CSE 100: Algorithm Design and Analysis
Midterm 3

Spring 2021

- This is a take-home exam with no proctoring. You can start at any time once you get access to this exam. You have to submit your solution by **4:30pm, 22-APR**, through CatCourses (Midterm 3 under Assignments). You can resubmit any number of times until the deadline. If there are any technical issues with uploading, it is extremely important to immediately contact the instructor or the TA.
- This is an open-book exam. The **only** resources you can use during the exam are all **course materials** uploaded to CatCourses plus the **textbook**. In particular, this means that you are NOT allowed to search for solutions on the internet. No calculator is allowed. You have to take the exam by yourself.
- There are 8 problems in total. You can earn some partial points if you show progress even if you can't solve problems completely.
- Please make your answers concise as well as precise. If there is something in the question that you believe is open to interpretation, then please go ahead and interpret, but state your assumptions in your answer.
- If you have questions, you can email the instructor and the TAs. However, we may not answer your questions on time as we can't be available all the time.

You're required to take the following honor pledge prior to taking the exam.

> *By completing this exam, I acknowledge and confirm that I will not give or receive any unauthorized assistance on this examination. I will conduct myself within the guidelines of the university academic integrity guidelines.*

1. (14 points) For each of the following claims, determine if it is true or false. *No* explanation is needed.

   (a) (2 points) In the tree representation corresponding to a Huffman code, there are at least two leaves of the highest depth.

   (b) (2 points) Let G be an undirected, weighted graph. Multiplying every edge weight by 2 does not change the shortest paths in G.

   (c) (2 points) A fixed-length code is prefix-free.

   (d) (2 points) Suppose in some graph G, the closest negative cycle to a node v has all its nodes at least 5 hops away from v. Consider any node x, whose shortest path from v has fewer than 5 hops. Bellman-Ford computes $d(v; x)$ correctly.

   (e) (2 points) If we are only interested in computing the length of LCS of two sequences of lengths m and n, we can compute it in $O(mn)$ time using $O(m + n)$ space.

   (f) (2 points) Suppose G has $\Theta(n)$ edges with integer weights on the range $O(n^3)$. The runtime of $Kruskal's$ and $Prim's$ algorithm will be asymptotically equivalent.

   (g) (2 points) The Huffman code algorithm can be implemented using priority queue. If we use binary heap to implement a min-priority queue, the Huffman code algorithm can be implemented in $O(n \log n)$ time.

2. (14 points) In the LCS problem, we are given as input two sequences, $X = \langle x_1, x_2, \ldots, x_m \rangle$ and $Y = \langle y_1, y_2, \ldots, y_n \rangle$ and would like to find a longest subsequence common to both. Towards this end, we defined $c[i, j]$ to be the length of LCS of X_i and Y_j, where $X_i := \langle x_1, x_2, ...x_i \rangle$ and $Y_j := \langle y_1, y_2, ...y_j \rangle$.

   (a) (4 points) Give a recursion for computing $c[i, j]$. Do not forget the base cases.

   (b) (10 points) Fill out the the following empty LCS DP table of entries $c[i, j]$.

i	j	0	1	2	3	4
		y_j	A	B	B	A
0	x_i					
1	A					
2	C					
3	B					
4	A					
5	B					

3. (14 points) Huffman Code. Suppose we have a text consisting only of a, b, c, d, e, f where each character appears with the following frequency:

a	b	c	d	e	f
5	6	9	2	4	1

   (a) (10 points) Show the code built by the Huffman algorithm, *both as a tree and as a list* (character, codeword). When combining two trees, *the tree with lowest root frequency becomes the left child* and the tree with the second-lowest root frequency becomes the right child. Left children are associated with the bit 0, right children with the bit 1.

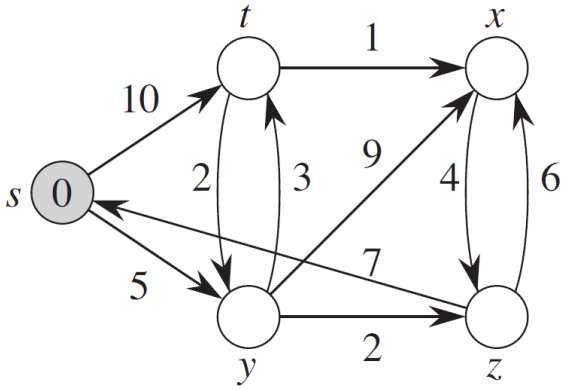
   (b) (2 points) How many bits are required to represent the input using this Huffman code?

   (c) (2 points) How many bits are required to represent the input using a fixed-length code? Of course, you want to use as few bits as possible.

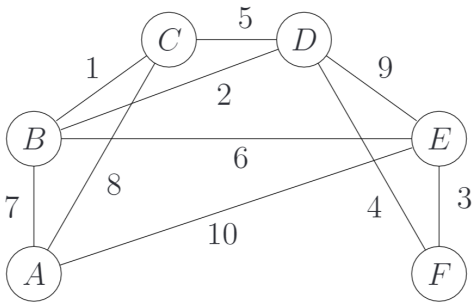
4. (17 points) Dijkstra shortest path algorithm. Do the following:

   (a) (7 points) Write the pseudocode of the Dijkstra algorithm.

   (b) (10 points) For the following graph, assuming you start from node s, calculate the shortest path distances between s and the rest of the vertices. Assume that the array $d[y]$ has the shortest distance values for every single vertice destination in alphabetical order, i.e., t, x, y, z. Show the intermediate values of the array at *each* iteration of the loop in the algorithm.



5. (12 points). Consider the following weighted undirected graph.



For each of the following two algorithms, show the minimum spanning tree and list the edges appearing in the MST *in the order they are added*. For example, the first edge added by Kruskal's algorithm is (B, C). Also provide the weight of the MST found.

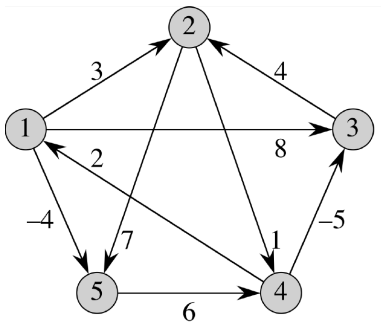
(a) (6 points) Kruskal's algorithm.

(b) (6 points) Prim's algorithm using A as initial vertex.

6. (5 points) The following is a pseudocode of a bottom-up DP algorithm for computing nth Fibonacci number. Give a pseudocode of a *top-down* DP algorithm for the problem, which uses memoization.

```
int F(n)
  Array A[0 ... n]
  A[0] = 0, A[1] = 1
  for i = 2; i <= n ; i++
     A[i] = A[i-1] + A[i-2]
  return A[i]
```

7. (10 points) Run the Floyd-Warshall algorithm on the weighted, directed graph shown below. Show the matrix $D^{(k)}$ that results for each iteration of the outer loop.



	1	2	3	4	5
1					
2					
3					
4					
5					

Table 1: $D^{(0)}$

	1	2	3	4	5
1					
2					
3					
4					
5					

Table 2: $D^{(1)}$

	1	2	3	4	5
1					
2					
3					
4					
5					

Table 3: $D^{(2)}$

	1	2	3	4	5
1					
2					
3					
4					
5					

Table 4: $D^{(3)}$

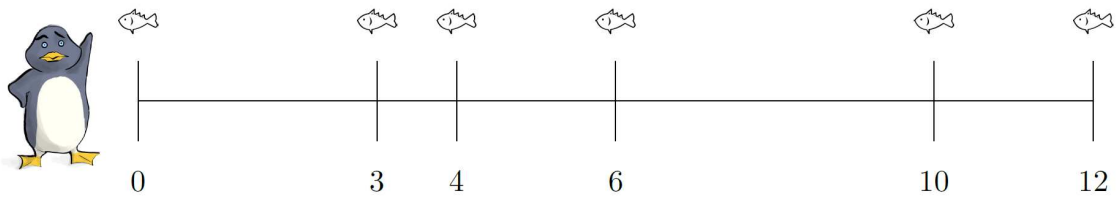
	1	2	3	4	5
1					
2					
3					
4					
5					

Table 5: $D^{(4)}$

	1	2	3	4	5
1					
2					
3					
4					
5					

Table 6: $D^{(5)}$

8. (14 points) Plucky the Pedantic Penguin is walking t miles across Antarctica. He needs to eat along the way, but he can only eat when theres a fishing hole for him to catch fish. He can walk at most m miles between meals, and there are n fishing holes along his route.
Plucky is given an array F so that $F[i]$ gives the distance from the start of his journey to the ith fishing hole. There are n fishing holes along the way, including at the beginning (so F[1] = 0 miles) and the end (F[n] = t miles). For example, the array F = [0, 3, 4, 6, 10, 12], with t = 12 corresponds to the setup below:



Plucky wants to stop as few times as possible, given that he can walk at most m miles without eating. (It is okay if he walks exactly m miles between meals). He starts out hungry, so he will always fish at 0 miles; he will also always fish at his destination (at t miles), whether or not hes hungry. In the example above, if $m = 4$, then Plucky should stop 5 times (including his stops at the beginning and the end), for example at 0, 4, 6, 10, 12 miles.
Plucky decides to use the following greedy algorithm:

```
scheduleFishStops( F, m, t ):
  n = length(F); assert F[1] = 0 and F[n] = t
  fishStops = [ F[1] ]
  lastMeal = F[1]
  for i = 2,...,n-1:
    if F[i] - lastMeal <= m and F[i+1] - lastMeal > m:
      fishStops.append( F[i] )
      lastMeal = F[i]
  if t - lastMeal > m:
    return "No way to make it there"
  else:
    fishStops.append(t)
    return fishStops
```

That is, Plucky will hold out for as long as he can, and only stop to fish if he wont be able to make it to the next stop. In the example above, he will stop at 0, 4, 6, 10, 12 miles.
In this problem, you will prove rigorously, by induction, that this strategy is correct.
Formally, say that an array S of length r is a **feasible schedule** if S is a sorted array containing r elements of F so that $S[1] = 0, S[r] = t$, and for all $i \in \{2, ..., r\}, S[i] - S[i-1] \leq m$. In this problem you will prove the following claim:

> **Claim.** Suppose that F is an array of n strictly increasing positive integers, so that $F[1] = 0$ and $F[n] = t$, and so that for all $i \in \{2, ..., n\}, F[i] - F[i-1] \leq m$. Then $scheduleFishStops(F, m, t)$ returns a shortest feasible schedule.

The point of this problem is to demonstrate that you can write a rigorous proof by induction.

(a) (4 points) State an inductive hypthesis for your proof by induction.

(b) (1 point) Prove the base case.

(c) (8 points) Prove the inductive step. Make sure you state explicitly what you are proving.

(d) (1 point) Prove the conclusion. That is, show that if your inductive argument succeeds, then it implies that **Claim** that you are trying to prove.