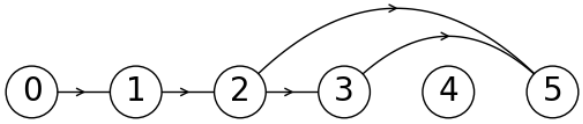
CS2302 - Data Structures

Spring 2020

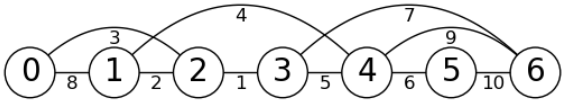
Exam # 3

'In my life, I have met many good people who fail a Data Structures exam, however, I have never met a good person who cheats in a Data Structures exam' - Mahatma Gandhi

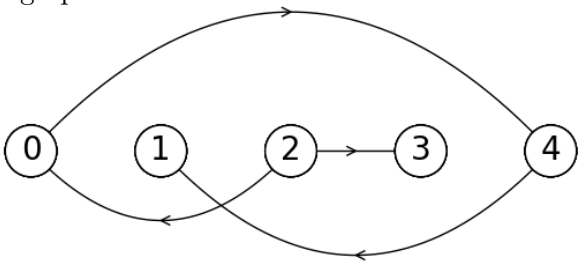
1. (20 points) A sink in a directed graph is a vertex with no outgoing edges. Write the function *count_sinks(G)* that receives a graph G represented as an adjacency list and returns the number of sinks that G contains. For example, for the graph below, *count_sinks(G)* should return 2, since vertices 4 and 5 are sinks.



2. (20 points) Write the function *first_kruskal_(G))* that receives an undirected weighted graph G represented as an adjacency list and returns a list of length 3 containing the first edge (in [source, dest, weight] format) that would be added to the minimum spanning tree of G by Kruskal's algorithm. For example, if G is the graph below, your function should return $[2, 3, 1]$



3. (20 points) Write the function *graph_from_prev(prev)* that receives the array (or list) *prev* computed by breadth-first search and returns an unweighted directed graph G represented by an adjacency matrix containing the vertices and edges encoded in *prev*. For example, if *prev = [2, 4, -1, 2, 0]*, your function should build and return the graph shown below.



4. (15 points) Write the function *num_sets(s)* that receives a disjoint set forest s and returns the number of sets it encodes.

5. (15 points) The function *subsetsum(S,g)* receives a set of positive integers S and an integer g and returns the subset of S whose elements add up to g, if it exists and *None* otherwise. Write the function *subsetsum_with_negatives(mySet,i)* that works even if S contains negative numbers. Hint: this can be done with a very small modification to the function provided in the starter code.

6. (20 points) The function *min_coins_greedy_with_max(C,D,Max)* is similar to the function *min_coins_greedy(C,D)* included in the starter code and described in class, which uses a greedy algorithm to find the minimum number of coins with denominations D to give C cents. However, in this version, as in real life, we don't have an unlimited number of coins of each denomination, instead, we the number of coins available is given by list *Max*, where *Max[i]* is the number of coins of denomination $D[i]$ available. Unfortunately, the function provided sometimes gives erroneous results. Fix it so it works correctly.