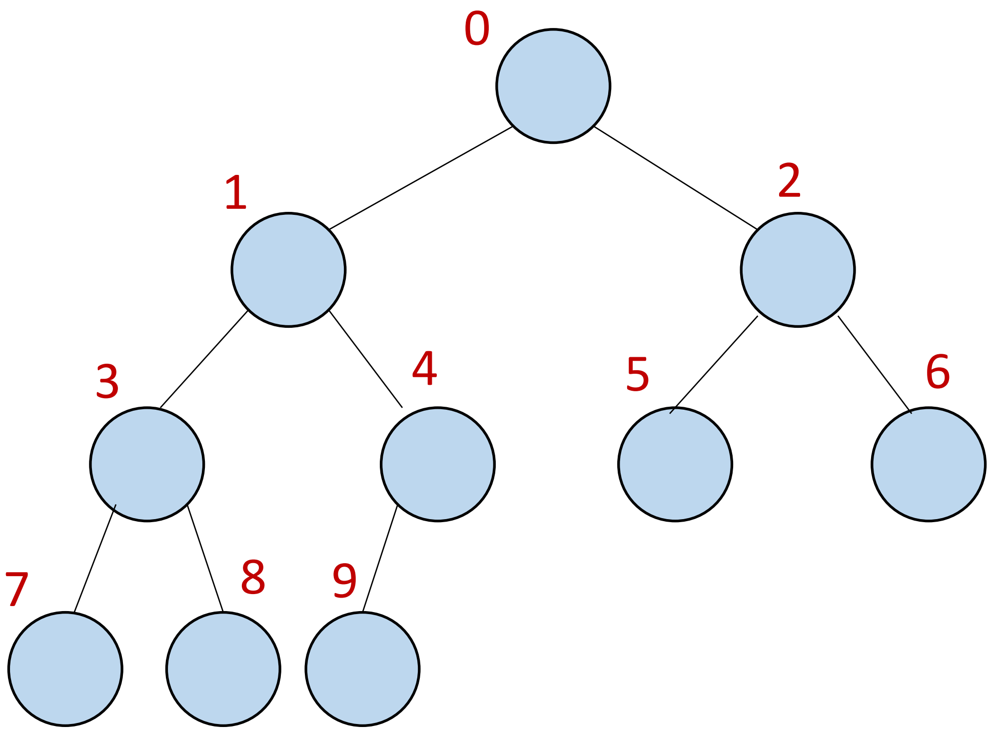
$$10\, n^2 + 3 = O(n^2) = O(n^3)$$
$$10\, n^2 + 3 = \Omega(n^2) = \Omega(n) = \Omega(1)$$
$$40\, n^2 + 10\, n \log 5n + 100\sqrt{n} \log n + 5n + 200 = \Theta(n^2)$$

Priority Queues: Heaps

- Data structure: an array that represents a full tree

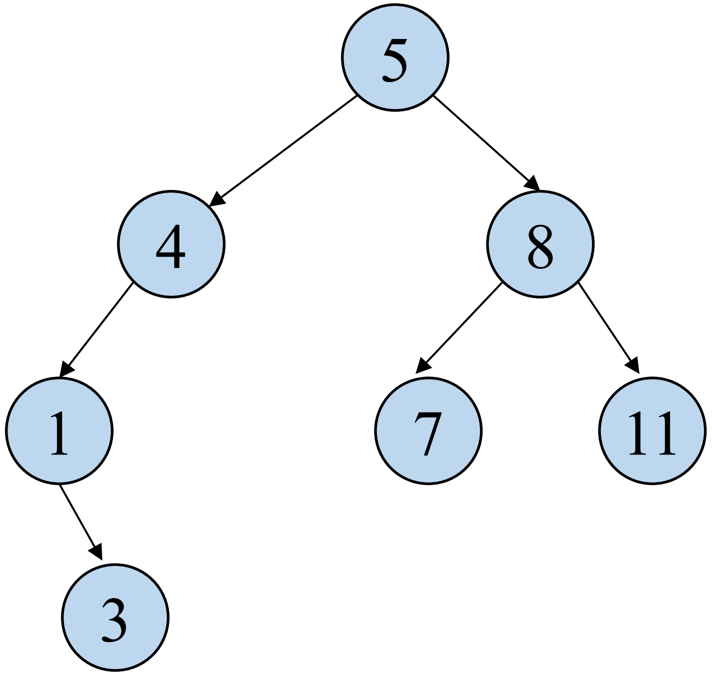
- Extract-Max, Insert, Delete: $O(\log n)$
- Decrease key: O(log n)
- Build max-heap: $O(n)$





Dictionaries: BSTs

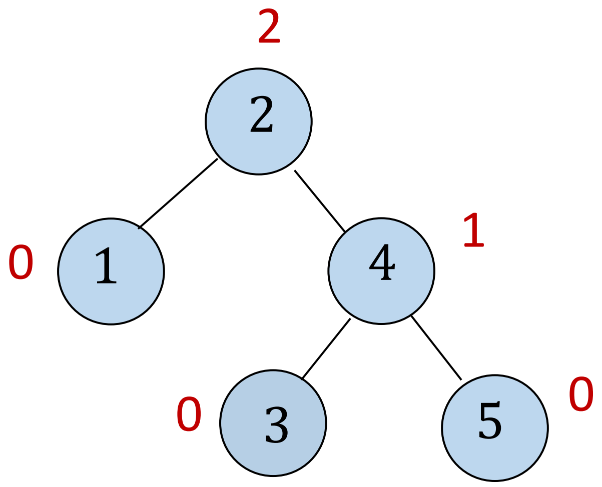
- Data structure: a linked structure that represents a tree which is not always full or balanced



- Successor and predecessor $O(h) = O(n)$
- Insert, Search, Delete: $O(h) = O(n)$

Balanced Trees, BST Augmentation

- AVL trees: For all the nodes the difference between the height of left and right subtree is at most 1.



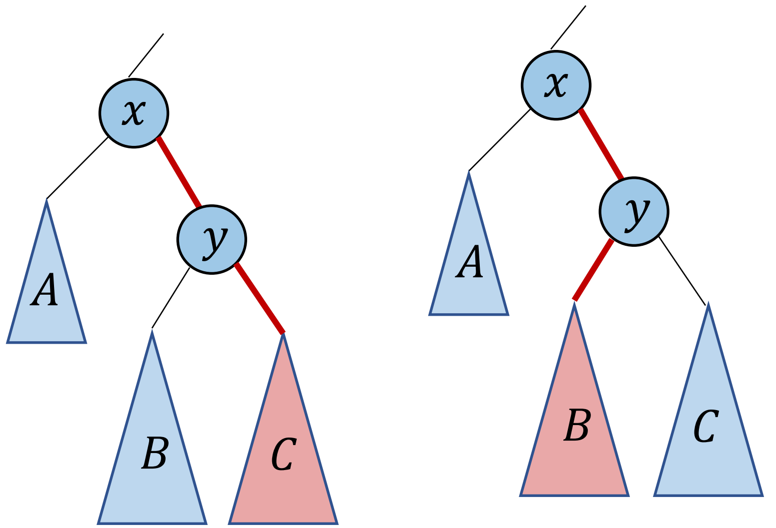
- Successor and predecessor $O(\log n)$
- Insert, Search, Delete: $O(\log n)$

- For each insert or delete
  at most 2 rotations

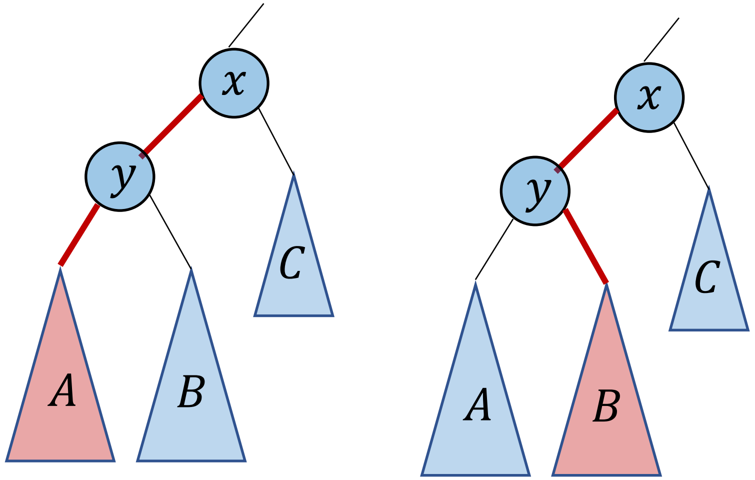
- Each rotation: $O(1)$

AVL-rebalancing

- Lowest unbalanced node:
  - Right heavy, right child: right heavy: single left rotation
  - Right heavy, right child: left heavy (Zig zag): double right-left rotation

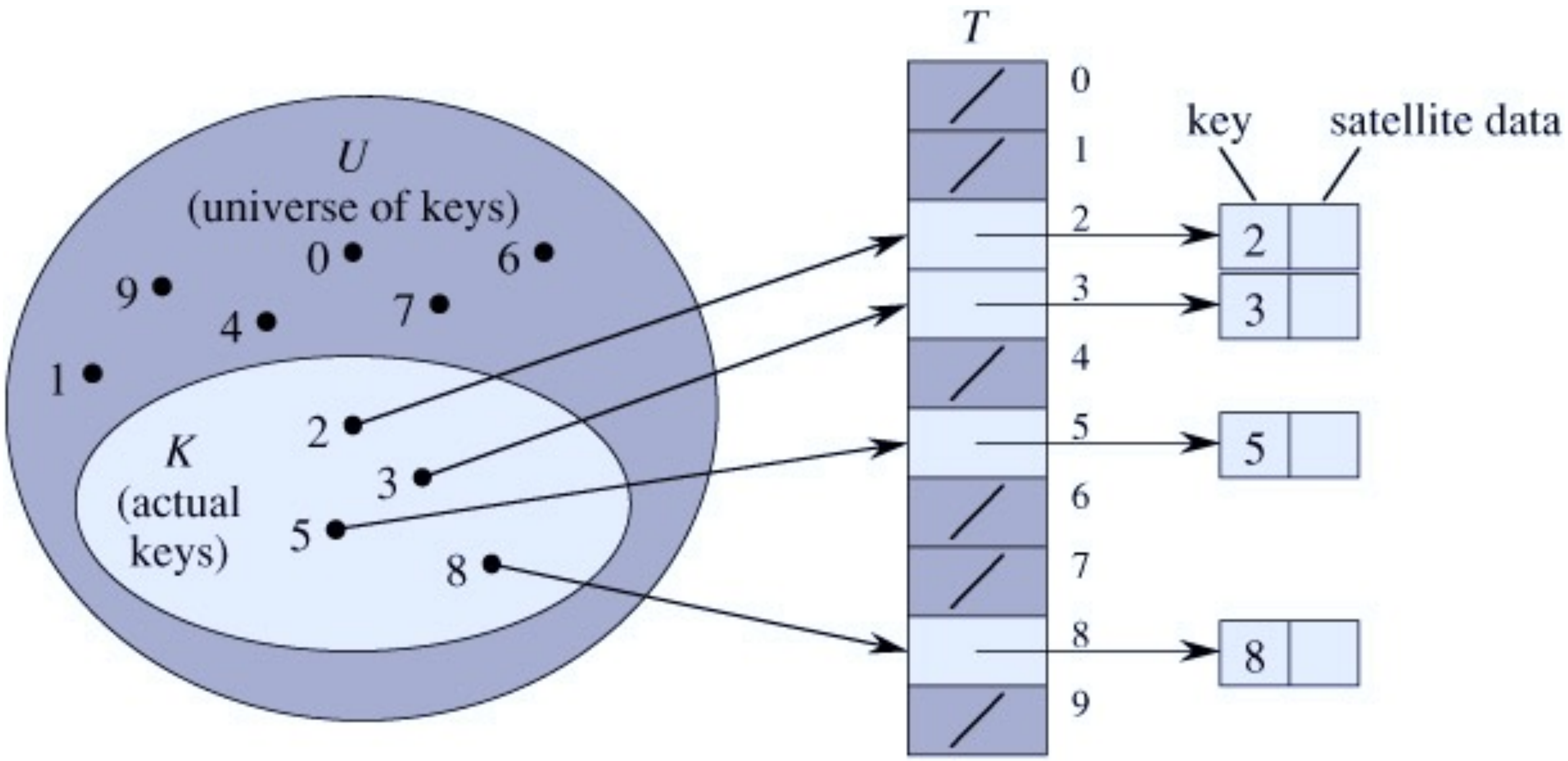


- The other case is symmetric



Hashing

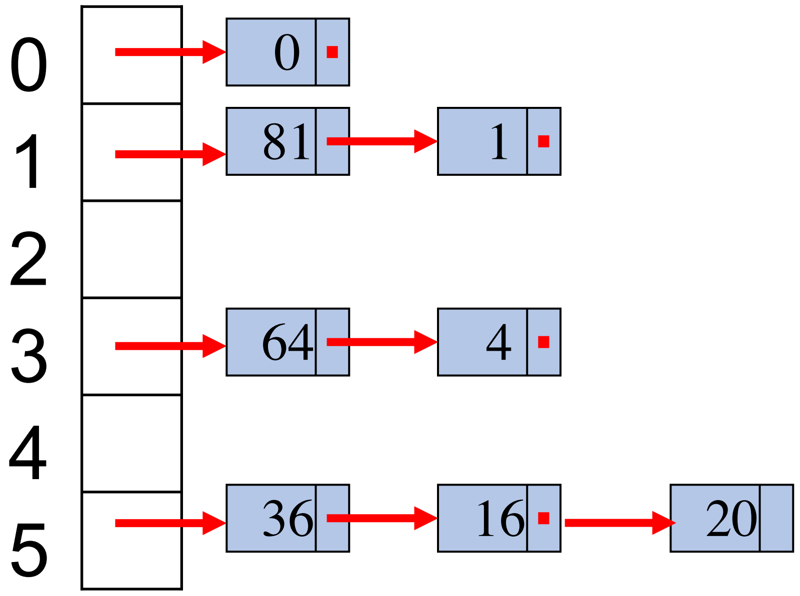
- Direct access table



Hashing

- Hash function maps a key to a value

- Collision handling
    - Chaining



- Open addressing
    - Linear probing
      $$h(k,i) = (h'(k) + i) \bmod m$$
    - Quadratic probing
      $$h(k,i) = (h'(k) + c_1 i + c_2 i^2) \bmod m$$
    - Double hashing
      $$h(k,i) = (h_1(k) + i h_2(k)) \bmod m$$

Quicksort

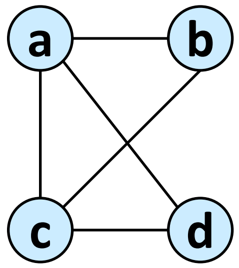
- Running time
  - Worst case: $\Theta(n^2)$
  - Average case for randomized version: $\Theta(n \log n)$
  - Best case: $\Theta(n \log n)$

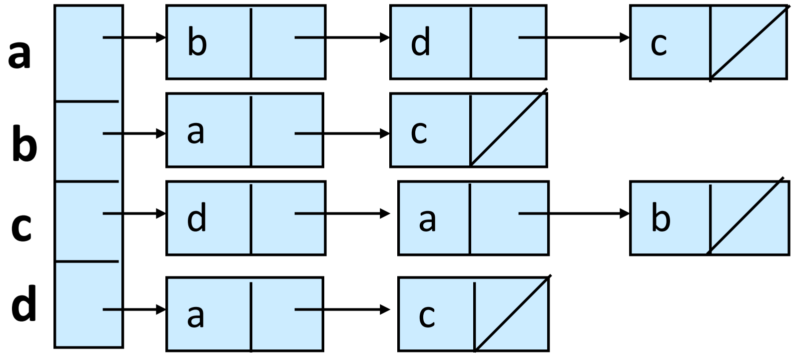
Graphs

- Types of graphs
  - Undirected: edge $(u, v) = (v, u)$; for all v, $(v, v) \notin E$ (No self loops)
  - Directed: (u, v) is edge from u to v, denoted as $u \rightarrow v$. Self loops are allowed.
  - Weighted: each edge has an associated weight
- If G is connected:
  - There is a path between every pair of vertices.
  - $E \geq V - 1$.
  - Furthermore, if $E = V - 1$, then G is a tree.
- $E = O(V^2)$

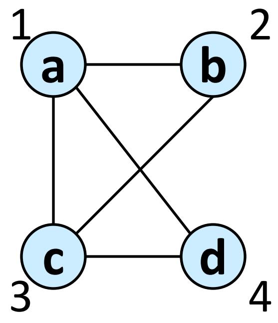
Graphs: representation

- Adjacency Lists.





- Adjacency Matrix.



	1	2	3	4
1	0	1	1	1
2	1	0	1	0
3	1	1	0	1
4	1	0	1	0

Graph Traverse: BFS, DFS

- To traverse means to visit the vertices in some systematic order.
- You should be familiar with various traversal methods for binary trees:
  - preorder: visit each node before its children.
  - postorder: visit each node after its children.
  - inorder (for binary trees only): visit left subtree, node, right subtree

- BFS:
  - Traverse a connected component graph and find the shortest path from the source to all nodes.
  - If the graph is not connected or is directed, does not traverse all the nodes
- DFS
  - Could traverse all the nodes even if not connected or directed.

BFS and DFS $\Theta(V + E)$ using the adjacency list.

Edge classification by DFS and BFS

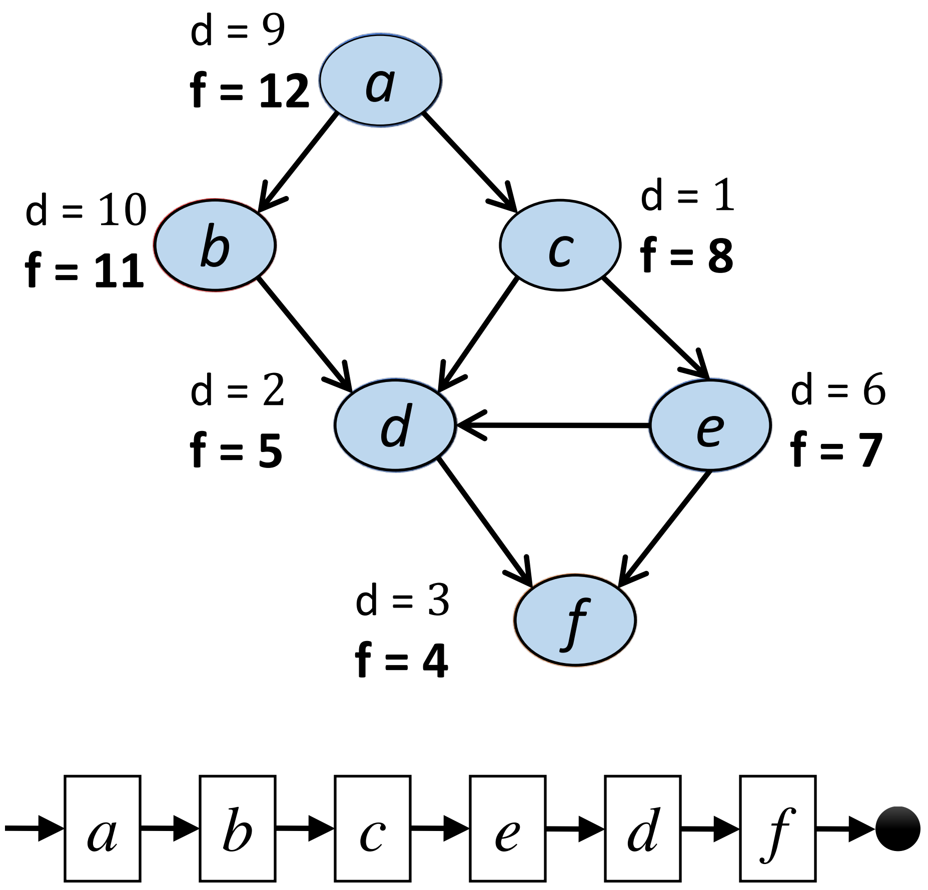
	Directed Graph	Undirected Graph
DFS	all	tree , back
BFS	tree, back, cross	tree, cross

a directed graph G is acyclic iff DFS on G yields no Back edges

Topological sort

- Is it possible to execute all the tasks in **G** in an order that respects all the precedence requirements given by the graph edges?
- The answer is "**yes**" *if and only if* the directed graph **G** has **no cycle**.

Such a **G** is called a Directed Acyclic Graph, or just a **DAG.**



Minimum Spanning Trees

- Cut, cross edge, light edge
- Idea: start with an empty set of edges and add light edges to MST

1. Prim's Algorithm: Using minheap $O(E \log V)$
2. Kruskal's algorithm: Using disjoint forest $O(E\ logV)$

- The lightest edge is always in MST.
- The heaviest edge is not always is MST.
- In general there might be more than one MST.
- If the weights are distinct there is a unique MST.

Disjoint Sets

- Link list Implementation

1. Make-Set(x): $\Theta(1)$
2. Find-Set(x): Θ (1)
3. Union(x,y): $\Theta(n)$

Total time m operations: Θ (m+n^2)

Heuristics:

Union by weight:

Union(x,y): O(log n) Amortized cost $O(m + n \log n)$

- Disjoint forest Implementation

1. Make-Set(x): Θ (1)
2. Find-Set(x): Θ (n)
3. Union(x,y): Θ (n)

Heuristics:

Union by ranks:

Find-Set(x): O(log n) Union(x,y): O(log n)

Union by ranks and Path Compression:

Total time m operations: Θ ($m \log^* n$)

Lower Bounds

- For problem $P, C(P)$ = best (minimum) worst-case running time of any algorithm that solves P.
- Techniques:
  - ✓ Information theory lower bounds
  - ✓ Adversary arguments
  - ✓ Reductions
- Finding the minimum in a set of n element cannot be better than $\Theta(n)$.
- Searching in a sorted list cannot be better than $\Theta(\log n)$.
- Comparison sort cannot be better than $\Theta(n \log n)$.
- Extract-max cannot be better than $\Theta(\log n)$.

Acknowledgements

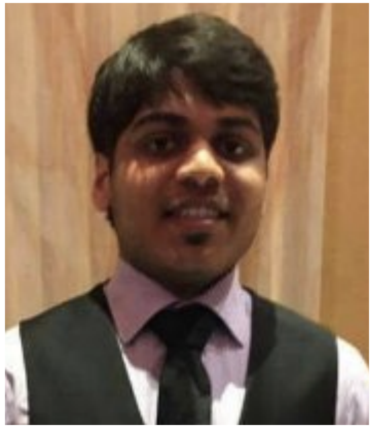
François Pitt

Faith Ellen

TAs

Robin Swanson

Sasa Milic



Roleen Nunes

- Tristan Aumentado-Armstrong
- Ruowei Jiang
- Morgan Shirley
- Ziqiao Meng
- Mohammad Amin Beiruti

Good luck ☺

- Final exam
- Internship and job Interviews
- Developing excellent software in your Start-up
- Impressing your boss